Conférence d'apprentissage

CAp 2005

dans le cadre de la plate-forme de l'AFIA
(Association Française pour l'Intelligence Artificielle)
du 30 mai au 3 juin 2005
à Nice

Président du comité de programme

François Denis

Président de la plate-forme AFIA : Fabien Gandon
Site Web de la plate-forme : http ://www-sop.inria.fr/acacia/afia2005/welcome.html

Préface

Ce volume contient les Actes de la 7ème Conférence d'Apprentissage (CAp) qui s'est tenue à Nice du mercredi 1er au vendredi 3 juin 2005. CAp, rendez-vous annuel des chercheurs de la communauté francophone du domaine, s'est déroulée au sein de la plate-forme organisée par l'Association Française d'Intelligence Artificielle (AFIA), comme chaque année de millésime impair. Ce regroupement donne l'occasion d'échanges entre participants à trois conférences : CAp, IC (Ingéniérie des connaissances) et RJCIA (Rencontre des Jeunes chercheurs en Intelligence artificielle).

Nous avons reçu 51 propositions d'articles cette année, ce qui constitue un nombre record depuis la création de CAp en 1999, attestant du dynamisme des chercheurs français en apprentissage automatique. Trois autres indicateurs me semblent témoigner de la bonne santé de notre thématique de recherche :

- *le nombre et la diversité des laboratoires et organismes de rattachement des auteurs* : CRIL, ENST Paris, ETIS, EURISE, GRAPPA, GREYC, IMAG, INRA, INRIA Rhône-Alpes, INRIA Sophia Antipolis, INSA Lyon, INSA-Rouen, INSERM, IRISA-ENSSAT, LENA, LMGM, LIFO, Lim&Bio, LIF, LIRIS, LIRMM, LORIA, LRI, LTCI, UTC, X, Xerox, univ. Montréal, Australian National Univ., Univ. Cath. de Louvain, Univ. de Tunis, ...
- *la multiplicité des thèmes traités* : inférence grammaticale, théorie de l'apprentissage, modèles probabilistes, règles d'association et treillis de Galois, méthodes à noyaux, réseaux bayésiens, algorithmes génétiques, optimisation, contraintes, ... et d'autres encore sortant des thèmes usuels ;
- *la répartition équilibrée des travaux sur un axe théorie-applications*, illustrant le lien marqué en apprentissage automatique entre la performance des méthodes et leurs fondements théoriques.

Chaque article a été évalué par trois relecteurs. Sur 51 soumissions, le comité de programme a retenu 22 articles en version longue (exposé de 25 mn pendant la conférence et article de 16 pages dans les Actes) et 14 autres en version courte (exposé de 5 minutes, article de 2 pages et poster : 11 auteurs ont accepté cette possibilité).

Je tiens à remercier vivement les membres du comité de programme et plus généralement tous les relecteurs pour la très grande qualité de leurs rapports à la fois exigeants et constructifs ainsi que Fabien Torre pour le très efficace logiciel de gestion qu'il a mis à ma disposition. Merci également aux organisateurs de la plate-forme et à Fabien Gandon au premier chef. Je remercie tout particulièrement Cécile Capponi, Yann Esposito et Liva Ralaivola pour le sérieux coup de main qu'ils m'ont donné au quotidien et en particulier pour la composition des présents Actes.

François DENIS,
Président du comité de programme de CAP 2005,
LIF, Université de Provence

Comité de programme :

Président : François Denis (LIF, Université de Provence)

- Florence d'Alché (LAMI, Université d'Evry)
- Frédéric Alexandre (LORIA, Nancy)
- Yoshua Bengio (DIRO, Montréal)
- Marc Bernard (EURISE, Saint-Etienne)
- Olivier Bousquet (Pertinence, Paris)
- Laurent Bréhelin (LIRMM, Montpellier)
- Stéphane Canu (PSI, INSA Rouen)
- Yann Chevaleyre (LIP6, Paris)
- Francois Coste (IRISA, Rennes)
- André Elisseeff (IBM, Zurich)
- Annie Foret (IRISA, Rennes)
- Remi Gilleron (GRAPPA, Lille)
- Yves Grandvalet (Heudiasyc, Université de Technologie de Compiègne)
- Yann Guermeur (LORIA, Nancy)
- Jean-Christophe Janodet (EURISE, Saint-Etienne)
- Frederic Koriche (LIRMM, Montpellier)
- Laurent Miclet (IRISA/ENSSAT, Lannion)
- Remi Munos (CMAP, Polytechnique)
- Claire Nedellec (MIG-INRA, Jouy-en-Josas)
- Engelbert Mephu Nguifo (CRIL, Lens)
- Richard Nock (GRIMAAG, Martinique)
- Liva Ralaivola (LIF, Université de Provence)
- Céline Robardet (INSA, Lyon)
- Céline Rouveirol (LRI, Orsay)
- Marc Sebban (EURISE, St-Etienne)
- Olivier Teytaud (LRI, Orsay)
- Marc Tommasi (GRAPPA, Lille)
- Véronique Ventos (LRI, Orsay)
- Jean-Philippe Vert (Ecole des Mines, Paris)
- Christel Vrain (LIFO, Orléans)
- Jean-Daniel Zucker (LIM&BIO, Paris 13)

Autres relecteurs :

Erick Alphonse (INRA, Jouy-en-Josas), Denis Béchet (LINA, Nantes), Sadok Ben Yahia (DSI, Tunis), Armelle Brun (LORIA, Nancy), G. Cleuziou (LIFO, Orléans), Rémi Coletta (LIRMM, Montpellier), Rémi Coulom (LIFL, Lille), Mélanie Courtine (Lim&Bio, Paris), Hugues Delalin (CRIL, Lens), Corneliu Hennegar, (Lim&Bio, Paris), Ingrid Jacquemin, (IRISA, Rennes), Marie Lahaye (IRISA, Rennes), Pierre-Alain Laur (GRIMAAG, Martinique), Michel Liquiere (LIRMM, Montpellier), Myriam Maumy, (IRMA, Strasbourg), Philippe Preux (LIFL, Lille), René Quiniou, (IRISA, Rennes), Valérie Renault (LIUM, Le Mans), Henry Soldano (LIPN, Paris), Frédéric Sur (LORIA, Nancy), Isabelle Tellier (LIFL, Lille), Fabien Torre (LIFL, Lille), Hélène Touzet (LIFL, Lille)

Table des matières

Théorie de l'apprentissage

Optimisation

Bio-Informatique

Coupling Maximum Entropy and Probabilistic Context-Free Grammar Models for XML Annotation of Documents

Boris Chidlovskii[1], Jérôme Fuselier[1,2]

[1] Xerox Research Centre Europe,
6, chemin de Maupertuis, 38240 Meylan, France
{chidlovskii,fuselier}@xrce.xerox.com

[2] Université de Savoie - Laboratoire SysCom,
Domaine Universitaire, 73376 Le Bourget-du-Lac, France
jerome.fuselier@univ-savoie.fr

Abstract : We consider the problem of semantic annotation of semi-structured documents according to a target XML schema. The task is to annotate a document in a tree-like manner where the annotation tree is an instance of a tree class defined by DTD or W3C XML Schema descriptions. In the probabilistic setting, we cope with the tree annotation problem as a *generalized probabilistic context-free parsing* of an observation sequence where each observation comes with a probability distribution over terminals supplied by a probabilistic classifier associated with the content of documents. We determine the most probable tree annotation by maximizing the joint probability of selecting a terminal sequence for the observation sequence and the most probable parse for the selected terminal sequence.

Nous considérons le problème de l'annotation sémantique de documents semi-structurés guidée par un schéma xml cible. Le but est d'annoter un document de façon arborescente où l'arbre d'annotation est l'instance d'une DTD ou d'un schéma W3C XML. Avec notre approche probabiliste, nous traitons le problème de l'annotation comme une *généralisation de la dérivation de grammaires hors-contextes probabilistes* pour des séquences d'observations. Chaque observation possède une distribution de probabilités sur les classes qui est fournie par un classifieur probabiliste associé au contenu du document. L'arbre d'annotation le plus probable est choisi en maximisant la probabilité jointe de la séquence d'observations et de l'arbre de dérivation associé à cette séquence.

Mots-clés : Apprentissage artificiel, Web sémantique, Extraction d'informations

1 Introduction

The future of the World Wide Web is often associated with the Semantic Web initiative which has as a target a wide-spread document reuse, re-purposing and exchange,

achieved by means of making document markup and annotation more machine-readable. The success of the Semantic Web initiative depends to a large extent on our capacity to move from *rendering-oriented* markup of documents, like PDF or HTML, to *semantic-oriented* document markup, like XML and RDF.

In this paper, we address the problem of *semantic annotation of HTML documents according to a target XML schema*. A tree-like annotation of a document requires that the annotation tree be an instance of the target schema, described in a DTD, W3C XML Schema or another schema language. Annotation trees naturally generalize flat annotations conventionally used in information extraction and wrapper induction for Web sites.

The migration of documents from rendering-oriented formats, like PDF and HTML, toward XML has recently become an important issue in various research communities (Christina Yip Chung, 2002; Curran & Wong, 1999; Kurgan *et al.*, 2002; Saikat Mukherjee, 2003; Skounakis *et al.*, 2003a). The majority of approaches either make certain assumptions about the source and target XML documents, like a conversion through a set of local transformations (Curran & Wong, 1999), or entail the transformation to particular tasks, such as the semantic annotation of dynamically generated Web pages in news portals (Saikat Mukherjee, 2003) or the extraction of logical structure from page images (Skounakis *et al.*, 2003a).

In this paper, we consider the general case of tree annotation of semi-structured documents. We make *no assumptions about the structure of the source and target documents* or their possible similarity. We represent the document content as a sequence of observations $\mathbf{x} = \{x_1, \ldots, x_n\}$, where each observation x_i is a content fragment. In the case of HTML documents, such a fragment may be one or multiple leaves, often surrounded with rich contextual information in the form of HTML tags, attributes, etc. The tree annotation of sequence $\mathbf{x}$ is given by a pair $(\mathbf{y}, d)$, where $\mathbf{y}$ and d refer to leaves and internal nodes of the tree, respectively. The sequence $\mathbf{y} = \{y_1, \ldots, y_n\}$ can be seen, on one side, as labels for observations in $\mathbf{x}$, and on the other side, as a terminal sequence for tree d that defines the internal tree structure over $\mathbf{y}$ according to the target XML schema.

In supervised learning, the document annotation system includes selecting the tree annotation model and training the model parameters from a training set S given by triples $(\mathbf{x}, \mathbf{y}, d)$. We adopt a probabilistic setting, by which we estimate the probability of an annotation tree $(\mathbf{y}, d)$ for a given observation sequence $\mathbf{x}$ and address the problem of finding the pair $(\mathbf{y}, d)$ of maximal likelihood.

We develop a modular architecture for the tree annotation of documents that includes two major components. The first component is a probabilistic context-free grammar (PCFG) which is a probabilistic extension to the corresponding (deterministic) XML schema definition. The PCFG rules may be obtained by rewriting the schema's element declarations (in the case of a DTD) or element and type definitions (in the case of a W3C XML Schema) and the rule probabilities are chosen by observing rule occurrences in the training set, similar to learning rule probabilities from tree-bank corpora for NLP tasks. PCFGs offer the efficient inside-outside algorithm for finding the most probable parse for a given sequence $\mathbf{y}$ of terminals. The complexity of the algorithm is $O(n^3 \cdot |N|)$, where n is the length of sequence $\mathbf{y}$ and $|N|$ is the number of non-terminals on the

PCFG.

The second component is a probabilistic classifier for predicting the terminals y for the observations x_i in $\mathbf{x}$. In the case of HTML documents, we use the maximum entropy framework (Berger *et al.*, 1996), which proved its efficiency when combining content, layout and structural features extracted from HTML documents for making probabilistic predictions $p(y)$ for x_i.

With the terminal predictions supplied by the content classifier, the tree annotation problem represents *the generalized case of probabilistic parsing*, where each position i in sequence $\mathbf{y}$ is defined not with a specific terminal, but with a terminal probability $p(y)$. Consequently, we consider the sequential and joint evaluations of the maximum likelihood tree annotation for observation sequences. In the joint case, we develop a generalized version of the inside-outside algorithm that determines the *most probable annotation tree* $(\mathbf{y}, d)$ according to the PCFG and the distributions $p(y)$ for all positions i in $\mathbf{x}$. We show that the complexity of the generalized inside-outside algorithm is $O(n^3 \cdot |N| + n \cdot |T| \cdot |N|)$, where n is the length of $\mathbf{x}$ and $\mathbf{y}$, and where $|N|$ and $|T|$ are the number of non-terminals and terminals in the PCFG.

We also show that the proposed extension of the inside-outside algorithm imposes the *conditional independence requirement*, similar to the Naive Bayes assumption, on estimating terminal probabilities. We test our method on two collections and report an important advantage of the joint evaluation over the sequential one.

2 XML annotation and schema

XML annotations of documents are trees where inner nodes determine the tree structure, and the leaf nodes and tag attributes refer to the document content. XML annotations can be abstracted as the class T of *unranked labeled rooted trees* defined over an alphabet Σ of tag names (Neven, 2002). The set of trees over Σ can be constrained by a *schema* D that is defined using DTD, W3C XML Schema or other schema languages.

DTDs and an important part of W3C XML Schema descriptions can be modeled as extended context-free grammars (Papakonstantinou & Vianu, 2000), where regular expressions over alphabet Σ are constructed by using the two basic operations of concatenation $\cdot$ and disjunction $|$ and with occurrence operators $*$ (Kleene closure), $?$ ($a? = a|\epsilon$) and $+$ ($a+ = a \cdot a*$). An *extended context free grammar* (ECFG) is defined by the 4-tuple $G = (T, N, \mathcal{S}, R)$, where T and N are disjoint sets of terminals and non-terminals in Σ, $\Sigma = T \cup N$; $\mathcal{S}$ is an initial nonterminal and R is a finite set of production rules of the form $A \rightarrow \alpha$ for $A \in N$, where α is a regular expression over $\Sigma = T \cup N$. The language $L(G)$ defined by an ECFG G is the set of terminal strings derivable from the starting symbol S of G. Formally, $L(G) = \{w \in \Sigma^* | S \Rightarrow w\}$, where $\Rightarrow$ denotes the transitive closure of the derivability relation. We represent as a *parse tree* d any sequential form that reflects the derivational steps. The *set of parse trees* for G forms the set $\mathcal{T}(G)$ of unranked labeled rooted trees constrained with schema G.

In practical cases, we deal with the structure of the documents which can be represented by DTD. Moreover, as DTD are strictly less powerful than CFG, we can work

with CFG or its probabilistic extension, given the fact that it will always be possible to map our results to DTD.

2.1 Tree annotation problem

When annotating HTML documents accordingly to a target XML schema, the main difficulty arises from the fact that the source documents are essentially layout-oriented, and the use of tags and attributes is not necessarily consistent with elements of the target schema. The irregular use of tags in documents, combined with complex relationships between elements in the target schema, makes the manual writing of HTML-to-XML transformation rules difficult and cumbersome.

In supervised learning, the content of source documents is presented as a sequence of observations $\mathbf{x} = \{x_1, \ldots, x_n\}$, where any observation x_i refers to a content fragment, surrounded by rich contextual information in the form of HTML tags, attributes, etc. The tree annotation model is defined as a mapping $X \rightarrow (Y, D)$ that maps the observation sequence **x** into a pair $(\mathbf{y}, d)$ where $\mathbf{y}$=$\{y_1, \ldots, y_n\}$ is a terminal sequence and d is a parse tree of $\mathbf{y}$ according to the target schema or equivalent PCFG G, $\mathcal{S} \Rightarrow \mathbf{y}$. The training set S for training the model parameters is given by a set of triples $(\mathbf{x}, \mathbf{y}, d)$.

To determine the most probable tree annotation $(\mathbf{y}, d)$ for a sequence $\mathbf{x}$, we attempt to maximize the joint probability $p(\mathbf{y}, d|\mathbf{x}, G)$, given the sequence $\mathbf{x}$ and PCFG G. Using the Bayes theorem, we have

$$p(\mathbf{y}, d|\mathbf{x}, G) = p(d|\mathbf{y}, x, G) \cdot p(\mathbf{y}|\mathbf{x}, \mathbf{G}), \tag{1}$$

As d is independent of x given $\mathbf{y}$ and $\mathbf{y}$ is independent of G, we can rewrite the equation 1 as follows :

$$p(\mathbf{y}, d|\mathbf{x}, G) = p(d|\mathbf{y}, G) \cdot p(\mathbf{y}|\mathbf{x}), \tag{2}$$

where $p(\mathbf{y}|\mathbf{x})$ is the probability of terminal sequence $\mathbf{y}$ for the observed sequence $\mathbf{x}$, and $p(d|\mathbf{y}, G)$ is the probability of the parse d for $\mathbf{y}$ according the PCFG G. The most probable tree annotation for $\mathbf{x}$ is a pair $(\mathbf{y}, d)$ that maximizes the probability in (2),

$$(\mathbf{y}, d)_{max} = \underset{(y,d)}{argmax}\, p(d|\mathbf{y}, G) \cdot p(\mathbf{y}|\mathbf{x}). \tag{3}$$

In the following, we build a probabilistic model for tree annotation of source documents consisting of two components to get the two probability estimates in (3). The first component is a probabilistic extension of the target XML schema; for a given terminal sequence $\mathbf{y}$, it finds the most probable parse $p(d|\mathbf{y}, G)$ for sequences according to the PCFG G, where rule probabilities are trained from the available training set. The second component is a probabilistic content classifier C, it estimates the conditional probabilities $p(y|x_i)$ for annotating observations x_i with terminals $y \in T$. Finally, for a given sequence of observations $\mathbf{x}$, we develop two methods for finding a tree annotation $(\mathbf{y}, d)$ that maximizes the joint probability $p(\mathbf{y}, d|\mathbf{x}, G)$ in (2).

The figure 1 outlines the tree annotation problem with the decomposition in two components.

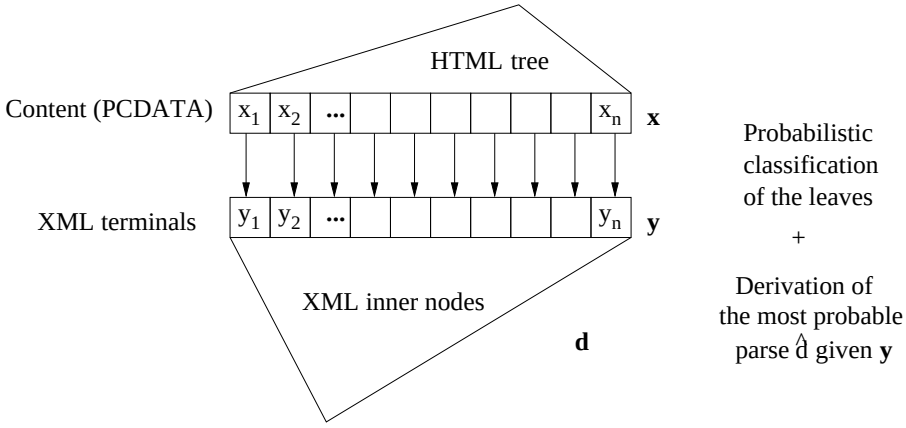


Figure 1: HTML to XML conversion schema.

3 Probabilistic context-free grammars

PCFGs are probabilistic extensions of CFGs, where each rule $A \rightarrow \alpha$ in R is associated with a real number p in the half-open interval (0; 1]. The values of p obey the restriction that for a given non-terminal $A \in N$, all rules for A must have p values that sum to 1,

$$\forall A \in N : \sum_{r=A\rightarrow\alpha, r\in R} p(r) = 1. \quad (4)$$

PCFGs have a normal form, called the Chomsky Normal Form (CNF), according to which any rule in R is either $A \rightarrow B\ C$ or $A \in b$, where A, B and C are non-terminals and b is a terminal. The rewriting of XML annotations requires the *binarization* of source ranked trees, often followed by an extension of the nonterminal set and the underlying set of rules. This is a consequence of rewriting nodes with multiple children as a sequence of binary nodes. The binarization rewrites any rule $A \rightarrow B\ C\ D$ as two rules $A \rightarrow BP$ and $P \rightarrow C\ D$, where P is a new non-terminal.

A PCFG defines a joint probability distribution over Y, a random variable over all possible sequences of terminals, and D, a random variable over all possible parses. Y and D are clearly not independent, because a complete parse specifies exactly one or few terminal sequences. We define the function $p(\mathbf{y}, d)$ of a given terminal sequence $\mathbf{y} \in Y$ and a parse $d \in D$ as the product of the p values for all of the rewriting rules $R(\mathbf{y}, d)$ used in $\mathcal{S} \Rightarrow \mathbf{y}$. We also consider the case where d does not actually correspond to $\mathbf{y}$,

$$p(\mathbf{y}, d) = \begin{cases} \prod_{r\in R(\mathbf{y},d)} p(r), & \text{if } d \text{ is a parse of } y \\ 0, & \text{otherwise.} \end{cases}$$

The values of p are in the closed interval [0; 1]. In the cases where d is a parse of $\mathbf{y}$, all $p(r)$ values in the product will lie in the half open interval (0; 1], and so will the

product. In the other case, 0 is in [0; 1] too. However, it is not always the case that $\sum_{d,\mathbf{y}} p(\mathbf{y}, d) = 1$.

The training of a PCFG takes as evidence the corpus of terminal sequences $\mathbf{y}$ with corresponding parses d from the training set S. It associates with each rule an expected probability of using the rule in producing the corpus. In the presence of parses for all terminal sequences, each rule probability is set to the expected count normalized so that the PCFG constraints (4) are satisfied:

$$p(A \rightarrow \alpha) = \frac{count(A \rightarrow \alpha)}{\sum_{A \rightarrow \beta \in R} count(A \rightarrow \beta)}.$$

3.1 Generalized probabilistic parsing

PCFGs are used as probabilistic models for natural languages, as they naturally reflect the "deep structure" of language sentences rather than the linear sequences of words. In a PCFG language model, a finite set of words serve as a terminal set and production rules for non-terminals express the full set of grammatical constructions in the language. Basic algorithms for PCFGs that find the most likely parse d for a given sequence $\mathbf{y}$ or choose rule probabilities that maximize the probability of sentence in a training set, represent (Lari & Young, 1990) efficient extensions of the Viterbi and Baum-Welsh algorithms for hidden Markov models.

The tree annotation model processes sequences of observations $\mathbf{x} = \{x_1, \ldots, x_n\}$ from the infinite set X, where the observations x_i are not words in a language (and therefore terminals in T) but complex instances, like HTML leaves or groups of leaves.

Content fragments are frequently targeted by various probabilistic classifiers that produce probability estimates for labeling an observation with a terminal in T, $p(y|x_i)$, $y \in T$, where $\sum_y p(y|x_i) = 1$. The tree annotation problem can therefore be seen as *a generalized version of probabilistic context-free parsing*, where the input sequence is given by the probability distribution over a terminal set and the most probable annotation tree requires maximizing the joint probability in (3).

A similar generalization of probabilistic parsing takes place in speech recognition. In the presence of a noisy channel for speech streams, parsing from a sequence of words is replaced by parsing from a word lattice, which is a compact representation of a set of sequence hypotheses, given by conditional probabilities obtained by special acoustic models from acoustic observations (Hall & Johnson, 2003).

4 Content classifier

To produce terminal estimates for the observations x_i, we adopt the maximum entropy framework, according to which the best model for estimating probability distributions from data is the one that is consistent with certain constraints derived from the training set, but otherwise makes the fewest possible assumptions (Berger *et al.*, 1996). The distribution with the fewest possible assumptions is one with the highest entropy, and closest to the uniform distribution. Each constraint expresses some characteristic of the training set that should also be present in the learned distribution. The constraint is

based on a binary feature, it constrains the expected value of the feature in the model to be equal to its expected value in the training set.

One important advantage of maximum entropy models is their flexibility, as they allow the extension of the rule system with additional syntactic, semantic and pragmatic features. Each feature f is binary and can depend on $y \in T$ and on any properties of the input sequence $\mathbf{x}$. In the case of tree annotation, we include the *content features* that express properties on content fragments, like $f_1(x, y)$ ="1 if y is **title** and x's length is less then 20 characters, 0 otherwise", as well as the *structural and layout features* that capture the HTML context of the observation x, like $f_2(x, y)$="1 if y is **author** and x's father is `span`, 0 otherwise".

With the constraints based on the selected features $f(x, y)$, the maximum entropy method attempts to maximize the conditional likelihood of $p(y|x)$ which is represented as an exponential model:

$$p(y|x) = \frac{1}{Z_\alpha(x)} exp\left(\sum_\alpha \lambda_\alpha \cdot f_\alpha(x, y)\right), \tag{5}$$

where $Z_\alpha(x)$ is a normalizing factor to ensure that all the probabilities sum to 1,

$$Z_\alpha(x) = \sum_y exp\left(\sum_\alpha \lambda_\alpha f_\alpha(x, y)\right). \tag{6}$$

For the iterative parameter estimation of the Maximum Entropy exponential models, we have selected one of the quasi Newton methods, namely the Limited Memory BFGS method, which is observed to be more effective than the Generalized Iterative Scaling (GIS) and Improved Iterative Scaling (IIS) for NLP and information extraction tasks (Malouf, 2002).

5 Sequential tree annotation

We use pairs $(\mathbf{x}, \mathbf{y})$ from triples $(\mathbf{x}, \mathbf{y}, d)$ of the training set S to train the content classifier C and pairs $(\mathbf{y}, d)$ to choose rule probabilities that maximize the likelihood for the instances in the training set. C predicts the terminal probabilities $p(\mathbf{y}|\mathbf{x})$ for any observation $\mathbf{x}$, while the inside-outside algorithm can find the parse d of the highest probability for a given terminal sequence $\mathbf{y}$.

By analogy with speech recognition, there exists a naive, sequential method to combine the two components C and G for computing a tree annotation for sequence $\mathbf{x}$. First, from C's estimates $p(\mathbf{y}|\mathbf{x})$, we determine the (top k) most probable sequences $\mathbf{y}_{max,j}$ for $\mathbf{x}$, $j = 1, \ldots, k$. Second, we find the most probable parses for all $\mathbf{y}_{max,j}$, $d_{max,j} = \underset{d}{argmax}\, p(d|\mathbf{y}_{max,j}, G)$; and finally, we choose the pair $(\mathbf{y}_{max,j}, d_{max,j})$ that maximizes the product $p(\mathbf{y}_{max,j}) \times p(d_{max,j})$.

The sequential method works well if the noise level is low (in speech recognition) or if the content classifier (in the tree annotation) is accurate enough in predicting terminals y for x_i. Unfortunately, it gives poor results once the classifier C is far from 100%

accuracy in $\mathbf{y}$ predictions, as it faces the impossibility of finding any parse for all the top k most probable sequences $\mathbf{y}_{max,j}$.

Example.

Consider an example target schema given by the following DTD:

```
<!ELEMENT   Book      (author, Section+)>
<!ELEMENT   Section   (title, (para | footnote)+)>
<!ELEMENT   author    (#PCDATA)>
<!ELEMENT   title     (#PCDATA)>
<!ELEMENT   para      (#PCDATA)>
<!ELEMENT   footnote  (#PCDATA)>
```

The reduction of the above schema definition to the Chomsky Normal Form will introduce extra non-terminals, so we get the PCFG $G = (T, N, \mathcal{S}, R)$, where the terminal set is T={`author, title, para, footnote`}, the nonterminal set is N= {Book, Author, SE, Section, TI, ELS, EL}, $\mathcal{S}$= Book, and R includes twelve production rules.

Assume that we have trained the content classifier C and the PCFG G and have obtained the following probabilities for the production rules in R:

(0.3) Book → AU Section	(0.7) Book → AU SE
(0.4) SE → Section Section	(0.6) SE → Section SE
(0.8) Section → TI ELS	(0.2) Section → TI EL
(0.4) ELS → EL EL	(0.6) ELS → EL ELS
(1.0) AU → `author`	(1.0) TI → `title`
(0.8) EL → `para`	(0.2) EL → `footnote`.

Assume now that we test the content classifier C and PCFG G on a sequence of five unlabeled observations $\mathbf{x} = \{x_1, \ldots, x_5\}$. Let the classifier C estimate the probability for terminals in T as given in the following table:

	x_1	x_2	x_3	x_4	x_5
`author`	0.3	0.2	0.1	0.1	0.2
`title`	**0.4**	**0.4**	0.3	0.3	**0.3**
`para`	0.1	0.2	**0.5**	0.2	0.2
`footnote`	0.2	0.2	0.1	**0.4**	0.2

According to the above probability distribution, the most probable terminal sequence $\mathbf{y}_{max}$ is composed of the most probable terminals for all x_i, $i = 1, \ldots, 5$. It is `'title title para footnote title'` with probability $p(\mathbf{y}_{max}) = p(\mathbf{y}_{max}|x) = \Pi_i \cdot p(\mathbf{y}_i^{max}|x_i) = 0.4 \cdot 0.4 \cdot 0.5 \cdot 0.4 \cdot 0.3 = 0.0096$. However, $\mathbf{y}_{max}$ has no corresponding parse tree in G. Instead, there exist two valid annotation trees for $\mathbf{x}$, $(\mathbf{y}_1, d_1)$ and $(\mathbf{y}_2, d_2)$, as shown in Figure 2. In Figure 2.b, the terminal sequence $\mathbf{y}_2$=`'author title para title para'` with the parse d_2=Book(AU SE(Section (TI EL) Section (TI EL))) maximizes the joint probability $p(\mathbf{y}, d|\mathbf{x}, G)$, with $p(\mathbf{y}_2) = 0.3 \cdot 0.4 \cdot 0.5 \cdot 0.3 \cdot 0.2 = 0.0036$, and

$p(d_2)$=p(Book → AU SE) · p(AU → author) ×
p(SE → Section Section) · p(Section → TI EL) ×

p(TI → title) · p(TI → title) ×
p(EL → para) · p(EL → para) ×
p(Section → TI EL)
$=0.7 \cdot 1.0 \cdot 0.4 \cdot 0.2 \cdot 1.0 \cdot 1.0 \cdot 0.8 \cdot 0.8 \cdot 0.2 = 0.007172.$

Jointly, we have $p(\mathbf{y}_2) \times p(d_2) \approx 2.58 \cdot 10^{-5}$. Similarly, for the annotation tree in Figure 2.a, we have $p(\mathbf{y}_1) \times p(d_1) = 0.0048 \cdot 0.0018432 \approx 8.85 \cdot 10^{-6}$.

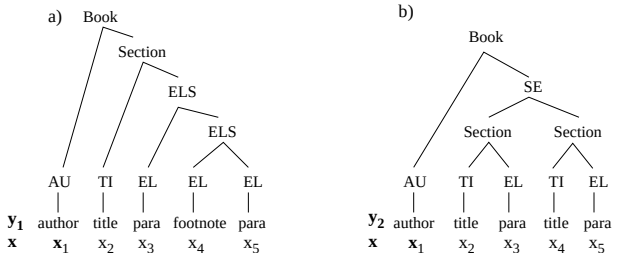


Figure 2: Tree annotations for the example sequence.

6 The most probable annotation tree

As the sequential method fails to find the most probable annotation tree, we try to couple the selection of terminal sequence $\mathbf{y}$ for $\mathbf{x}$ with finding the most probable parse d for $\mathbf{y}$, such that $(\mathbf{y}, d)$ maximizes the probability product $p(d|\mathbf{y}, G) \cdot p(\mathbf{y}|\mathbf{x})$ in (3). To this end, we extend the basic inside-outside algorithm for terminal PCFGs (Lari & Young, 1990). It is a dynamic programming algorithm which is efficient at calculating the best parse tree which yields a given sentence of the PCFG. As in the Viterbi algorithm where only the forward function may be used, we may only use the inside probability. We redefine the *inside probability* as the most probable joint probability of the subsequence of $\mathbf{y}$ beginning with index i and ending with index j, and the most probable partial parse tree spanning the subsequence $\mathbf{y}_i^j$ and rooted at nonterminal A:

$$\beta_A(i,j) = max_{A,\mathbf{y}_i^j}\, p(A_{i,j} \Rightarrow \mathbf{y}_i^j) \cdot p(\mathbf{y}_i^j|\mathbf{x}). \tag{7}$$

The inside probability is calculated recursively, by taking the maximum over all possible ways that the nonterminal A could be expanded in a parse,

$$\begin{aligned}\beta_A(i,j) = \quad & max_{i \le q \le j}\, p(A \rightarrow BC) \cdot p(B \Rightarrow \mathbf{y}_i^q) \times \\ & p(C \Rightarrow \mathbf{y}_{q+1}^j) \cdot p(\mathbf{y}_i^j|\mathbf{x}).\end{aligned}$$

To proceed further, we make the *independence assumption* about $p(\mathbf{y}|\mathbf{x})$, meaning that for any q, $i \le q \le j$, we have $p(\mathbf{y}_i^j|\mathbf{x}) = p(\mathbf{y}_i^q|\mathbf{x}) \cdot p(\mathbf{y}_{q+1}^j|\mathbf{x})$. Then, we can rewrite the above as follows

$$\beta_A(i,j) = max_{i \le q \le j} p(A \rightarrow BC) \cdot p(B \Rightarrow \mathbf{y}_i^q) \times \quad (8)$$

$$p(C \Rightarrow \mathbf{y}_{q+1}^j) \cdot p(\mathbf{y}_i^q|\mathbf{x}) \cdot p(\mathbf{y}_{q+1}^j|\mathbf{x}) \quad (9)$$

$$= max_{i \le q \le j} p(A \rightarrow BC) \cdot \beta_B(i,q) \cdot \beta_C(q+1,j) \quad (10)$$

The recursion is terminated at the $\beta_{\mathcal{S}}(1,n)$ which gives the probability of the most likely tree annotation $(\mathbf{y}, d)$,

$$\beta_{\mathcal{S}}(1,n) = max\ p(\mathcal{S} \Rightarrow \mathbf{y}_1^n) \cdot p(\mathbf{y}_1^n|\mathbf{x}),$$

where n is the length of both sequences $\mathbf{x}$ and $\mathbf{y}$.

The initialization step requires some extra work, as we should select among all terminals in T being candidates for y_k:

$$\beta_A(k,k) = max_{y_k} p(A \rightarrow y_k) \cdot p(y_k|\mathbf{x}). \quad (11)$$

It can be shown that the redefined inside function converges to a local maximum in the (Y, D) space. The extra work during the initialization step takes $O(n \cdot |T| \cdot |N|)$ time which brings the total complexity of the extended IO algorithm to $O(n^3 \cdot |N| + n \cdot |T| \cdot |N|)$.

The independence assumption established above represents the *terminal conditional independence*, $p(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{n} p(y_i|\mathbf{x})$ and matches the Naive Bayes assumption. The assumption is frequent in text processing; it simplifies the computation by ignoring the correlations between terminals. Here however it becomes a requirement for the content classifier. In other words, as far as the PCFG is assumed to capture all (short- and long- distance) relations between terminals, the extended inside algorithm (10)-(11) imposes the terminal conditional independence when building the probabilistic model. This directly impacts the feature selection for the maximum entropy model, by disallowing features that include terminals of neighbor observations y_{i-1}, y_{i+1}, etc, as in the maximum entropy extension with HMM and CRF models (McCallum *et al.*, 2000; Lafferty *et al.*, 2001).

7 Experimental results

We have tested our method for XML annotation on two collections. One is the collection of 39 Shakespearean plays available in both HTML and XML format.[1] 60 scenes with 17 to 189 leaves were randomly selected for the evaluation. The DTD fragment for scenes consists of 4 terminals and 6 non-terminals. After the binarization, the PCFG in CNF contains 8 non-terminals and 18 rules.

The second collection, called TechDoc, includes 60 technical documents from repair manuals. [2] The target documents have a fine-grained semantic granularity and are much deeper than in the Shakespeare collection; the longest document has 218 leaves.

[1] http://metalab.unc.edu/bosak/xml/eg/shaks200.zip.

[2] Available from authors on request.

Method	TechDoc		Shakespeare	
	TER	NER	TER	NER
ME	86.23	–	**100.0**	–
MEMM	78.16	–	99.91	–
Seq-ME-PCFG	86.23	9.38	**100.0**	82.87
Jnt-ME-PCFG	**87.59**	**72.95**	99.97	**99.79**
Jnt-MEMM-PCFG	75.27	56.25	98.09	94.01

Table 1: Evaluation results.

The target schema is given by a complex DTD with 27 terminals and 35 nonterminals. The binarization increased the number of non-terminals to 53. For both collections, a content observation refers to a PCDATA leaf in HTML.

To evaluate the annotation accuracy, we use two metrics. The *terminal error ratio* (TER) is similar to the word error ratio used in natural language tasks; it measures the percentage of correctly determined terminals in test documents. The second metric is the *non-terminal error ratio* (NER) which is the percentage of correctly annotated sub-trees.

As content classifiers, we test with the maximum entropy (ME) classifier which gives us the best results and is the most convenient to use all kind of features. For the ME model, we extract 38 *content features* for each observation, such as the number of words in the fragment, its length, POS tags, textual separators, etc. Second, we extract 14 *layout and structural features* include surrounding tags and all associated attributes. Beyond the ME models, we use the maximum entropy Markov models (MEMM) which extends the ME with hidden Markov structure and terminal conditional features (McCallum *et al.*, 2000). The automaton structure used in MEMM has one state per terminal.

In all tests, a cross-validation with four folds is used. ME and MEMM were first tested alone on both collections. The corresponding TER values for the most probable terminal sequences $\mathbf{y}_{max}$ serve a reference for methods coupling the classifiers with the PCFG. When coupling the ME classifier with the PCFG, we test both the sequential and joint methods. Additionally, we included a special case MEMM-PCFG where the content classifier is MEMM and therefore the terminal conditional independence is not respected.

The results of all the tests are collected in Table 1. The joint method shows an important advantage over the sequential method, in particular in the TechDoc case, where the ME content classifier alone achieves 86.23% accuracy and the joint method reduces the errors in terminals by 1.36%. Instead, coupling MEMM with the PCFG reports a decrease of TER values and a much less important NER increase.

8 Relevant Work

Since the importance of semantic annotation of documents has been widely recognized, the migration of documents from rendering-oriented formats, like PDF and HTML, toward XML has become an important research issue in different research communities (Christina Yip Chung, 2002; Curran & Wong, 1999; Kurgan *et al.*, 2002; Saikat Mukherjee, 2003; Skounakis *et al.*, 2003a). The majority of approaches either constrain the XML conversion to a domain specific problem, or make different kinds of assumptions about the structure of source and target documents. In (Saikat Mukherjee, 2003), the conversion method assumes that source HTML documents are dynamically generated through a form filling procedure, as in Web news portals, while a subject ontology available on the portal permits the semantic annotation of the generated documents.

Transformation-based learning is used for automatic translation from HTML to XML in (Curran & Wong, 1999). It assumes that source documents can be transformed into target XML documents through a series of proximity tag operations, including insert, replace, remove and swap. The translation model trains a set of transformation templates that minimizes an error driven evaluation function.

In document analysis research, Ishitani in (Skounakis *et al.*, 2003a) applies OCR-based techniques and the XY-cut algorithm in order to extract the logical structure from page images and to map it into a pivot XML structure. While logical structure extraction can be automated to a large extent, the mapping from the pivot XML to the target XML schema remains manual.

In natural language tasks, various information extraction methods often exploit the sequential nature of data to extract different entities and extend learning models with grammatical structures, like HMM (McCallum *et al.*, 2000) or undirected graphical models, like Conditional Random Fields (Lafferty *et al.*, 2001). Moreover, a hierarchy of HMMs is used in (Skounakis *et al.*, 2003b) to improve the accuracy of extracting specific classes of entities and relationships among entities. A hierarchical HMM uses multiple levels of states to describe the input on different level of granularity and achieve a richer representation of information in documents.

9 Conclusion

We propose a probabilistic method for the XML annotation of semi-structured documents. The tree annotation problem is reduced to the *generalized probabilistic context-free parsing* of an observation sequence. We determine the most probable tree annotation by maximizing the joint probability of selecting a terminal sequence for the observation sequence and the most probable parse for the selected terminal sequence.

We extend the inside-outside algorithm for probabilistic context-free grammars. We benefit from the available tree annotation that allows us to extend the inside function in a rigorous manner, and avoid the extension of the outside function which might require some approximation.

The experimental results are promising. In future work, we plan to address different challenges in automating the HTML-to-XML conversion. We are particularly interested

in extending the annotation model with the source tree structures that have been ignored so far.

10 Acknowledgement

This work is partially supported by VIKEF Integrated Project co-funded by the European Community's Sixth Framework Programme.

References

BERGER A. L., PIETRA S. D. & PIETRA V. J. D. (1996). A maximum entropy approach to natural language processing. *Computational Linguistics*, **22**(1), 39–71.

CHRISTINA YIP CHUNG, MICHAEL GERTZ N. S. (2002). Reverse engineering for web data: From visual to semantic structures. In *18th International Conference on Data Engineering (ICDE'02), San Jose, California*.

CURRAN J. & WONG R. (1999). Transformation-based learning for automatic translation from HTML to XML. In *Proceedings of the Fourth Australasian Document Computing Symposium (ADCS99)*.

HALL K. & JOHNSON M. (2003). Language modeling using efficient best-first bottomup parsing. In *IEEE Automatic Speech Recognition and Understanding Workshop*, p. 220–228.

KURGAN L., SWIERCZ W. & CIOS K. (2002). Semantic mapping of XML tags using inductive machine learning. In *Proc. of the 2002 International Conference on Machine Learning and Applications (ICMLA'02), Las Vegas, NE*, p. 99–109.

LAFFERTY J., MCCALLUM A. & PEREIRA F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, p. 282–289: Morgan Kaufmann, San Francisco, CA.

LARI K. & YOUNG S. J. (1990). The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, **4**, 35–56.

MALOUF R. (2002). A comparison of algorithms for maximum entropy parameter estimation. In *Proc. 6th Conf. on Natural Language Learning*, p. 49–55.

MCCALLUM A., FREITAG D. & PEREIRA F. (2000). Maximum entropy Markov models for information extraction and segmentation. In *Proc. 17th International Conf. on Machine Learning*, p. 591–598: Morgan Kaufmann, San Francisco, CA.

NEVEN F. (2002). Automata Theory for XML Researchers. *SIGMOD Record*, **31**(3), 39–46.

PAPAKONSTANTINOU Y. & VIANU V. (2000). DTD Inference for Views of XML Data. In *Proc. of 19 ACM Symposium on Principles of Database Systems (PODS), Dallas, Texas, USA*, p. 35–46.

SAIKAT MUKHERJEE, GUIZHEN YANG I. R. (2003). Automatic annotation of content-rich web documents: Structural and semantic analysis. In *International Semantic Web Conference*.

SKOUNAKIS M., CRAVEN M. & RAY S. (2003a). Document transformation system from papers to xml data based on pivot document method. In *Proceedings of the 7th International Conference on Document Analysis and Recognition (ICDAR'03), Edinburgh, Scotland.*, p. 250–255.

SKOUNAKIS M., CRAVEN M. & RAY S. (2003b). Hierarchical hidden markov models for information extraction. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence, Acapulco, Mexico*.

Constrained Sequence Mining based on Probabilistic Finite State Automata*

Stéphanie Jacquemont, François Jacquenet, Marc Sebban

Equipe de Recherche en Informatique de Saint-Etienne
Université Jean Monnet, 23 rue du Dr. Paul Michelon, 42023 Saint-Etienne Cedex 2
{jacqstep,jacquene,sebbanma}@univ-st-etienne.fr

Résumé : Dans ce papier, nous présentons un nouvel algorithme de sequence mining ayant la particularité d'extraire des patterns fréquents sous contraintes à partir d'un automate probabiliste (PDFA). Alors que les PDFAs ont été très peu étudiés jusqu'à présent dans le domaine du sequence mining, nous montrons dans cet article l'avantage d'exploiter une telle représentation condensée, apprise à partir des données, plutôt que de manipuler les séquences elles-mêmes, souvent très (trop) nombreuses. Nous proposons deux types de contraintes afin d'extraire du PDFA des patterns spécifiques permettant ainsi de réduire considérablement l'espace de recherche. Les expériences menées sur des bases de données artificielles et réelles montrent l'efficacité de notre approche.

Mots-clés : sequence mining, automates probabilistes, contraintes.

1 Introduction

The sequence mining task consists in finding patterns, *i.e* sequences of events shared in a database by a large number of instances which can take the form of texts, DNA sequences, web site usage logs, etc. By automatically extracting such *frequent* patterns, one aims at discovering useful knowledge, for example about a particular disease, customer behaviors, network alarms, web site access strategies, etc (Han *et al.*, 2002). We say that a sequential pattern w is *frequent* if the number of sequences of the database that contain w (called *support* or *frequency*) is greater than a threshold given by the user (called the *minimal support*) (Agrawal & Srikant, 1995). In such a context, many sequence mining algorithms have been proposed during the last decade (Agrawal & Srikant, 1995; Srikant & Agrawal, 1996; Mannila *et al.*, 1997; Zaki, 1998). However, over the past few years two new trends seem to independently emerge.

The first one concerns the way the database is scanned while discovering frequent patterns among millions of sequences. Some works have been done, for example (Han

* This work was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778, and by the ACI Masses de Données 2004 Bingo. This publication only reflects the authors' views.

et al., 2000; Pei *et al.*, 2001), in order to build a subtile representation of the database to avoid multiple naive scannings of it. However, in case of huge databases, this *exact* representation of sequences requires all the same high computational and storage costs. Then, rather than building a costly *exact* representation of the data, an other solution would consist in *learning* a more compact summary of the database in the form of a generative model, such as a grammar or an automaton. This idea of using a learned and generalized representation of the sequences has been proposed in (Hingston, 2002). Hingston shows how one can use a probabilistic deterministic finite automata (PDFA) for providing answers efficiently to queries of sequence mining algorithms about frequencies of patterns. He also proposes a sequence mining algorithm for extracting frequent patterns by analyzing the transitions of the PDFA. In this paper, we also claim that a learned PDFA is a suitable and relevant representation of the sequence database to extract knowledge. We provide here interesting improvements of Hingston's approach by introducing *constraints* on the frequent sequences extracted from a PDFA. This concept of constraint is a second current trend in sequence mining.

Actually, despite the use of *minimal support* tuned by the user (that is in practice a tricky task), an *unconstrained* search can produce millions of patterns or may even be intractable. A new recent strategy consists in extracting frequent patterns under constraints: length and width restrictions, minimum or maximum gap between sequence elements, time window of occurrence (Zaki, 2000), or regular expressions (Garofalakis *et al.*, 2002) (see also (Pei *et al.*, 2002)). Moreover, as Zaki claims in (Zaki, 2000), there exist many domains (such as in bio-informatics) where the user may be interested in interactively adding syntactic constraints on the mined sequences. In this paper, we introduce two new constraints that we adapt to the specific context of PDFAs.

The first belongs to the same family of constraints as the one of *time window* proposed in (Zaki, 2000), and consists in extracting only frequent sequences which begin after a given prefix length. To take into account this constraint in the computation of probability estimates, we propose an extension of Hingston's formulae.

The second constraint on the extracted frequent sequences is based on their statistical relevance. Roughly speaking, we mean that a frequent sequence does not always express a significant information: *frequent* does not mean *relevant*? Let us take a simple example for describing that. Two series of experiments A and B are carried out by tossing a coin respectively 10 and 10000 times. Let us assume we obtain respectively 8 and 8000 *tails* (and of course 2 and 2000 *heads*), resulting in two databases of 10 and 10000 sequences (of only one event). We fix the minimal support to 70%, *i.e.* respectively 7 and 7000 sequences. In such a context, the sequence *tails* is frequent in both databases. Does it mean that the knowledge "*tails is frequent in A with a support of* 80%" expresses the same information than "*tails is frequent in B with a support of* 80%"? Absolutely not. Actually, while it is highly probable, with a *non-unbalanced* coin, to obtain more than 70% of *tails* over 10 trials, this event is so improbable over 10000 trials, that it could lead to challenge the balance of the coin itself. We see that for two same frequent patterns deduced from the same relative support, the extracted knowledge (and its consequences !) is very different. Since tuning the minimal support is a tricky task, we propose to contrain a frequent sequence to be also statistically relevant. We introduce in this paper a statistical test-based approach that we adapt to sequence

mining from PDFA.

The rest of this paper is organized as follows. First, after a presentation of the advantages of an automaton-based sequence mining algorithm, we describe Hingston's method. We carry out here a series of experiments that shows the efficiency of a PDFA for estimating true probabilities. In Section 3, we outline the main steps of our methodology by defining our constraints and combining them in a sequence mining algorithm. Section 4 deals with experiments, carried out using the algorithm ALERGIA (Carrasco & Oncina, 1994) for learning the PDFA, that show the efficiency of our approach.

2 On using PDFA for sequence mining

2.1 Advantages of an automaton-based approach

Using a PDFA for mining sequences has not received wide attention. However, the advantages of using such a generalized representation of the sequences are undeniable.

First, as we will see later, it allows us to extract frequent patterns by analyzing the paths of the automaton. However, we must make here an important remark concerning the kind of information we will be able to find from a PDFA. By definition, the problem of mining sequences is to find all frequent patterns in the database according to a given minimal support sup. However, by learning a generalized representation of data in the form of an automaton, and by mining it instead of the input sequences themselves, we can not claim that the extracted sequences occur exactly more than sup times in the database. In other words, since we use a probabilistic representation of the data, we can *only* ensure that the extracted sequences are *probably* frequent. Fortunately, if the size of the database is large enough, the probabilities estimated from the PDFA converge to the ones observed in the set of sequences. In this case, all the extracted sequences from the automaton will be also frequent in the database. To illustrate this phenomenon, we present in the next section some empirical results showing that, when the number of sequences increases, the estimated probabilities computed from the PDFA get closer to the ones issued from the database. Then, they can be used efficiency for discovering knowledge from PDFA.

The second advantage of an automaton-based approach is directly linked to the previous remark. The fact that an learned PDFA is a generalized representation of the input-sequences, using it for sequence mining might result in the discovery of new knowledge. In other words, it is possible to extract patterns that do not occur in the database. This amazing phenomenon is due to the fact that the most of the learning algorithms (such as ALERGIA (Carrasco & Oncina, 1994) that we will use in our experiments) work with a state merging procedure, starting from a prefix tree acceptor and merging states that are judged statistically compatible. From a theoretical standpoint, if the database contains characteristic sequences, it is possible to prove that one can exactly learn the target distribution of the sequences (see theoretical results about learning in the limit (Oncina, 1992; Higuera, 1997)). Let us take a simple example from which one can discover interesting knowledge that does not occur in any sequence of the database. We asked a group of children to geometrically describe a square. We gave them an alphabet of five characteristics: *A: four congruent sides, B: quadrilateral, C: polygon, D: four right angles,*

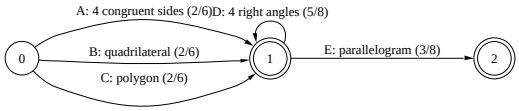


FIG. 1: *An example of* PDFA. *Final states characterizing the end of a sequence are described with a double circle. The observed proportion of sequences which use each transition is indicated between parentheses.*

E: parallelogram. We obtained the 6 following necessary, but insufficient answers, to describe this shape, ordered according to the letters of their components: BD (*quadrilateral* and *four right angles*), CD (*polygon* and *four right angles*), AD (*four congruent sides* and *four right angles*), AE (*four congruent sides* and *parallelogram*), BDE (*quadrilateral, four right angles* and *parallelogram*), CDE (*polygon, four right angles* and *parallelogram*). Let us assume now that we have learned from these sequences the automaton shown in Figure 1.

Beyond of the paths describing the 6 input-sequences, this generalized automaton expresses additional information. Assuming that no sequence uses more than one time the cycle of state 1, the automaton describes also the knowledge ADE (*four congruent sides, four right angles* and *parallelogram*), BE (*quadrilateral* and *parallelogram*) and CE (*polygon* and *parallelogram*). Among them, the first one is the only one which describes perfectly, *i.e. with necessary and sufficient conditions*, a geometrical square. Interestingly, we can observe than this knowledge is also the only one which has been used *in part* by input-sequences. *In part* means that ADE does not occur in the database, but some input sequences have followed the transitions AD, and others the transitions DE. We will see later that this information will be important for estimating the relevance of a frequent pattern.

The model we are going to handle is then a PDFA, which can come from an expert knowledge or from a learning pre-process. Let us now more formally define a PDFA.

Definition 1
A PDFA *is a tuple* $A =< Q, \Sigma, q, q_0, \pi, \pi_F >$ *where:*
- Q *is a finite set of states*
- Σ *is the alphabet*
- $q : Q \times \Sigma \rightarrow Q$ *is an application called transition function*
- q_0 *is the initial state*
- $\pi : Q \times \Sigma \times Q \rightarrow [0, 1]$ *is a probability function which associates at each transition its probability.*
- $\pi_F : Q \rightarrow [0, 1]$ *is a probability function which associates to each state* $S \in Q$ *a probability to be final.*
- A *must be deterministic, that means that* $\forall S \in Q, \forall z \in \Sigma$, *the cardinality of the set* $\{x | q(S, z) = x\}$ *is bounded by 1.*

Since the automaton is deterministic, it results that the two first arguments of the func-

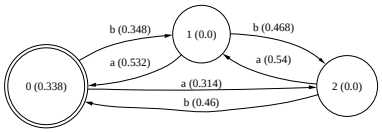


FIG. 2: *An example of* PDFA.

tion π are sufficient to characterize a transition, that means that $\pi(S, z)$ will represent in the following the probability $\pi(S, z, q(S, z))$. Figure 2 shows an example of PDFA where $Q = \{0, 1, 2\}$, $\Sigma = \{a, b\}$, $q(0, a) = 2$ for instance, $q_0 = 0$, and $\pi_F(0) = 0.338$.

2.2 Related work

As we said before, very few work has been done in automata-based sequence mining. As far as we know, the first main work in this domain is probably the one of Hingston (Hingston, 2002), which uses an Apriori-like system for mining sequences from PDFA. He claims that such a strategy can provide answers efficiently to queries about frequencies asked by sequence mining algorithms. One of the objectives of this paper aims at extending Hingston's method to constrained sequence mining. For this reason, we now detail the main steps of his method, consisting in estimating the probability of occurrence of particular patterns. By combining them, it is possible to derive formulae to compute probabilities for any desired ordering of symbols and n-grams.

2.2.1 Estimation of the probability that a sequence contains a symbol x

Given a PDFA $A =< Q, \Sigma, q, q_0, \pi, \pi_F >$. To assess with $\hat{p}(x)$ the true proportion $p(x)$ of sequences of the database that contain the letter x, Hingston defines first the probability $P(S, x)$ that a random path in A starting from state S contains an x. This is ensured either if a path begins with an x (with probability $\pi(S, x)$), or if it begins with some other symbol $z \in Q$ and is followed by a path starting at the next state (given by $q(S, z)$) containing an x. This can be written with the following recursive formula:

$$P(S, x) = \pi(S, x) + \sum_{z \neq x \in \Sigma} (\pi(S, z) \times P(q(S, z), x)) \quad (1)$$

which can be easily rewritten as follows:

$$P(S, x) = \pi(S, x) + \sum_{T \in Q} \left(\sum_{z \neq x, q(S,z)=T} \pi(S, z) \right) \times P(T, x) \quad (2)$$

Obviously, if $S = q_0$, $P(S, x)$ is exactly equal to $\hat{p}(x)$, *i.e.* the probability we are looking for[1]. Compute $P(S, x)$ requires to solve a system of linear equations for which

1. Note that $\hat{p}(x)$ must only depend on prefixes that explains why the probabilities π_F are not used here.

Hingston proposes an efficient solution based on matrix products. He defines the matrix $\rho(x)$ whose components are $\rho_{S,T}(x) = \sum_{z \neq x, q(S,z)=T} \pi(S,z)$. Let $P(x)$ be the vector of values of $P(S,x)$ and $\pi(x)$ the vector of values of $\pi(S,x)$, $\forall S$. Equation 2 becomes:

$$P(x) = \pi(x) + \rho(x)P(x) = (I - \rho(x))^{-1}\pi(x) \quad (3)$$

where I is the identity matrix. Since the matrix $\rho(x)$ and the vector $\pi(x)$ are directly built from the conditional probabilities of the PDFA, the computation of the vector $P(x)$ becomes very easy to achieve.

Example : Let us take again the PDFA of Figure 2 to explain these formulae. We aim at assessing with $\hat{p}(a) = P(0,a)$ the true proportion $p(a)$ of sequences that contain the letter a. In this case, the vector $\pi(a)$ has the following components $\pi(0,a) = 0.314$, $\pi(1,a) = 0.532$, $\pi(2,a) = 0.54$. For the matrix $\rho(a)$, we get:

$$\begin{pmatrix} 0 & 0.348 & 0 \\ 0 & 0 & 0.468 \\ 0.46 & 0 & 0 \end{pmatrix}$$

Computing $(I - \rho(x))^{-1}$, we get

$$\begin{pmatrix} 1.081 & 0.376 & 0.176 \\ 0.233 & 1.081 & 0.506 \\ 0.498 & 0.173 & 1.081 \end{pmatrix}$$

We deduce that $P(x) = (0.635, 0.921, 0.832)$ and that $\hat{p}(a) = P(0,a) = 0.635$. The estimation of the proportion of sequences that contain an a is then equal to 0.635.

2.2.2 Estimation of the probability that a sequence contains a pattern xy

Based on the same principle, Hingston shows that it is possible to estimate the proportion of strings that contain a bi-gram xy. Let $P(S,xy)$ be the probability that a path starting at state S contains the bi-gram xy. One can derive as in the previous section:

$$P(xy) = (I - \rho(x))^{-1}\tau(xy) \quad (4)$$

where $\tau(xy) = \pi(S,x)\pi(q(S,x),y)$.

It is also possible to assess the proportion of strings containing the symbol x, followed later by a y, noted $P(S, < x, y >)$. This quantity can be split in two parts:

- the proportion of strings that contain in a first part the symbol x. One must compute $F(S,T,x)$ that corresponds to the probability that a random path starting at state S and ending at state T contains exactly one x, which is the last symbol on the path (for more details see (Hingston, 2002)),
- the proportion of strings that, in a following part, contain a y (*i.e.* $P(T,y)$).

One can easily deduce that $P(S, < x, y >) = \sum_T F(S,T,x)P(T,y)$.

By combining $P(S,x)$, $P(S,xy)$ and $P(S, < x, y >)$, one can derive formulae to compute probabilities for any desired ordering of symbols and n-grams.

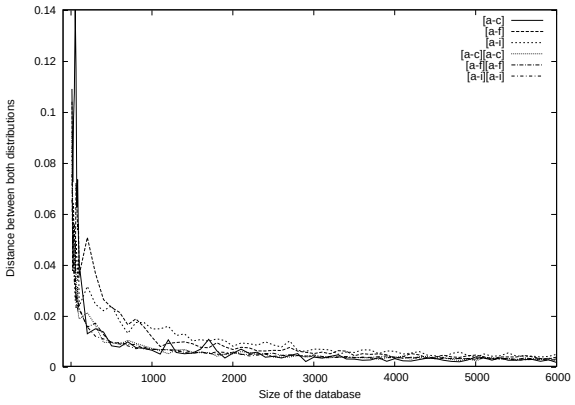


FIG. 3: *Average difference between estimated and observed probabilities of patterns composed of one letter or one bi-gram (according to a given regular expression).*

2.2.3 Experimental results

Hingston claims, despite the fact that no empirical study has been done in (Hingston, 2002), that the estimates $P(S, x)$, $P(S, xy)$ and $P(S, < x, y >)$ converge toward the true proportions in the database when the number of sequences increases. To assess the efficiency of a PDFA to correctly estimate these frequencies, we have implemented Hingston's algorithm and carried out a series of experiments. The experimental setup was the following. We simulated a target distribution from a given alphabet Σ. In order to simplify, this theoretical distribution has been modelized in the form of an automaton with only one state and $|\Sigma|$ output transitions. From this automaton, we generated by sampling learning sets in function of an increasing number of sequences (from 10 to 6000). For each of them, we learned a PDFA using ALERGIA (Carrasco & Oncina, 1994), and then we computed $P(q_0, x)$ and $P(q_0, < x, y >)$[2] and compared them with the true frequencies $p(x)$ and $p(< x, y >)$ observed in the database. Figure 3 shows the behavior of the average difference between the estimated and the true frequencies. We can observe that in all cases the difference converges toward 0 while the size of the set of sequences increases. We can then confirm that, at the limit, a PDFA is able to correctly estimate the true proportion of patterns in the database.

Before concluding this section, note that Hingston proposed a sequence mining algorithm using the previous estimates (for more details see (Hingston, 2002)). Moreover, his method has been efficiently used in a new classification rule induction algorithm proposed in (Psomopoulous *et al.*, 2004) in the particular case of proteins, for which patterns are not ordered. Finally, we also dealt with the extraction of decision rules from non-probabilistic finite automata in (Jacquenet *et al.*, 2004). However, this work was limited to the discovery of rules located at the end of the input-sequences.

2. Since the behavior of $P(S, xy)$ follows the one of $P(S, x)$, we did not carry out experiments for it.

3 On using PDFA for constrained sequence mining

In the following, we improve Hingston's approach by constraining the sequence mining algorithm to discover *particular* frequent patterns. Two types of constraints are proposed. The first one allows us to discover frequent patterns according to a given prefix-length. This constraint is interesting in domains, such as in bio-informatics, where the location in the sequence (and not only the frequency) can express the meaning of the pattern. In order to take into account this constraint in a PDFA-based mining algorithm, and then to permit us the computation of the estimates, we must adapt Hingston's formulae to this new framework. This is the goal of the following subsection. Then, we present our second constraint based on the statistical inference theory. This constraint aims at extracting among the frequent patterns only the ones that are statistically relevant. Adapted in this article to the context of PDFA, they can be obviously used by standard sequence mining algorithms.

3.1 A prefix length constraint

We aim at modifying $P(S, x)$ and $P(S, xy)$ to take into account a prefix length constraint of size δ, that can result in an interesting reduction of the search space. Let us recall that $P(S, x)$ (resp. $P(S, xy)$) is an estimation of the proportion of sequences that contain, from state S, the symbol x (resp. the bi-gram xy). We want to extend them respectively to $P(S, x, \delta)$ and $P(S, xy, \delta)$, *i.e* we are looking for the proportion of sequences that contain the symbol x (resp. the pattern xy) after a prefix of length δ.

Note we will not extend the probability $P(S, < x, y >)$, that explains why we did not enter in the previous section in the details of the calculation of the function $F(S, T, x)$. We think that imposing a prefix length constraint in this context would not be relevant. Actually, in such frequent patterns, the most important information is that the symbol y occurs *after* (the location is here not important) the symbol x.

3.1.1 From $P(S, x)$ to $P(S, x, \delta)$

First, we are interesting in converting $P(S, x)$ into $P(S, x, \delta)$ for taking into account the constraint δ. A component $P(S, x, \delta)$ of the vector $P(x, \delta)$ corresponds to the proportion of sequences of the database containing the symbol x at a distance δ from the state S. Note that it means that x can also occur before the distance δ.

Let us take again the example of Figure 2. If we assume that we are looking for the proportion $P(0, a, 2)$ of sequences containing the letter a in third position, we can establish using Hingston's notations that:

$$\begin{aligned} P(0,a,2) &= \pi(0,b)\times\pi(1,b)\times\pi(2,a) + \pi(0,b)\times\pi(1,a)\times\pi(0,a) \\ &\quad +\pi(0,a)\times\pi(2,a)\times\pi(1,a) + \pi(0,a)\times\pi(2,b)\times\pi(0,a) \\ &= 0.348\times 0.468\times 0.54 + 0.348\times 0.532\times 0.314 \\ &\quad +0.314\times 0.54\times 0.532 + 0.314\times 0.46\times 0.314 \\ &= \sum_{z\in\Sigma} \pi(0,z)\times P(q(0,z),a,1) = 0.282. \end{aligned}$$

Generalizing, and given a PDFA $A =< Q, \Sigma, q, q_0, \pi, \pi_F >$, we get

$$P(S, x, \delta) = \sum_{z \in \Sigma} \pi(S, z) \times P(q(S, z), x, \delta - 1) \tag{5}$$

which can be rewritten as follows

$$P(S, x, \delta) = \sum_{T \in Q} \left(\sum_{z, q(S,z)=T} \pi(S, z) \right) \times P(T, x, \delta - 1). \tag{6}$$

Since an x can also occur before the distance δ (that means that the constraint $z \neq x$ does not exist), we can not use the matrix $\rho(x)$ previously proposed by Hingston in Equation 3. Let us introduce the following matrix μ which is now *independent* of x.

$$\mu_{S,T} = \sum_{z, q(S,z)=T} \pi(S, z).$$

We can rewrite Equation 6 as

$$P(S, x, \delta) = \sum_{T \in Q} \mu_{S,T} \times P(T, x, \delta - 1). \tag{7}$$

Let $P(x, \delta)$ be the vector of values of $P(S, x, \delta)$, Equation 7 becomes:

$$P(x, \delta) = \mu \times P(x, \delta - 1). \tag{8}$$

This is a geometric series of common ratio μ and first term $P(S, x, 0) = \pi(S, x)$. Writing $\pi(x)$ for the vector of values of $\pi(S, x)$, we obtain:

$$P(x, \delta) = \mu^{\delta} \times \pi(x). \tag{9}$$

All the components of the matrices μ and $\pi(x)$ are directly obtained from the conditional probabilities of the PDFA. Of course, as for Hingston, we are only interested in the case of $S = q_0$, which estimates the probability that a sequence contains a prefix of length δ before the occurence of x.

As in Section 2.2.3, we carried out a series of experiments to verify if our extended formula $P(S, x, \delta)$ correctly estimates, for different values of δ, the observed frequencies in the database. Figure 4 shows that the convergence is reached very quickly.

3.1.2 From $P(S, xy)$ to $P(S, \omega, \delta)$

The second objective consists in extending the probability of occurrence of a bi-gram xy by taking into account the prefix-length constraint. In fact, we propose here to directly generalize to a pattern w, in the form of a n-gram, at a distance δ from S.

Definition 2
A pattern $w = (w_1, ..., w_k)$ is an ordered set of k symbols $w_i \in \Sigma, \forall i = 1, \ldots, k$.

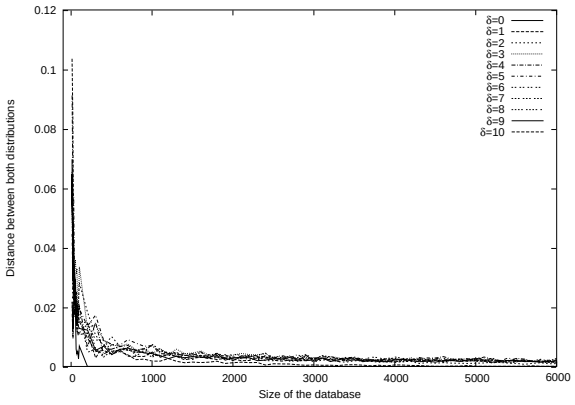


FIG. 4: *Average difference, in function of the sequence set size, between estimated and observed probabilities of patterns composed of one letter, for different values of δ.*

First, we have to generalize the function $\tau(S, xy)$ (Equation 4) to a pattern w. We get, $\tau(S, w_1 \dots w_n) = \pi(S, w_1) \times \tau(q(S, w_1), w_2 \dots w_n)$. Then,

$$P(S, w, \delta) = \sum_{z \in \Sigma} (\pi(S, z) \times P(q(S, z), w, \delta - 1)) \tag{10}$$

$$P(S, w, 0) = \pi(S, w_1) \times \pi(q(S, w_1), w_2) \times \ldots = \tau(S, w) \tag{11}$$

Using exactly the same principle (with the matrix μ) as the one for a single letter, and using vectors, we get:

$$P(w, \delta) = \mu^{\delta} \times \tau(w)$$

3.2 Statistical relevance constraints

After a first constraint based on a prefix length, we introduce in this section a second type of constraints based on the need of relevance of the extracted frequent patterns. As we saw in introduction with the example of a coin toss, a frequent pattern (according to a given minimal support) can be statistically irrelevant. We show here that the relevance can be assessed using statistical tests applied on the proportions estimated from the PDFA. In this section, we propose two statistical tests, that we call *relevance constraints*, to validate *step by step* the symbols of a *relevant* frequent pattern.

The first *relevance* constraint allows us to verify a first *absolute* condition. Let $w = (w_1 ... w_l)$ be a current *relevant* pattern of size l in the PDFA, at a distance δ from the initial state. The relevance of an additional $l + 1^{th}$ symbol w_{l+1} will be ensured if the proportion of sequences that contain the pattern $w' = (w_1 ... w_{l+1})$ at a distance δ from the initial state (estimated by $P(q_0, (w_1 ... w_{l+1}), \delta)$) covers a significant part of the probability density of all sequences.

The second constraint expresses a *relative* condition. Given a current relevant frequent pattern $w = (w_1 ... w_l)$ and an additional symbol w_{l+1} satisfying the first relevance

constraint. The proportion of sequences that satisfy w at a distance δ from the initial state must be approximately the same as the one of the new pattern $w' = (w_1...w_{l+1})$.

According to the two previous conditions, we are going to define *recursively* the notion of relevance (of a symbol or of a pattern). For this reason, let us assume in the following that we have already a frequent and relevant pattern.

3.2.1 Relevance constraint of a symbol

Definition 3
Let A be a PDFA *in which the pattern $w = (w_1...w_l)$ is frequent and relevant at a distance δ from q_0. Let w_{l+1} be a new symbol which, concatenated with w, makes a new pattern $w' = (w_1...w_{l+1})$. w_{l+1} is statistically relevant for w' iff $P(q_0, (w_1...w_{l+1}), \delta)$ is significantly higher than 0, using a test of proportion.*

The notion of *significance* is defined by a test of proportion (called PROPORTION_TEST) aiming at verifying if $P(q_0, (w_1...w_{l+1}), \delta)$ is high enough, *i.e.* if the new resulting pattern $w' = (w_1...w_{l+1})$ statistically covers a sufficient part of the probability density of the input-sequences. For simplifying the notations, let us consider, without any loss of generality, that $P(q_0, (w_1...w_{l+1}), \delta) = \hat{p}(w', \delta)$. We are interested in verifying if this estimate of the true proportion $p(w', \delta)$ (the proportion of input-sequences containing the pattern w' after a prefix of length δ) is relevant. To achieve this task, we test the null hypothesis $H_0 : p(w', \delta) = 0$, against the alternative one $H_a : p(w', \delta) > 0$.

If the number of input-sequences is high enough, the statistic $\hat{p}(w', \delta)$ asymptotically follows the normal law. We have then to determine the threshold k which defines the bound of rejection of H_0, and which corresponds to the $(1 - \alpha)$-percentile (U_α)of the distribution of $p(w', \delta)$ under H_0. It is easy to show that $P(\hat{p}(w', \delta) > k) = \alpha$ iff

$$k = U_\alpha \sqrt{\frac{\hat{p}(w', \delta)(1 - \hat{p}(w', \delta))}{n}}$$

where n represents the number of sequences in the database. The decision rule is the following: if $\hat{p}(w', \delta) > k$, the constraint of relevance of the symbol w_{l+1} is satisfied.

Note that in the previous definition, we assumed that we had a current relevant frequent pattern w. Of course, from an algorithmic standpoint, we will initialize w, during a first step, to the empty string, w' containing only the additional symbol w_{l+1}. Such a manner to proceed will allow us to find the first set of relevant patterns with the PROPORTION_TEST. Let us take an example. Figure 5 shows a PDFA with 6 states, where $\Sigma = \{a, b, c\}$, built from a set of 100 sequences. Given the pattern $w' = (a)$, and a given minimal support of 40%, w' is then considered as *frequent* at a distance 0 from the initial state 0, because $P(0, a, 0) > 0.4$. Is its new (and unique here) component a also relevant? With a risk $\alpha = 2.5\%$, $U_\alpha = 1.96$ and $k = 0.088$. Since $P(0, a, 0) > k$, the symbol a is relevant for the pattern w'. Since $w' = (a)$, we can also deduce here that w' is relevant.

3.2.2 Conditionally relevance constraint of a pattern

However, we think that an unique constraint on each additional symbol is not sufficient to accept $w' = (w_1...w_{l+1})$ as being a relevant pattern. Actually, we would like

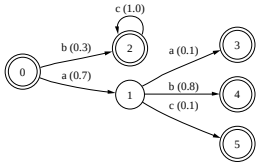


FIG. 5: *An example of* PDFA *with 6 states built from* $\Sigma = \{a, b, c\}$.

also to verify if there exists a statistical dependence in the PDFA between the previous pattern $w = (w_1...w_l)$ and the new one $w' = (w_1...w_{l+1})$. Roughly speaking, we mean that the high majority of the sequences that contained w must also satisfy w'. This is what we call a *conditionally constraint*.

Definition 4
Let A be a PDFA *in which the pattern $w = (w_1...w_l)$ is frequent and relevant at a distance δ from q_0. Let w_{l+1} be a new symbol which, concatenated with w, makes a new pattern $w' = (w_1...w_{l+1})$. w' is statistically relevant conditionally to w iff w' is significantly dependent of w, using a Chi-Square test.*

This dependence can be assessed by analyzing the nature of all the different symbols occurring in the PDFA after the pattern w. Consider again the example of Figure 5 for which we have already validated the relevance of the pattern $w' = (a)$ at a distance $\delta = 0$ from the initial state 0. According to the conditional probabilities of this PDFA, we could wonder if the pattern $w' = (ab)$ is also relevant. To deal with this problem, we must achieve two tasks. First, the PROPORTION_TEST must be run to verify if the additional symbol b is relevant for w'. In this case, with $\alpha = 2.5\%$ and $U_\alpha = 1.96$, $\hat{p}(w', 0) = P(0, ab, 0) = \pi(0, a).\pi(1, b) = 0.56$ and $k = 0.083$. Since $\hat{p}(w', 0) > k$, the additional symbol b is relevant for w'. Second, we must now verify if there exists a statistical dependence between $w = (a)$ and $w' = (ab)$. In order to carry out this task, we generate an output vector $\overrightarrow{V_{output}}$ of dimension m, where m corresponds to the number of different outgoing transitions from the states in which the last symbol w_l of w ends. In our example, $\overrightarrow{V_{output}}$ has three components $\overrightarrow{V_{output}}(z), z = 1, ..., 3$ because there is a set $O = \{a, b, c\}$ of three outgoing transitions from the state 1. Each component $\overrightarrow{V_{output}}(z)$ corresponds to the expected number of times the symbol z follows the pattern w in the database LS, that means that $\overrightarrow{V_{output}}(z) = P(q_0, z, \delta) \times |LS|$. We arrange the vector $\overrightarrow{V_{output}}$ such that the considered symbol w_{l+1} (here b) is the first component of the vector (the order of the other components does not matter). In our example, $\overrightarrow{V_{output}} = (56, 7, 7)$.

We aim now at testing the dependence between $\overrightarrow{V_{output}}$ and an input vector $\overrightarrow{V_{input}}$ for which the first component is the expected number of times the pattern w occures in the database (the other components are null). Then, $\overrightarrow{V_{input}} = (70, 0, 0)$. From the two vectors $\overrightarrow{V_{input}}$ and $\overrightarrow{V_{output}}$, we run a test of independence based on a Chi-square test

(called CHI_SQUARE_TEST). We build the following statistic X^2, such that:

$$X^2 = \sum_{z \in O} \frac{[(\overrightarrow{V_{input}}(z) - \Psi(z))^2 + (\overrightarrow{V_{output}}(z) - \Psi(z))^2]}{\Psi(z)}$$

where $\Psi(z) = \frac{\overrightarrow{V_{input}}(z) + \overrightarrow{V_{output}}(z)}{2}$ is the average vector of $\overrightarrow{V_{input}}$ and $\overrightarrow{V_{output}}$. From a statistical standpoint, X^2 follows a Chi-square distribution with $2 \times m - 1$ degrees of freedom. It is then possible to test if X^2 is higher than $X_\alpha^{2\times m-1}$, which is the $(1-\alpha)$-percentile of the Chi-square distribution. The decision rule is the following: if $X^2 < X_\alpha^{2\times m-1}$, the dependence between w and w' is ensured and the conditionally relevant constraint is verifyied. By combining the two previous relevance constraints, we are now able to define what we call a frequent and relevant pattern.

Definition 5
Let A be a PDFA *in which the pattern $w = (w_1...w_l)$ is frequent and relevant at a distance δ from q_0. Let w_{l+1} be a new symbol which, concatenated with w, makes a new pattern $w' = (w_1...w_{l+1})$. w' is frequent and statistically relevant at a distance δ from q_0 iff (i) $P(q_0, w', \delta)$ is higher than the minimal support, (ii) the new symbol w_{l+1} is statistically relevant for w' and (iii) w' is statistically relevant conditionally to w.*

3.3 A new sequence mining algorithm

Combining all the concepts we presented before, we propose a new sequence mining algorithm. It aims at discovering from a PDFA all frequent and statistically relevant patterns in the form of n-grams, according to a minimal support σ and a given prefix length constraint δ. Of course, we can also run it several times with different values of δ to avoid to have to fix in advance a given prefix length without any knowledge about the studied domain. Let us recall that our algorithm only extracts relevant frequent n-grams. However, it is possible to use our relevance constraints on the probabilities $P(S, < x, y >)$ (with a possible gap between x and y) estimated in Hingston's algorithm.

The pseudo-code of our algorithm ACSM (for Automata-based Constrained Sequence Mining) is presented in Algorithm 1. During a first step (from lines 2 to 9), it initializes a first set G of relevant frequent patterns composed of only one symbol. Since no patterns have been extracted yet, only the support test (line 4) and the PROPORTION_TEST (line 5) are run. The paths of the PDFA that do not satisfy these two tests will not be studied anymore, that will allow us to dramatically reduce the search space. The second part of our algorithm tests additionnal symbols to search for larger frequent relevant patterns. In this case, three conditions must be satisfyed: the support test (lines 17), the PROPORTION_TEST (line 18) and the CHI_SQUARE_TEST (line 19).

4 Experimental results

We experimentally observed in the previous sections the ability of a PDFA to correctly estimate the true proportions of patterns in a database. For this reason, the only main

Algorithm 1: ACSM

Input: A PDFA A = (Q, Σ, q ,q_0, π, π_F), a support threshold σ, two risks α_1 and α_2, δ a prefix length constraint
Output: a set of relevant frequent patterns G

begin
  $G_1 \leftarrow \emptyset$;
  foreach $l \in \Sigma$ **do**
    if $P(q_0, l, \delta) \geq \sigma$ **then**
      if `Proportion_Test` $(P(q_0, l, \delta), \alpha_1)$ *is satisfied* **then**
        $G_1 \leftarrow G_1 \cup l^\delta$;
      end
    end
  end
  $G \leftarrow G_1$;
  $n \leftarrow 1$;
  while $G_n \neq \emptyset$ **do**
    $G_{n+1} \leftarrow \emptyset$;
    foreach $\omega^\delta = (w_1 ... w_n)^\delta \in G_n$ **do**
      foreach $l \in \Sigma$ **do**
        $\omega^\delta \leftarrow (w_1 ... w_n l)^\delta$;
        if $P(q_0, \omega^\delta, \delta) \geq \sigma$ **then**
          if `Proportion_Test` $(P(q_0, (\omega_n l), (\delta + n - 1)), \alpha_1)$ *is satisfied* **then**
            if `Chi_square_Test` $(\overrightarrow{V_{input}}, \overrightarrow{V_{output}}, \alpha_2)$ *is satisfied* **then**
              $G_{n+1} \leftarrow G_{n+1} \cup \omega^\delta$;
            end
          end
        end
      end
    end
    $G \leftarrow G \cup G_{n+1}$;
    $n \leftarrow n + 1$;
  end
  return G ;
end

goal of this section is to evaluate the effect of our constraints to reduce the number of extracted frequent patterns. We carried out various series of experiments from two databases. First, we used a real database which corresponds to the set of all female first-names extracted from the french calendar (called FIRSTNAME). Its size is deliberately not large, because we are above all interested, in a first series of experiments, in analyzing the behavior of our constraints on small datasets. Then, we used the synthetic sequence generator IBM DATAGEN[3] for generating a larger database (called SYND) containing 100000 sequences of average size 10 events. From FIRSTNAME and SYND, we learned two PDFAs using ALERGIA.

The first chart of Figure 6 shows the direct impact of our relevance constraints on the number of frequent patterns extracted from FIRSTNAME. We applied our relevance tests on the estimates of Hingston's algorithm. We observe that incorporating these constraints significantly decreases the number of extracted patterns. Note that when we apply both tests (here with $\alpha_1 = 10\%$ and $\alpha_2 = 50\%$), the number of patterns drops dramatically. Without surprise, while the minimum support increases, the statistical relevance of the extracted patterns is more frequently validated, that explains that all the curves tend to join the one of Hingston.

For the three remaining charts, we used our algorithm ACSM, which requires four

3. http://www.cs.rpi.edu/∼zaki/software/

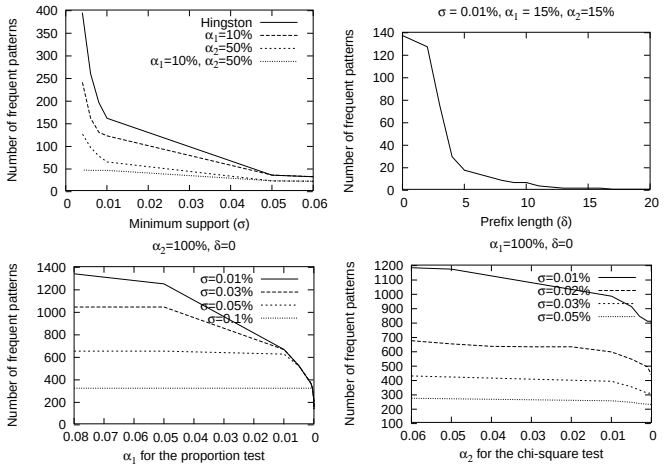


FIG. 6: *Effect of minimum support, prefix length, and relevance constraints on the number of extracted patterns.*

parameters: σ, δ, α_1 and α_2. The objective was to evaluate the effect of each constraint on the number of patterns extracted by our new system. The second chart shows the influence of the prefix length constraint in the case of FIRSTNAME. We can see that strengthening this constraint leads to extracting a decreasing number of patterns. For this experiment we tested various values for σ, α_1 and α_2 and the various charts we got had always the same shape. The two remaining charts of Figure 6 shows the effects of the statistical relevance constraints on SYND. We tested the influence of the relevance constraint of a symbol without incorporating the conditionally relevance constraint of a pattern (*i.e.* fixing α_2 to 100%) and reciprocally (fixing α_1 to 100%). Firstly, we observe again that the stronger one of these constraints is (*i.e.* the lower the α_1 or α_2 parameter is), the more the number of extracted patterns decreases. Secondly, once again, we observe that the more the minimum support constraint increases, the more the relevance constraints become useless.

5 Conclusion

In this paper, we have presented an automata-based approach for constrained sequence mining. We have seen that building a PDFA from the data and then mining that structure presents many advantages compared to more classic sequence mining tools that directly mine the sequences. Our framework extends the one of Hingston in several ways by incorporating a prefix length constraint and two statistical relevance constraints. The experiments we made have shown that those constraints lead to extracting less frequent patterns (but the most relevant ones!) which is really important for the users that are often overwhelmed by huge amount of useless patterns while mining data.

We now want to focus on several points. First we would like to integrate other constraints

in the ACSM algorithm. Then, we plan to study the impact of noisy data on the system and the way it deals with them. We also want to use it in the context of biological data in order to explore the power of our new constraints on such a field of applications.

Références

AGRAWAL R. & SRIKANT R. (1995). Mining sequential patterns. In *ICDE '95: Proceedings of the Eleventh International Conference on Data Engineering*, p. 3–14: IEEE Computer Society.

CARRASCO R. C. & ONCINA J. (1994). Learning stochastic regular grammars by means of a state merging method. In *Proceedings of ICGI'94*, p. 139–152: Springer-Verlag.

GAROFALAKIS M. N., RASTOGI R. & SHIM K. (2002). Mining sequential patterns with regular expression constraints. *IEEE Trans. on Knowledge and Data Eng.*, **14**(3), 530–552.

HAN J., ALTMAN R. B., KUMAR V., MANNILA H. & PREGIBON D. (2002). Emerging scientific applications in data mining. *Comm. of the ACM*, **45**(8), 54–58.

HAN J., PEI J., MORTAZAVI-ASL B., CHEN Q., DAYAL U. & HSU M.-C. (2000). Freespan: Frequent pattern-projected sequential pattern mining. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, p. 355–359.

HIGUERA C. D. L. (1997). Characteristic sets for polynomial grammatical inference. *Mach. Learn.*, **27**(2), 125–138.

HINGSTON P. (2002). Using finite state automata for sequence mining. In *Proceedings of the twenty-fifth Australasian conference on Computer science*, p. 105–110.

JACQUENET F., SEBBAN M. & VALÉTUDIE G. (2004). Mining decision rules from deterministic finite automata. In *Proceedings of the 16th IEEE ICTAI*, p. 362–367.

MANNILA H., TOIVONEN H. & VERKAMO A. I. (1997). Discovery of frequent episodes in event sequences. *Data Min. Knowl. Discov.*, **1**(3), 259–289.

ONCINA, J.; GARCÍA P. (1992). *Identifying regular languages in polynomial time*. World Scientific Publishing. Advances in Structural and Syntactic Pattern Recognition,.

PEI J., HAN J., PINTO H., CHEN Q., DAYAL U. & HSU M.-C. (2001). Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth. In *Proceedings of the International Conference on Data Engineering*, p. 215–224.

PEI J., HAN J. & WANG W. (2002). Mining sequential patterns with constraints in large databases. In *Proceedings of CIKM 2002*, p. 18–25: ACM Press.

PSOMOPOULOUS F., DIPLARIS S. & MITKAS P. (2004). A finite state automata based technique for protein classification rules induction. In *Second European Workshop on Data Mining and Text Mining in Bioinformatics*, p. 58–64.

SRIKANT R. & AGRAWAL R. (1996). Mining sequential patterns: Generalizations and performance improvements. In *EDBT '96: Proceedings of EDBT'96*, p. 3–17: Springer-Verlag.

ZAKI M. J. (1998). Efficient enumeration of frequent sequences. In *Proceedings of CIKM'98*, p. 68–75: ACM Press.

ZAKI M. J. (2000). Sequence mining in categorical domains: incorporating constraints. In *Proceedings of CIKM 2000*, p. 422–429: ACM Press.

Définitions et premières expériences en apprentissage par analogie dans les séquences.

Laurent Miclet, Sabri Bayoudh, Arnaud Delhay

Projet Cordial, IRISA-ENSSAT
6 rue de Kerampont - BP 80518, F-22305 Lannion Cedex, France
{miclet,bayoudh,delhay}@enssat.fr

Résumé : Cet article donne une définition de l'analogie entre séquences fondée sur la distance d'édition et donne deux algorithmes (l'un rapide et approché, l'autre plus complexe, mais optimal) pour calculer la *dissemblance analogique* entre quatre séquences. Cette notion et ces algorithmes sont ensuite utilisés dans une expérience d'apprentissage sur une base de données artificielle, pour montrer la résistance au bruit de ce type d'apprentissage.

Mots-clés : Séquences, Distance d'édition, Analogie, Apprentissage supervisé.

1 Introduction

Cet article présente des définitions, des algorithmes originaux et quelques expériences dans le domaine de l'apprentissage non paramétrique (CORNUÉJOLS & MICLET, 2002) dans l'univers des séquences.

Le principe de l'apprentissage non paramétrique est de ne faire aucune hypothèse sur la distribution statistique des classes, ni d'utiliser d'éventuelles connaissances *a priori*. La technique la plus simple de ce domaine est celle de l'apprentissage par *plus proche voisin*, qui requiert seulement que l'on sache calculer au moins une ressemblance, au mieux une distance, entre les objets. A un objet extérieur à l'ensemble d'apprentissage, cette méthode attribue la classe de l'élément de l'ensemble d'apprentissage qui lui est le plus ressemblant.

Dans la même famille, la technique de l'apprentissage par analogie fait appel à un argumentaire plus sophistiqué. Donnons-en un exemple intuitif. Soit la séquence de lettres `cherchant`, dont on veut apprendre la supervision ou la classe[1] ; supposons que dans l'ensemble d'apprentissage se trouvent les trois séquences : `voler`, `volant`, `chercher`, avec respectivement pour classes *infinitif, participe présent, infinitif*. On attribuera à `cherchant` la classe *participe présent*, par un raisonnement qui s'énonce

1. Le terme *supervision* est plus général que celui d'*étiquette* ou de *classe*, puisqu'il couvre aussi bien les cas de l'apprentissage de règles de classification, de la régression et le cas où la supervision est un objet strucuré, comme une séquence.

informellement comme ceci. Puisque, dans l'univers des séquences

`voler` est à `volant` comme `chercher` est à `cherchant`

la supervision de `cherchant` est donc la solution de l'équation (dans l'univers des classes) :

infinitif est à *participe présent* comme *infinitif* est à x d'où : $x =$*participe présent*

Plan de l'article

Cet article donne dans sa première partie des définitions pour les relations « est à » et « comme », fondées sur la distance d'édition entre séquences. Il définit la notion de *dissemblance analogique* entre quatre séquences, qui permet de quantifier de manière cohérente la relation « comme ». Il propose deux algorithmes, l'un permettant de calculer une approximation de la dissemblance analogique, l'autre sa valeur exacte. Dans sa seconde partie, il propose aussi une première série d'expériences, sur des données artificielles, pour vérifier la validité de ces définitions et l'efficacité de ce type d'apprentissage, en particulier sa résistance au bruit.

2 L'analogie entre séquences

2.1 L'analogie et ses propriétés

2.1.1 Définition de l'analogie

Le *raisonnement par analogie* a été longuement décrit et étudié depuis les philosophes grecs ; ses applications récentes intéressent en particulier les sciences cognitives (SHAVLIK & DIETTERICH, 1990; WILSON & KEIL, 1999), la linguistique et l'intelligence artificielle. Lepage donne dans (LEPAGE, 2003) une histoire encyclopédique de ce concept et de ses applications à la science du raisonnement et à la linguistique.

Le fait que quatre objets A, B, C et D sont en relation d'analogie s'énonce :

"A *est à* B *comme* C *est à* D"

Selon la nature des objets, la signification de cette relation varie. Par exemple, une analogie sur la sémantique de mots français est :

"`jument` *est à* `poulain` *comme* `vache` *est à* `veau`"

En revanche, la relation analogique suivante est purement morphologique :

"`abcd` *est à* `ecd` *comme* `abfab` *est à* `efab`"

Elle ne porte que sur la transformation des lettres dans des séquences :

> Pour transformer `abcd` en `ecd`, exactement comme pour transformer `abfab` en `efab`, il faut remplacer en début de séquence `ab` par `e`.

C'est à ce genre d'analogie dans les séquences que nous nous intéressons. Nous ne traitons pas les analogies du type :

"`ab` *est à* `abab` *comme* `abfg` *est à* `abfgabfg`" ni
"`abc` *est à* `abd` *comme* `aababc` *est à* `aababcd`"

qui ont pourtant toutes les deux de bonnes raisons cognitives d'être considérées comme correctes. L'argument est purement opérationnel : nous utilisons une notion de ressemblance entre séquences qui ne traite pas naturellement ce type de relations. Notre approche est donc limitée du point de vue de la simulation du comportement cognitif.

2.1.2 Propriétés de l'analogie, équations analogiques.

La *relation d'analogie* "A *est à* B *comme* C *est à* D" entre quatre objets sera désormais notée : $A : B :: C : D$

Si un terme de la relation d'analogie est inconnu, on peut considérer qu'on est en présence d'une équation. *Résoudre une équation analogique* consiste à trouver une ou plusieurs valeurs à l'inconnue X pour vérifier la relation (ou à prouver qu'il n'y a pas de solution) : $A : B :: C : X$

Classiquement, l'écriture de la relation analogique $A : B :: C : D$ impose que deux axiomes[2] soient satisfaits. Ils s'expriment comme une équivalence entre la relation de départ et deux autres relations analogiques (LEPAGE & ANDO, 1996) :

Symétrie de la relation comme : $C : D :: A : B$
Echange des termes médians : $A : C :: B : D$

On montre alors facilement que cinq autres équations analogiques se déduisent par équivalence de ces axiomes à partir de $A : B :: C : D$

$B : A :: D : C$ $D : B :: C : A$ $D : C :: B : A$ $B : D :: A : C$ $C : A :: D : B$

Au total, ces axiomes donnent donc pour équivalentes huit équations entre quatre objets. On peut imposer un autre axiome, le *déterminisme* qui exige que l'une des deux équations triviales suivantes n'ait qu'une seule solution[3] :

$$A : A :: B : X \quad \Rightarrow \quad X = B$$
$$A : B :: A : X \quad \Rightarrow \quad X = B$$

On peut maintenant[4] , en se fondant sur les axiomes ci-dessus, donner une définition de la solution à une équation analogique :

2. L'axiome de symétrie, que nous qualifions de « classique » n'est pas dans tous les cas une propriété naturelle de l'analogie. Par exemple, (HOFSTADTER & the Fluid Analogies Research Group, 1994) donne les deux analogies sur les séquences, dans l'alphabet de l'anglais : "`bcd` *est à* `bce` *comme* `xyz` *est à* `wyz`" et "`xyz` *est à* `wyz` *comme* `bcd` *est à* `acd`". Le premier cas, puisque la lettre `z` n'a pas de successeur, fait appel à un « effet de rebond ». Le second se résoud naturellement en utilisant l'ordre des lettres. Mais comme nous utilisons une notion de distance (symétrique) dans les alphabets, nous l'étendons naturellement à la distance entre séquences et à la relation « comme ».

3. L'autre équation en est une conséquence.

4. Il y a 24 façons d'arranger quatre objets en une équation analogique, qui se réduisent à trois classes d'équivalences dont les représentants sont par exemple : $A : B :: C : D$, $A : B :: D : C$ et $A : C :: D : B$

Définition 1
X est une solution correcte *de l'équation analogique* $A : B :: C : X$ *si X est une solution de cette équation et est aussi une solution des deux équations :*

$$C : X :: A : B \quad et \quad A : C :: B : X$$

Une équation analogique peut avoir zéro, une seule ou plusieurs solutions.

2.2 Une définition de l'analogie entre séquences par la distance d'édition : la *dissemblance analogique*.

2.2.1 La distance d'édition entre deux séquences.

Soit Σ un ensemble fini appelé *alphabet*. Les *lettres* $a, b, c, ...$ sont les éléments de Σ. On note $u, v, \ldots$ ou $X, Y, \ldots$ les éléments de Σ^*, appelés *séquences* ou *phrases* ou *mots*. Une séquence $u = u_1 u_2 \ldots u_{|u|}$ est une liste ordonnée de lettres de Σ ; sa *longueur* est notée $|u|$. ϵ, la *séquence vide*, est la séquence de longueur nulle. Si $u = u_1 u_2 \ldots u_{|u|}$ et $v = v_1 v_2 \ldots v_{|v|}$, leur *concaténation* est $uv = u_1 u_2 \ldots u_{|u|} v_1 v_2 \ldots v_{|v|}$.

Pour présenter la distance d'édition entre deux séquences, nous allons rappeller quelques notions et citer un théorème démontré dans (WAGNER & FISHER, 1974). La première notion est celle d'*édition de séquences*. Elle est elle-même fondée sur trois opérations d'*édition de lettres*, ou *transformations élémentaires* : l'*insertion* d'une lettre dans une séquence, la *suppression* d'une lettre et la *substitution* d'une lettre à une autre lettre. A chacune de ces opérations est associé un coût, qui est en général la valeur d'une certaine distance δ sur $(\Sigma \cup \{\epsilon\}) \times (\Sigma \cup \{\epsilon\})$.

Nous notons $S_{a \to b}$ la substitution de a en b (de coût $\delta(a, b)$), $S_{a \to \epsilon}$ la suppression de a (de coût $\delta(a, \epsilon)$) et $S_{\epsilon \to b}$ l'insertion de b (de coût $\delta(\epsilon, b)$). La substitution (ϵ, ϵ) n'est pas considérée comme une opération d'édition. Le *coût de l'édition* d'une séquence est la somme des coûts de l'édition de ses lettres. L'édition d'une séquence de lettres permet donc de la transformer en une autre séquence de lettres par une séquence de transformations élémentaires, avec un certain coût.

Par exemple, on peut passer de la séquence *abc* à la séquence *cd* avec la séquence de transformations élémentaires $S_{a \to \epsilon} S_{b \to c} S_{c \to d}$ de coût $\delta(a, \epsilon) + \delta(b, c) + \delta(c, d)$ ou avec la séquence de transformations élémentaires $S_{a \to \epsilon} S_{b \to \epsilon} S_{c \to c} S_{\epsilon \to d}$ de coût $\delta(a, \epsilon) + \delta(b, \epsilon) + \delta(c, c) + \delta(\epsilon, d)$.

Définition 2
Un alignement *ou* édition *entre deux séquences u et v est une séquence de transformations élémentaires entre les lettres de u et de v. La* distance d'édition $D(u, v)$ *entre deux séquences u et v est le coût de l'alignement de coût minimum parmi tous les alignements possibles entre les deux séquences.*

On représente un alignement par un tableau à deux lignes, une pour u et une pour v, chaque mot complété par des ϵ, les deux mots obtenus ayant la même longueur.

Par exemple, voici les deux alignements précédents entre $u = abc$ et $v = cd$:

$$\begin{array}{ccc} a & b & c \\ | & | & | \\ \epsilon & c & d \end{array} \qquad \begin{array}{cccc} a & b & c & \epsilon \\ | & | & | & | \\ \epsilon & \epsilon & c & d \end{array}$$

L'algorithme de Wagner et Fisher (WAGNER & FISHER, 1974) , dans sa version complète, prend en entrée deux phrases u et v et produit en résultat l'alignement optimal, c'est à dire la séquence optimale de transformations élémentaires et bien sûr son coût (SANKOFF & KRUSKAL, 1983). Nous notons $\mathcal{S}(u, v)$ cette séquence optimale[5].

Une conséquence du calcul par programmation dynamique est le remarquable résultat suivant (CROCHEMORE & *et al.*, 2001), qui justifie le nom de *distance* d'édition:

Théorème 1
Si δ vérifie les axiomes d'une distance[6] *sur $(\Sigma \cup \{\epsilon\})$ alors D, la distance d'édition entre séquences calculée à partir de δ, vérifie aussi les axiomes d'une distance sur Σ^*.*

2.2.2 La relation « est à ».

Nous avons choisi de définir le terme $u : v$ d'une analogie entre séquences comme leur alignement optimal, c'est à dire comme la séquence optimale de transformations calculée par l'algorithme de Wagner et Fisher. Ce choix permet de décrire précisément comment u se transforme en v : il modélise donc bien une relation « est à ».

Par exemple, pour la table de distances :

δ	a	b	c	ϵ
a	0	1.5	1.5	1
b	1.5	0	1.3	1
c	1.5	1.3	0	1
ϵ	1	1	1	1

la transformation optimale (ici unique) entre $u = abbcc$ et $v = bbbc$ est la séquence

$$\mathcal{S}(u, v) = S_{a\to\epsilon} S_{b\to b} S_{b\to b} S_{c\to b} S_{c\to c}$$

de coût

$$\delta(a, \epsilon) + \delta(b, b) + \delta(b, b) + \delta(c, b) + \delta(c, c) = 1 + 0 + 0 + 1.3 + 0 = 2.3$$

Nous posons donc que $\mathcal{S}(u, v) = S_{a\to\epsilon} S_{b\to b} S_{b\to b} S_{c\to b} S_{c\to c}$ définit ce que « u est à v ».

Nous avons déjà noté que cette transformation optimale peut ne pas être unique. Par exemple, en prenant une autre distance δ, telle que $\forall a, b \in (\Sigma \cup \{\epsilon\}), a \neq b, \delta(a, b) = 1$, et $\forall a \in \Sigma, \delta(a, a) = 0$, les transformations entre $u = abbcc$ et $v = bbbc$

$$S_{a\to\epsilon} S_{b\to b} S_{b\to b} S_{c\to b} S_{c\to c}$$

de coût

$$\delta(a, \epsilon) + \delta(b, b) + \delta(b, b) + \delta(c, b) + \delta(c, c) = 1 + 0 + 0 + 1 + 0 = 2$$

5. Elle peut ne pas être unique et nous traiterons ce problème un peu plus loin. Nous la supposons unique pour le moment.

6. Pour être exact, δ est une application de $((\Sigma \cup \{\epsilon\}) \times (\Sigma \cup \{\epsilon\})) \setminus \{(\epsilon, \epsilon)\}$ dans $\mathbb{R}^+$ qui respecte les axiomes d'une distance.

et

$$S_{a\to b}S_{b\to b}S_{b\to b}S_{c\to c}S_{c\to\epsilon}$$

de coût

$$\delta(a,b)+\delta(b,b)+\delta(b,b)+\delta(c,c)+\delta(c,\epsilon)=1+0+0+1+0=2$$

sont toutes les deux optimales. Nous reviendrons plus loin sur cette difficulté.

2.2.3 La relation « comme ».

Soit l'analogie $u : v :: w : x$. Nous connaissons désormais le terme $u : v$, qui est la séquence optimale $\mathcal{S}(u,v)$, supposée pour le moment unique, de transformations élémentaires entre u et v et nous connaissons aussi $w : x$ qui est la séquence $\mathcal{S}(w,x)$.

La relation « exactement comme ».

Examinons d'abord un exemple où la relation « comme » est l'identité, avant de la généraliser. Soit les séquences : $u = abbccccb$, $v = bbbcccc$, $w = abbccaab$ et $bbbcaac$. Le calcul, sur la table des distances donnée pour l'exemple du paragraphe 2.2.2, donne :

$$\mathcal{S}(u,v)=S_{a\to\epsilon}S_{b\to b}S_{b\to b}S_{c\to b}S_{c\to c}S_{c\to c}S_{c\to c}S_{b\to c}$$

$$\mathcal{S}(w,x)=S_{a\to\epsilon}S_{b\to b}S_{b\to b}S_{c\to b}S_{c\to c}S_{a\to a}S_{a\to a}S_{b\to c}$$

Il est naturel de mettre toutes les transformations élémentaires d'égalité, qui doivent être de coût nul, dans une même classe d'équivalence[7] notée S_e. Nous obtenons alors :

$$\mathcal{S}(u,v)=S_{a\to\epsilon}S_eS_eS_{c\to b}S_eS_eS_eS_{b\to c}$$

$$\mathcal{S}(w,x)=S_{a\to\epsilon}S_eS_eS_{c\to b}S_eS_eS_eS_{b\to c}$$

Dans cet exemple, puisque $\mathcal{S}(u,v)=\mathcal{S}(w,x)$, la relation « comme » de l'équation analogique *abbccaab* : *bbbcaac* :: *abbcaabb* : *bbbcaac* est donc une identité. Ce cas particulier est à la base des algorithmes de *résolution d'équations analogiques*, pour lesquels u, v et w sont donnés et x est l'inconnue (LEPAGE, 2003; DELHAY & MICLET, 2004).

Si nous voulons généraliser ce cas particulier, il nous faut définir complètement une distance Δ entre transformations élémentaires. Nous pourrons alors poser la seconde partie de notre construction :

> *La valeur de la relation « comme » est donnée par la distance d'édition, calculée à partir de Δ, entre $\mathcal{S}(u,v)$ et $\mathcal{S}(w,x)$.*

Nous cherchons donc maintenant à définir une distance Δ entre transformations élémentaires sur un alphabet $(\Sigma\cup\{\epsilon\})$, lui-même muni d'une distance δ.

7. Ceci se justifie par le fait que, pour tout a et b, l'équation $a : a :: b : b$ est exacte.

Des contraintes sur Δ.

Pour être cohérent avec les axiomes de l'analogie et la remarque du paragraphe précédent, il faut imposer les conditions suivantes :

– La distance entre deux opérations de transformation identiques est nulle.

$$\forall a, b \in (\Sigma \cup \{\epsilon\}), \Delta(S_{a\to b}, S_{a\to b}) = 0$$

– Insérer ou supprimer des opérations d'égalités de lettres se fait à coût nul[8].

$$\forall a, b \in (\Sigma \cup \{\epsilon\}), \Delta(-, S_{a\to a}) = \Delta(S_{a\to a}, -) = 0$$

– La distance entre deux transformations d'égalité est nulle.

$$\forall a, b \in (\Sigma \cup \{\epsilon\}), \Delta(S_{a\to a}, S_{b\to b}) = 0$$

En effet, si l'une de ces propriétés n'était pas respectée, la distance d'édition obtenue à partir de Δ sur quatre phrases en analogie exacte ne serait pas nulle.

Une proposition pour la distance Δ.

Nous proposons de définir Δ par la formule suivante :

$$\Delta(S_{a\to b}, S_{c\to d}) = Min \begin{cases} \delta(a,b) + \delta(c,d) \\ \delta(a,c) + \delta(b,d) \end{cases}$$

Cette définition remplit les contraintes précédentes et assure que Δ possède la propriété de l'inégalité triangulaire[9] : $\Delta(S_{a\to b}, S_{c\to d}) + \Delta(S_{c\to d}, S_{e\to f}) \geq \Delta(S_{a\to b}, S_{e\to f})$. Cependant, on n'a pas : $\Delta(S_{a\to b}, S_{c\to d}) = 0 \Leftrightarrow$ *a* : *b* :: *c* : *d*

Fin de la construction de la relation « comme ».

Une fois définie la distance Δ entre séquences d'opérations élémentaires, il est facile d'appliquer à nouveau l'algorithme de Wagner et Fisher avec Δ sur le couple de séquences $\mathcal{S}(u, v)$ et $\mathcal{S}(w, x)$. Le coût obtenu quantifie la relation « comme ».

Définition 3
Nous appelons dissemblance analogique approchée[10] *entre les phrases* u, v, w *et* x, *notée* $\widehat{DA}(u, v, w, x)$, *le coût d'édition obtenu avec la distance* Δ *entre* $\mathcal{S}(u, v)$ *et* $\mathcal{S}(w, x)$. *Ces deux derniers termes représentent les deux séquences optimales*[11] *de transformations élémentaires entre* u *et* v, *d'une part, et entre* w *et* x, *d'autre part, obtenues avec la distance* δ.

8. Le mot vide dans l'alphabet des transformations élémentaires est noté : « $-$ »
9. Nous ne donnons pas la démonstration ici, faute de place. Elle est plutôt technique.
10. La nécessité de cet adjectif sera expliquée au paragraphe 2.2.4
11. Encore une fois, nous supposerons pour le moment ces séquences optimales uniques.

Un exemple.

Pour le tableau de distances :

δ	a	b	c	ϵ
a	0	1.2	1.5	2
b	1.2	0	1.7	2
c	1.5	1.7	0	2
ϵ	2	2	2	2

l'alignement optimal unique entre $u = cbacba$ et $v = babba$ (de valeur 3.7) est :

$$\begin{array}{ccccccccc} u & = & c & b & a & c & b & a \\ & & | & | & | & | & | & | \\ v & = & \epsilon & b & a & b & b & a \end{array}$$

et celui, unique aussi, entre $w = cbacbc$ et $x = bcabbc$ (de valeur 5.1) est :

$$\begin{array}{ccccccccc} w & = & c & b & a & c & b & c \\ & & | & | & | & | & | & | \\ x & = & b & c & a & b & b & c \end{array}$$

et finalement la dissemblance approchée est calculée comme valant 3.7, par l'alignement optimal suivant selon la distance Δ :

$$\begin{array}{ccccccccc} \mathcal{S}(u,v) & = & S_{c\to\epsilon} & S_{b\to b} & S_{a\to a} & S_{c\to b} & S_{b\to b} & S_{a\to a} \\ & & | & | & | & | & | & | \\ \mathcal{S}(w,x) & = & S_{c\to b} & S_{b\to c} & S_{a\to a} & S_{c\to b} & S_{b\to b} & S_{c\to c} \end{array}$$

Le cas où les séquences d'opérations élémentaires ne sont pas uniques.

Nous avons supposé jusqu'ici que, dans le calcul de la relation analogique entre séquences $u : v :: w : x$, l'algorithme de Wagner et Fisher produisait une solution unique avec la distance δ pour calculer l'alignement optimal $\mathcal{S}(u, v)$ entre u et v, comme $\mathcal{S}(w, x)$ entre w et x.

Il est en principe facile de relâcher cette contrainte : notons $\aleph(u, v)$ (respectivement $\aleph(w, x)$) l'ensemble des séquences de transformations élémentaires de coût optimal entre u et v (respectivement entre w et x). Nous pouvons définir l'exactitude de la relation « comme », c'est à dire la distance analogique approchée entre u et v d'une part, w et x d'autre part, comme le coût d'un alignement optimal entre deux séquences de transformations optimales, quand l'une parcourt $\aleph(u, v)$ et l'autre parcourt $\aleph(w, x)$. On effectue donc au pire $|\aleph(u, v)| \times |\aleph(w, x)|$ alignements entre séquences de transformations élémentaires. En pratique, il est possible de réduire les calculs, car les séquences de transformations optimales sont structurées en graphe sans cycle (SANKOFF & KRUSKAL, 1983).

Quelques problèmes avec la dissemblance analogique approchée.

Pour être cohérent avec les axiomes de l'analogie, il serait souhaitable que l'on ait la même dissemblance analogique entre les huit quadruplets de séquences dont l'analogie

se déduit des axiomes, comme on l'a vu au paragraphe 2.1.2. Ce n'est en général pas le cas de la dissemblance analogique approchée (voir le paragraphe 2.2.4).

C'est en particulier pour remédier à cette difficulté que nous allons voir dans le paragraphe suivant comment définir et calculer directement, sur les mêmes principes, ce que nous appelons la *dissemblance analogique* entre quatre séquences sur Σ, une notion complètement cohérente avec les axiomes de l'analogie et répondant de plus à un critère d'optimalité.

2.2.4 Construction simultanée de « est à » et de « comme ». Dissemblance analogique entre quatre séquences.

Les paragraphes précédents ont donné une construction qui permet de mesurer l'exactitude de la relation « comme », après que les séquences de transformations qui définissent les deux relations « est à » aient été construites. On peut se demander si cette construction en deux étapes est optimale : il pourrait exister une séquence de transformations $\mathcal{S}'(u,v)$, de coût supérieur à $\mathcal{S}(u,v)$, et une séquence $\mathcal{S}'(w,x)$, de coût supérieur à $\mathcal{S}(w,x)$, telles que la distance d'édition entre $\mathcal{S}'(u,v)$ et $\mathcal{S}'(w,x)$ soit inférieure à celle entre $\mathcal{S}(u,v)$ et $\mathcal{S}(w,x)$.

Reprenons l'exemple du paragraphe 2.2.3 : $\widehat{DA}(cbacba, babba, cbacbc, bcabbc) = 3.7$. Nous pouvons montrer que la construction de $\widehat{DA}$ en deux étapes séparées n'est en effet pas optimale. Il existe deux couples d'alignements entre u et v d'une part, w et x d'autre part, qui fournissent un meilleur résultat. Le premier n'est pas optimal (il a une valeur de 5.4), le second est l'alignement optimal (valeur de 5.1).

$$\begin{array}{cccccc} c & b & a & c & b & a \\ | & | & | & | & | & | \\ b & \epsilon & a & b & b & a \end{array} \qquad \begin{array}{cccccc} c & b & a & c & b & c \\ | & | & | & | & | & | \\ b & c & a & b & b & c \end{array}$$

Leur alignement optimal par Δ se fait comme suit pour une valeur de 2 :

$$\begin{array}{cccccc} S_{c\to b} & S_{b\to\epsilon} & S_{a\to a} & S_{c\to b} & S_{b\to b} & S_{a\to a} \\ | & | & | & | & | & | \\ S_{c\to b} & S_{b\to c} & S_{a\to a} & S_{c\to b} & S_{b\to b} & S_{c\to c} \end{array}$$

Ceci amène la définition suivante :

Définition 4
La dissemblance analogique $DA(u,v,w,x)$ *est le coût minimal de la distance d'édition calculée par la distance* Δ *entre* $\mathcal{S}'(u,v)$ *et* $\mathcal{S}'(w,x)$*, quand ces deux termes parcourent l'un tous les alignements entre* u *et* v *et l'autre tous les alignements entre* w *et* x*.*

2.2.5 Propriétés de la dissemblance analogique entre quatre séquences.

Nous avons, grâce à cette nouvelle définition, les résultats suivants [12].

Propriété 1
Symétrie. $\forall (u,v,w,x) \in (\Sigma^*)^4 : DA(u,v,w,x) = DA(w,x,u,v)$

12. Faute de place, nous les donnons ici sans leur démonstration. Il est intéressant de noter qu'ils sont en général faux si l'on remplace DA par $\widehat{DA}$.

Inégalité triangulaire. $\forall(u,v,w,x,z,t) \in (\Sigma^*)^6 : DA(u,v,w,x) \leq DA(u,v,z,t) + DA(z,t,w,x)$

Propriété 2
Echange des médians. $\forall(u,v,w,x) \in (\Sigma^*)^4 : DA(u,v,w,x) = DA(u,w,v,x)$

Non-échange des deux premiers termes. *En général,* $\forall(u,v,w,x) \in (\Sigma^*)^4$ *:*

$DA(u,v,w,x) \neq DA(v,u,w,x)$

Ces résultats assurent donc la cohérence de toute notre construction avec les axiomes de l'analogie[13]. Nous donnons dans le paragraphe suivant son algorithme de calcul.

2.2.6 Algorithmes : résolution d'une équation analogique entre séquences et calcul de la dissemblance analogique entre quatre séquences.

Algorithme de résolution d'une équation analogique entre séquences.

La résolution d'une équation analogique $u : v :: w : x$ consiste à calculer x, connaissant u, v et w. Nous avons traité ce problème dans le cadre de la distance d'édition, en supposant que la relation « comme » est l'identité, dans (MICLET & DELHAY, 2003; DELHAY & MICLET, 2004). Nous avons donné deux algorithmes du même type que ceux traités ici : un approché et un optimal. Une question que nous n'avons pas abordée est l'écriture un algorithme pour résoudre une équation analogique avec une dissemblance non nulle (c'est à dire en levant l'hypothèse que « comme » est l'identité).

Algorithme de calcul de la dissemblance analogique approchée entre quatre séquences.

Cet algorithme a été expliqué ci-dessus : il consiste à calculer l'ensemble des séquences de transformations optimales $\aleph(u,v)$ entre u et v et l'ensemble des séquences de transformations optimales $\aleph(w,x)$ entre w et x, à l'aide de la distance δ. Dans une deuxième phase, il calcule le coût minimal, en utilisant Δ, pour transformer une séquence dans $\aleph(u,v)$ en une séquence dans $\aleph(w,x)$. Sa complexité est au pire en

$$\mathcal{O}\Big(\big(\underbrace{|u|.|v|}_{u\ sur\ v} + \underbrace{|w|.|x|}_{w\ sur\ x} \big) + \underbrace{\big((|u|+|v|).(|w|+|x|)\big)}_{\text{taille des alignements de séq. d'édition}} \times \underbrace{|\aleph(u,v)| \times |\aleph(w,x)|}_{\text{nb. d'alignements de séq. d'édition}} \Big)$$

Algorithme de calcul de la dissemblance analogique entre quatre séquences.

L'algorithme que nous proposons ici est une généralisation de celui de Wagner et Fisher ; il examine les quatre phrases de manière synchrone et cumule le coût optimal de leur dissemblance analogique par programmation dynamique. Rappelons que :

$$\Delta(S_{a\to b}, S_{c\to d}) = Min \begin{cases} \delta(a,b) + \delta(c,d) \\ \delta(a,c) + \delta(b,d) \end{cases}$$

Les données d'entrée de cet algorithme sont donc les quatre phrases et la table des valeurs de la distance δ sur $(\Sigma \cup \{\epsilon\}) \times (\Sigma \cup \{\epsilon\})$. Il donne en sortie la dissemblance

13. Cependant, on n'a pas en général : $\Delta(S_{a\to b}, S_{c\to d}) = 0 \Leftrightarrow\ a : b :: c : d$.

analogique $DA(u,v,w,x)$, qui est aussi égale à $DA(w,x,u,v)$, à $DA(u,w,v,x)$, à $DA(v,u,x,w)$, à $DA(x,v,w,u)$, à $DA(x,w,v,u)$, à $DA(v,x,w,u)$, à $DA(w,u,x,v)$ et à $DA(x,v,w,u)$.

Le calcul se fait par la récurrence suivante :

Initialisation

$C_{w_0x_0}^{u_0v_0} \leftarrow 0$;

pour $i = 1, |u|$ **faire** $C_{w_0x_0}^{u_iv_0} \leftarrow \sum_{k=1}^{k=i} \Delta(S_{u_i\to\epsilon}, S_{\epsilon\to\epsilon})$ **fait** ;

pour $j = 1, |v|$ **faire** $C_{w_0x_0}^{u_0v_j} \leftarrow \sum_{k=1}^{k=j} \Delta(S_{\epsilon\to v_j}, S_{\epsilon\to\epsilon})$ **fait** ;

pour $i = 1, |w|$ **faire** $C_{w_kx_0}^{u_0v_0} \leftarrow \sum_{i=1}^{i=k} \Delta(S_{\epsilon\to\epsilon}, S_{w_k\to\epsilon})$ **fait** ;

pour $j = 1, |x|$ **faire** $C_{w_0x_l}^{u_0v_0} \leftarrow \sum_{k=1}^{k=l} \Delta(S_{\epsilon\to\epsilon}, S_{\epsilon\to x_k})$ **fait** ;

Récurrence

$$C_{w_kx_l}^{u_iv_j} = Min \begin{cases} C_{w_{k-1}x_{l-1}}^{u_{i-1}v_{j-1}} + \Delta(S_{u_i\to v_j}, S_{w_k\to x_l}) & u_i : v_j :: w_k : x_l \\ C_{w_{k-1}x_l}^{u_{i-1}v_{j-1}} + \Delta(S_{u_i\to v_j}, S_{w_k\to\epsilon}) & u_i : v_j :: w_k : \epsilon \\ C_{w_kx_{l-1}}^{u_{i-1}v_{j-1}} + \Delta(S_{u_i\to v_j}, S_{\epsilon\to x_l}) & u_i : v_j :: \epsilon : x_l \\ C_{w_kx_l}^{u_{i-1}v_{j-1}} + \Delta(S_{u_i\to v_j}, S_{\epsilon\to\epsilon}) & u_i : v_j :: \epsilon : \epsilon \\ C_{w_{k-1}x_{l-1}}^{u_iv_{j-1}} + \Delta(S_{\epsilon\to v_j}, S_{w_k\to x_l}) & \epsilon : v_j :: w_k : x_l \\ C_{w_kx_{l-1}}^{u_iv_{j-1}} + \Delta(S_{\epsilon\to v_j}, S_{\epsilon\to x_l}) & \epsilon : v_j :: \epsilon : x_l \\ C_{w_{k-1}x_l}^{u_iv_{j-1}} + \Delta(S_{\epsilon\to v_j}, S_{w_k\to\epsilon}) & \epsilon : v_j :: w_k : \epsilon \\ C_{w_kx_l}^{u_iv_{j-1}} + \Delta(S_{\epsilon\to v_j}, S_{\epsilon\to\epsilon}) & \epsilon : v_j :: \epsilon : \epsilon \\ C_{w_{k-1}x_{l-1}}^{u_{i-1}v_j} + \Delta(S_{u_i\to\epsilon}, S_{w_k\to x_l}) & u_i : \epsilon :: w_k : x_l \\ C_{w_kx_{l-1}}^{u_{i-1}v_j} + \Delta(S_{u_i\to\epsilon}, S_{\epsilon\to x_l}) & u_i : \epsilon :: \epsilon : x_l \\ C_{w_{k-1}x_l}^{u_{i-1}v_j} + \Delta(S_{u_i\to\epsilon}, S_{w_k\to\epsilon}) & u_i : \epsilon :: w_k : \epsilon \\ C_{w_kx_l}^{u_{i-1}v_j} + \Delta(S_{u_i\to\epsilon}, S_{\epsilon\to\epsilon}) & u_i : \epsilon :: \epsilon : \epsilon \\ C_{w_{k-1}x_{l-1}}^{u_iv_j} + \Delta(S_{\epsilon\to-}, S_{w_k\to x_l}) & \epsilon : \epsilon :: w_k : x_l \\ C_{w_kx_{l-1}}^{u_iv_j} + \Delta(S_{\epsilon\to\epsilon}, S_{\epsilon\to x_l}) & \epsilon : \epsilon :: \epsilon : x_l \\ C_{w_{k-1}x_l}^{u_iv_j} + \Delta(S_{\epsilon\to\epsilon}, S_{w_k\to\epsilon}) & \epsilon : \epsilon :: w_k : \epsilon \end{cases}$$

Terminaison

Quand $i = |u|$ et $j = |v|$ et $k = |w|$ et $l = |x|$.

Résultat

$C_{w_{|w|}x_{|x|}}^{u_{|u|}v_{|v|}}$ est la dissemblance analogique $DA(u,v,w,x)$ selon la distance δ.

Complexité

Cet algorithme est en $\mathcal{O}\big(|u|.|v|.|w|.|x|\big)$, donc en puissance quatrième de la longueur de la plus longue des des phrases. Il est donc *a priori* beaucoup plus lent que l'algorithme approché.

3 Premières expériences en apprentissage par analogie dans les séquences.

Le matériel expérimental que nous utilisons est un ensemble de données artificielles de petite taille. Nous construisons un ensemble de séquences dans lequel nous connaissons les quadruplets en analogie. Nous prenons une de ces séquences et nous la bruitons. L'expérience réalisée consiste à constater si, oui ou non, le meilleur triplet, celui qui a la dissemblance la moindre avec la séquence bruitée, est encore le même. Il ne s'agit pas encore d'un véritable test de la capacité de l'apprentissage par analogie dans les séquences. Le but est pour le moment essentiellement de chercher à savoir comment la dissemblance analogique entre séquences, telle que nous l'avons définie par la distance d'édition, se comporte en présence de bruit.

3.1 Constitution de la base de données.

3.1.1 Les séquences.

Une base de données est constituée de cent séquences de même longueur $2n$, qui sont quatre par quatre en relation d'analogie exacte (et de dissemblance analogique nulle). Pour assurer cette propriété, un ensemble de quatre séquences u, v, w et x de longueur $2n$ est composé en tirant au hasard quatre séquences X, Y, Z et T de longueur n, qui sont ensuite concaténées comme suit : $u = XZ$, $v = XT$, $w = YZ$, $x = YT$. Bien que les composants puissent être très différents, il y a une dissemblance analogique nulle entre les phrases composées[14]. En répétant 25 fois cette opération, on dispose ainsi de 25 quadruplets en analogie exacte. On vérifie aussi qu'il n'existe pas d'autres quadruplets en analogie dans les 100 phrases. Ainsi, chaque séquence de la base de données n'est en analogie qu'avec le triplet composé avec elle.

3.1.2 L'alphabet et la distance δ.

Nous avons choisi un alphabet Σ de $2p$ lettres, défini à partir de $p+1$ traits (voir (MICLET & DELHAY, 2004)). Par exemple pour $p = 3$, l'alphabet est $\Sigma = \{a, b, c, A, B, C\}$ et il est défini à partir des 4 traits binaires suivants (en colonnes):

a	1	0	0	0
b	0	1	0	0
c	0	0	1	0
A	1	0	0	1
B	0	1	0	1
C	0	0	1	1

14. Ce n'est pas le cas en général de quatre phrases en analogie. Cette propriété provient ici des contraintes qui ont servi à définir la distance Δ au paragraphe 2.2.3. Les longueurs de X, Y, Z et T pourraient d'ailleurs ne pas être les mêmes. Sur la construction de quadruplets en analogie par de telles alternances, voir (YVON *et al.*, 2004).

Les trois premiers traits définissent le caractère et le dernier définit sa « casse » (majuscule ou minuscule). Nous en avons déduit trois distances, dont voici deux :

δ_1	a	b	c	A	B	C	ϵ
a	0	2	2	1	3	3	4
b	2	0	2	3	1	3	4
c	2	2	0	3	3	1	4
A	1	3	3	0	2	2	4
B	3	1	3	2	0	2	4
C	3	3	1	2	2	0	4
ϵ	4	4	4	4	4	4	

δ_3	a	b	c	A	B	C	ϵ
a	0	1.5	1.5	1.2	1.7	1.7	2
b	1.5	0	1.5	1.7	1.2	1.7	2
c	1.5	1.5	0	1.7	1.7	1.2	2
A	1.2	1.7	1.7	0	1.5	1.5	2
B	1.7	1.2	1.7	1.5	0	1.5	2
C	1.7	1.7	1.2	1.5	1.5	0	2
ϵ	2	2	2	2	2	2	

La première est la distance de Hamming entre les lettres vus commes des vecteurs binaires de traits, dont nous avons montré dans (MICLET & DELHAY, 2004) qu'elle est cohérente avec la relation d'analogie, c'est à dire que pour tout quadruplet en analogie $a : b ::: d$ δ_1 vérifie : $\delta_1(a,b) = \delta_1(c,d)$ et $\delta_1(a,c) = \delta_1(b,d)$.

Le coût d'insertion et de suppression n'est pas défini par le système de traits. Lui donner une valeur forte comme dans δ_1 permet *a priori* d'éviter des dissemblances faibles dans des schémas du type *au* : *av* :: *wa* : *xa* . Nous constaterons expérimentalement cette propriété en comparant δ_1 avec une distance appelée δ_2, qui ne diffère de δ_1 que par la valeur des insertions et des suppressions, fixée à 2 au lieu de 4.

Quant à δ_3, elle n'est pas très différente de δ_1, mais elle est construite pour fournir des ensembles $\aleph(u,v)$ et $\aleph(w,x)$ de taille aussi petite que possible. Ceci n'est pas indifférent compte tenu du protocole expliqué au paragraphe suivant.

3.2 Protocole expérimental.

3.2.1 Déroulement d'une expérience.

Une expérience se déroule ainsi : une séquence est enlevée de la base de données et bruitée avec un certain taux (le calcul du bruitage sera expliqué au paragraphe suivant). Ensuite, on cherche dans les 99 phrases restantes le triplet qui a la plus petite dissemblance analogique (approchée ou non) avec la phrase bruitée. Si c'est le triplet original, le score de cette expérience augmente de 1, sinon il reste inchangé. On recommence pour chaque séquence. En fin d'expérience, on dispose d'un score entre 0 et 100 qui indique la robustesse au bruit de la dissemblance analogique. On sait, par construction, que pour un bruitage de taux nul, le score de l'expérience est de 100.

3.2.2 Le bruitage.

Les séquences sont bruitées avec un certain taux de bruit τ, entre 0 et 100. La manière de bruiter est définie par un *transducteur* représenté en figure 1. Le bruitage peut être *uniforme* ou varier en sens inverse de la distance δ d'une insertion, d'une suppression et d'une substitution. Chaque lettre est dans ce cas transformée selon la probabilité de transition dans le transducteur. Nous avons choisi deux façons de définir ces probabilités de transitions à partir d'une distance, donc au total trois types de bruit.

Supposons, pour l'exemple, travailler avec l'alphabet $\Sigma = \{a, b, A, B\}$. La première méthode de bruitage (dite en « $1/\delta$ ») fait dépendre la probabilité de transition d'une

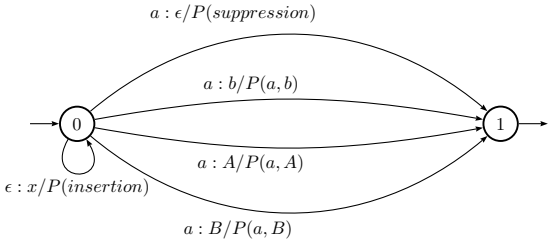


FIG. 1: *Automate de bruitage des séquences*

lettre a vers une autre lettre x de l'inverse des distances de lettre à lettre :

$$P(a,x) = \tau . \frac{\frac{1}{\delta(a,x)}}{\Sigma_{y \in \Sigma - \{a\}} \frac{1}{\delta(a,y)} + \frac{1}{\delta_{insertion}} + \frac{1}{\delta_{suppression}}}$$

La seconde méthode (dite en « $1 + max - \delta$ ») définit une probabilité de transition comme dépendant de la différence des distances lettre à lettre avec le maximum de ces distances. Par exemple, en faisant varier le bruitage, la méthode $1/\delta$ donne, pour δ_3 :

Séquence originale	aAbdDD	adEdebbEaa	AdAcbBec
$\tau = 0.1$	aAEdDD	adEdebbEaa	AdAcBec
$\tau = 0.2$	aAbdDD	adEdebbEaA	AdAcbbec
$\tau = 0.3$	AAbdDD	aadEdBbbCeaa	adacDBcc

3.2.3 Dissemblance analogique approchée et dissemblance analogique.

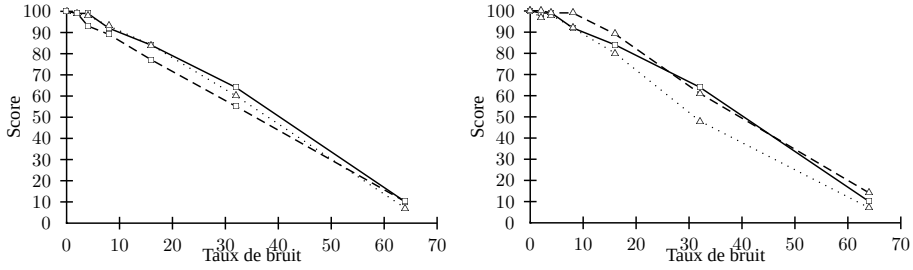
L'algorithme de calcul de la dissemblance analogique approchée compare en principe toutes les paires d'alignements optimaux entre u et v d'une part, w et x d'autre part. En pratique, nous ne comparons que la première paire, dans l'ordre de fonctionnement de l'algorithme. Il est possible que cette nouvelle approximation affaiblisse un peu la méthode, car une paire d'alignements optimaux pourrait se révéler meilleure dans le calcul ultérieur. C'est ce qui semble résulter de l'expérience relatée sur la figure 2(b). La distance δ_3 a été en effet construite, contrairement à δ_1, pour fournir des ensembles $\aleph(u, v)$ et $\aleph(w, x)$ de taille aussi petite que possible.

L'algorithme de calcul de la dissemblance analogique est beaucoup plus long à expérimenter. Nous comptons l'expérience comme positive si *un* des triplets à distance analogique minimale est le bon. Cette méthode est optimiste (et elle suppose que l'on connaît le résultat !). En pratique, nous avons constaté, en particulier pour la distance δ_3, que le nombre des triplets à distance analogique minimale est la plupart du temps égal à un (en moyenne, environ 1.1).

3.3 Résultats et premières conclusions.

La figure 2(a) compare, pour la distance de traits donnée ci-dessus, pour un alphabet à 8 lettres et des séquences de longueur 10, les trois modes de bruitage pour la distance

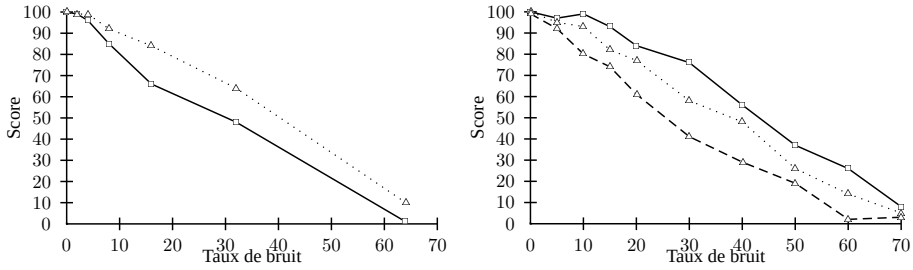
δ_1 et un bruitage uniforme. La figure 2(b) compare sur les mêmes données les distances δ_1, δ_2 et δ_3. Les figures 3(a) et 3(b) comparent les deux dissemblances analogiques et analysent l'influence de la longueur des séquences.



(a) Influence du type de bruitage.

(b) Influence du choix de la distance.

FIG. 2: ***(a)*** *: Comparaison de de la méthode $1/\delta$ (en trait plein), de la méthode $1 + max - \delta$ (en pointillés) et d'une génération uniforme de bruit (en trait interrompu).* ***(b)*** *: Comparaison de δ_1 (en trait plein), de δ_2 (en pointillés) et de δ_3 (en trait interrompu). Le générateur de bruit est du type $1/\delta$ dans les trois cas. L'algorithme utilisé calcule la dissemblance analogique approchée.*



(a) Comparaison entre la dissemblance analogique approchée et la dissemblance analogique.

(b) Influence de la longueur $2n$ des séquences de la base de données. En trait plein, $n = 10$; en pointillé, $n = 8$, en trait interrompu, $n = 6$.

FIG. 3: ***(a)*** *: La distance analogique approchée est en pointillés et la dissemblance analogique en trait plein. La distance est δ_1. Le générateur de bruit est du type $1/\delta$.* ***(b)*** *: Dans toutes les autres figures, la base de données est composée de phrases de longueur 8 sur un alphabet de taille 10. Ici, la taille des séquences varie, sur ce même alphabet. La base de données comporte 100 séquences, la distance est δ_3, la méthode est la dissemblance analogique approchée et le bruit est en $1/\delta$.*

En première analyse, on peut tirer les conclusions suivantes :

- La dissemblance analogique offre une certaine résistance au bruit : le bon triplet analogique est retrouvé dans environ les deux tiers des cas, pour cent phrases de longueur 8, avec un taux de bruitage de 32%. Compte tenu du nombre de paramètres de cette expérience, il est difficile d'en tirer plus d'enseignements.

- Le type de bruit n'a pas d'importance.

- Un coût de suppression et d'insertion élevé est plutôt souhaitable. L'analyse fine de ce phénomène montre en effet qu'il permet d'éviter des dissemblances faibles dans des schémas du type $au : av :: wa : xa$.

- Sur ce protocole expérimental, la dissemblance analogique approchée est significativement meilleure que la dissemblance analogique. Il est difficile de tirer des conclusions de ce constat avant de confronter les deux algorithmes à un véritable test d'apprentissage supervisé.

4 Conclusion et perspectives

Dans cet article nous avons proposé une approche pour l'apprentissage par analogie basée sur la distance d'édition et la définition d'une *dissemblance analogique*. Nous avons proposé deux algorithmes pour calculer cette dissemblance. Le premier est sous-optimal et de complexité quadratique, le second est optimal mais a une complexité algorithmique plus élevée (d'ordre 4).

Nous avons ensuite expérimenté ces algorithmes sur un corpus jouet, construit avec des séquences de même longueur (8 symboles pris dans un alphabet de taille 10). En étudiant différentes façon de bruiter les échantillons, nous avons observé que la dissemblance analogique, approchée ou non, est résistante au bruit.

L'algorithme calculant la dissemblance analogique est d'une complexité relativement élevée (en $\mathcal{O}(n^4)$ si n est la taille de la plus grande séquence). Mais surtout, la recherche de la dissemblance analogique, telle que nous l'avons implantée, est en $\mathcal{O}(m^3)$ si m est la taille du corpus, ce qui est largement rédhibitoire pour sur des corpus de grande taille.

Un de nos objectifs prioritaire est donc de réduire cette complexité, afin de pouvoir appliquer nos algorithmes sur des corpus réalistes et des alphabet plus grands. La construction proposée, qui garantit que les propriétés de distance sont préservées, permet d'envisager des algorithmes rapides du type de ceux utilisés dans la méthode des plus proches voisins. Nous pourrons alors comparer nos résultats de reconnaissance en apprentissage par analogie à ceux obtenus en apprentissage par plus proches voisins, par exemple. Les applications visées sont d'abord les technologies vocales, pour des séquences de graphèmes et de phonèmes (reconnaissance et synthèse de la parole), ensuite les bio-séquences. Nous nous intéressons en particulier aux systèmes de séquences dans lesquels une distance naturelle peut être introduite, soit par connaissances *a priori*, soit par apprentissage.

Références

CORNUÉJOLS A. & MICLET L. (2002). *Apprentissage artificiel : concepts et algorithmes*. Eyrolles.

CROCHEMORE M. & *et al.* (2001). *Algorithmique du texte.* Vuibert.

DELHAY A. & MICLET L. (2004). Solving analogical equations for learning by analogy with sequences. In *Proceedings of CAP-2004, PUG*, p. 347–362, Montpellier.

HOFSTADTER D. & THE FLUID ANALOGIES RESEARCH GROUP (1994). *Fluid Concepts and Creative Analogies*. New York: Basic Books.

LEPAGE Y. (2003). *De l'analogie rendant compte de la commutation en linguistique*. HDR, Université Joseph Fourier Grenoble I.

LEPAGE Y. & ANDO S.-I. (1996). Saussurian analogy: a theoretical account and its application. In *Proceedings of COLING-96*, p. 717–722, København.

MICLET L. & DELHAY A. (2003). *Analogy on Sequences: a Definition and an Algorithm*. Rapport interne 4969, INRIA.

MICLET L. & DELHAY A. (2004). *Relation d'analogie et distance sur un alphabet défini par des traits*. Rapport interne 5244, INRIA. in French.

D. SANKOFF & J. KRUSKAL, Eds. (1983). *Time Warps, String Edits and Macromolecules: the Theory and Practice of Sequence Comparison*. Addidon-Wesley.

J. W. SHAVLIK & T. G. DIETTERICH, Eds. (1990). *Readings in Machine Learning*. Morgan Kaufmann.

WAGNER R. & FISHER M. (1974). The string-to-string correction problem. *JACM*.

R. A. WILSON & F. C. KEIL, Eds. (1999). *Encyclopedia of the Cognitive Sciences*. MIT Press.

YVON F., STROPPA N., DELHAY A. & MICLET L. (2004). *Solving analogical equations on words*. Rapport interne ENST2004D005, École Nationale Supérieure des Télécommunications.

Phase transitions in grammatical inference

Nicolas Pernot[1], Antoine Cornuéjols[1,2], Michèle Sebag[1]

[1] Équipe TAO, Laboratoire de Recherche en Informatique,
Bâtiment 490, Université Paris-Sud, 91405 - Orsay Cedex
antoine@lri.fr et http://www.lri.fr/∼antoine

[2] Institut d'Informatique d'Entreprise,
18, allée Jean Rostand, 91025 - Evry Cedex

Résumé : L'analyse théorique de l'apprentissage inductif a montré que la validité du principe de minimisation du risque empirique est régie par des propriétés statistiques portant en particulier sur la "capacité" de l'espace des hypothèses. De même, la découverte, il y a quelques années, d'un phénomène de transition de phase en programmation logique inductive prouve que d'autres caractéristiques fondamentales du problème d'apprentissage peuvent également contrôler la possibilité de l'apprentissage sous des conditions très générales.

Ce papier porte sur l'examen de l'inférence grammaticale. Nous montrons que si il n'existe pas de phénomène de transition de phase lorsque l'on considère l'ensemble de l'espace des hypothèses, il existe un "trou" bien plus sévère dans l'espace de recherche effectif des algorithmes d'induction par fusion d'états dans le cas des automates finis déterministes (DFA). L'examen des heuristiques de recherche des algorithmes RPNI et RED-BLUE montre que si ces algorithmes parviennent à surmonter ce problème en partie, ils sont en revanche portés à surgénéraliser. Le papier suggère finalement quelques pistes pour l'utilisation de nouveaux opérateurs de généralisation afin de pallier le phénomène de transition de phase.

Mots-clés : Inférence grammaticale, transition de phase, heuristiques de recherche.

1 Introduction

It is now well-known that the feasibility of inductive learning is ruled by statistical properties linking the empirical risk minimization principle and the "capacity" of the hypothesis space (Vapnik, 1995). In part thanks to this analysis, new induction criterions have been proposed (e.g. structural risk minimization). While this powerful framework leads to a much deeper understanding of machine learning and to many theoretical and applicative breakthroughs, it basically involves only static information on the learning search space, e.g. the so-called VC-dimension. The dynamics of the learning search is not considered.

In parallel and independently, a new paradigm has been studied in the Constraint Satisfaction community since the early 90s, motivated by computational complexity

concerns : where are the really hard problems ? (Cheeseman *et al.*, 1991) Indeed, the worst case complexity analysis poorly accounts for the fact that, despite an exponential worst-case complexity, empirically, the complexity is low for most CSP instances. These remarks led to developing the so-called *phase transition framework* (Hogg *et al.*, 1996), which considers the satisfiability and the resolution complexity of CSP instances as random variables depending on order parameters of the problem instance (e.g. constraint density and tightness). This framework unveiled a much interesting structure of the CSP landscape. Specifically, the landscape is divided into three regions : the YES region, corresponding to underconstrained problems, where the satisfiability probability is close to 1 and the average complexity is low ; the NO region, corresponding to overconstrained problems, where the satisfiability probability is close to 0 and the average complexity is low too ; last, a narrow region separating the YES and NO regions, referred to as *phase transition region*, where the satisfiability probability abruptly drops from 1 to 0 and which concentrates on average the computationally heaviest CSP instances.

The phase transition paradigm has been transported to relational machine learning and inductive logic programming (ILP) by (Giordana & Saitta, 2000), motivated by the fact that the covering test most used in ILP is equivalent to a CSP. As anticipated, a phase transition phenomenon appears in the framework of ILP : a wide YES (respectively NO) region includes all hypotheses which cover (resp. reject) all examples, and hypotheses separating the examples lie in the narrow PT, where the average computational complexity of the covering test reaches its maximum.

Besides computational complexity, the PT phenomenon has far-reaching effects on the success of relational learning (Botta *et al.*, 2003). For instance, a wide *Failure Region* is observed : for all target concepts/training sets in this region, no learning algorithms among the prominent ILP ones could find hypotheses better than random guessing (Botta *et al.*, 2003).

These negative results lead to a better understanding of the intrinsic limits of the existing ILP algorithms and search biases. Formally, consider a greedy specialization search strategy : starting its exploration in the YES region, the system is almost bound to make random specialization choices, for all hypotheses in this region cover every example on average. The YES region constitutes a rugged plateau from a search perspective, and there is little chance that the algorithm ends in the right part of the PT region, where good hypotheses lie. A similar reasoning goes for algorithms that follow a greedy generalization strategy.

The phase transition paradigm thus provides another perspective on the pitfalls facing machine learning, focusing on the combinatoric search aspects while statistical learning focuses on the statistical aspects.

The main question studied in this paper is whether the PT phenomenon is limited to relational learning, or threatens the feasibility and tractability of other learning settings as well.

A learning setting with intermediate complexity between full relational learning and propositional learning is thus considered, that of grammatical inference (GI) (Angluin, 1988; Pitt, 1989; Sakakibara, 1997). Only the case of Deterministic and Non-Deterministic Finite-State Automata (FSA, section 2), will be considered through the paper. Specifically, the phase transition phenomenon will be investigated with respect to three

distributions on the FSA space, incorporating gradually increasing knowledge on the syntactical and search biases of GI algorithms.

The first one, the uniform distribution, incorporates no information and considers the whole space of FSA. Using a set of order parameters, the average coverage of the automata is studied analytically and empirically.

The second one reflects the bias introduced by the generalization relations defined on the FSA space and exploited by GI algorithms. The vast majority of these algorithms first construct a least general generalization of the positive examples, or Prefix Tree Acceptor (PTA), and restrict the search to the generalizations of the PTA, or *generalization cone*[1]. Indeed, it appears that the generalization cone presents a quite specific structure compared to the whole FSA space.

Lastly, the third distribution takes into account the search biases of GI algorithms, guiding the search trajectory in the generalization cone. Due to space limitations, the study is restricted to two prominent GI algorithms, namely RPNI (Oncina & Garcia, 1992) and RED-BLUE (Lang *et al.*, 1998).

This paper is organized as follows. Section 2 briefly introduces the domain of Grammatical Inference, the principles of the inference algorithms and defines the order parameters used in the rest of the paper. Section 3 investigates the existence and potential implications of phase transition phenomenons in the whole FSA space (section 3.2) and in the generalization cone (section 3.3). Section 4 focuses on the actual landscape explored by GI algorithms, specifically considering the search trajectories of RPNI and RED-BLUE. Section 5 discusses the scope of the presented study and lays out some perspectives for future research.

2 Grammatical inference

After introducing general notations and definitions, this section briefly discusses the state of the art and introduces the order parameters used in the rest of the paper.

2.1 Notations and definitions

Grammatical inference is concerned with inferring grammars from positive (and possibly negative) examples. Only regular grammars are considered in this paper. They form the bottom class of the hierarchy of formal grammars as defined by Chomsky, yet are sufficiently rich to express many interesting sequential structures. Their identification from positive examples only is known to be unrealizable, while it is feasible with a complete set of examples (Gold, 1967).

It is known that any regular language can be produced by a finite-state automaton (FSA), and that any FSA generates a regular language. In the remaining of the paper, we will mostly use the terminology of finite-state automata. A FSA is a 5-tuple $A = \langle \Sigma, \mathcal{Q}, \mathcal{Q}_0, F, \delta \rangle$ where Σ is a finite alphabet, $\mathcal{Q}$ is a finite set of states, $\mathcal{Q}_0 \subseteq \mathcal{Q}$ is the

[1]More precisely, the Prefix Tree Acceptor is obtained by merging the states that share the same prefix in the Maximal Canonical Automaton (MCA), which represents the whole positive learning set as an automaton. A PTA is therefore a DFA with a tree-like structure.

set of initial states, $F \subseteq \mathcal{Q}$ is the set of final states, δ is the transition function defined from $\mathcal{Q} \times \Sigma$ to $2^{\mathcal{Q}}$.

A positive example of a FSA is a string on Σ, produced by following any path in the graph linking one initial state q_0 to any accepting state.

A finite state-automaton (FSA) is deterministic (DFA) if $\mathcal{Q}_0$ contains exactly one element q_0 and if $\forall q \in \mathcal{Q}, \forall x \in \Sigma, Card(\delta(q,x)) \leq 1$. Otherwise it is non-deterministic (NFA). Every NFA can be translated into an equivalent DFA, but at the price of being possibly exponentially more complex in terms of number of states. Given any FSA A', there exists a minimum state DFA (also called *canonical DFA*) A such that $L(A) = L(A')$ (where $L(A)$ denotes the set of strings accepted by A). Without loss of generality, it can be assumed that the target automaton being learned is a canonical DFA.

A set S^+ is said to be *structurally complete* with respect to a DFA A if S^+ covers each transition of A and uses every element of the set of final states of A as an accepting state. Clearly, $L(PTA(S^+)) = S^+$.

Given a FSA A and a partition π on the set of states $\mathcal{Q}$ of A, the *quotient automaton* is obtained by merging the states of A that belong to the same block of partition π (see (Dupont *et al.*, 1994) for more details). Note that a quotient automaton of a DFA might be a NFA and vice versa. The set of all quotient automata obtained by systematically merging the states of a DFA A represents a lattice of FSA. This lattice is ordered by the *grammar cover* relation $\preceq$. The transitive closure of $\preceq$ is denoted by $\ll$. We say that $A_{\pi_i} \ll A_{\pi_j}$ *iff* $L(A_{\pi_i}) \subseteq L(A_{\pi_j})$. Given a canonical DFA A and a set S^+ that is structurally complete with respect to A, the lattice derived from $PTA(S^+)$ is guaranteed to contain A.

From these unverifiable assumptions, follows the paradigmatic approach of most grammatical inference algorithms (see, e.g., (Angluin, 1988; Coste, 1999; Dupont *et al.*, 1994; Dupont & Miclet, 1998; Pitt, 1989; Sakakibara, 1997)), which equates generalization with state merging in the candidate automaton.

2.2 Learning biases in grammatical inference

The core task of GI algorithms is thus to select iteratively a pair of states to be merged. The differences among algorithms is related to the choice of : (i) the search criterion (which merge is the best one) ; (ii) the search strategy (how is the search space explored) ; and (iii) the stopping criterion.

We shall consider here the setting of learning FSAs from positive and negative examples, and describe the algorithms studied in section 3.In this setting, the stopping criterion is determined from the negative examples : generalization proceeds as long as the candidate solutions remain correct, not covering any negative example[2].

The RPNI algorithm (Oncina & Garcia, 1992) uses a depth first search strategy, and retains the pair of states which is closest to the start state (obtained through a natural numbering of the states), such that their generalization (FSA obtained by merging the two states and subsequently applying the determinisation operator) does not cover any

[2]In this paper, we follow the tradition of the literature in concept learning, where a concept *covers* an example if the example belongs to the subset of the example space denoted by the concept. In grammatical inference, *covering* can be equated with the *recognition* of a string by the automaton.

negative example. RPNI performs a greedy search, albeit with backtracking possibility. Overall, it is thus computationally efficient.

The RED-BLUE algorithm (also known as BLUE-FRINGE) (Lang *et al.*, 1998) uses a beam search from a candidate list, selecting the pair of states after the *Evidence-Driven* State Merging (EDSM) criterion, i.e. such that their generalization involves a minimal number of final states. RED-BLUE thus also performs a limited backtracking search, based on a more complex criterion and a wider search width compared to RPNI.

2.3 Order Parameters

Following the methodology introduced in (Giordana & Saitta, 2000), the PT phenomenon is investigated along dimensions called *order parameters*. They were chosen here in accordance with the parameters used in the *Abbaddingo* challenge (Lang *et al.*, 1998) :

- The size Σ of the alphabet considered.
- The number Q of states in the DFA.
- The number B of output edges on each state.
- The number L of letters on each edge.
- The fraction a of accepting states, taken in [0,1].
- The length ℓ of the test example.
- The maximal length ℓ_{learn} of the learning examples in S^+ (as explained below).

The first part of our investigation (section 3) bears on properties of the hypothesis space independently of any target automaton. For this part, we have used a *random sampling* strategy (all ℓ letters in the string are independently and uniformly drawn in Σ). In the second part (section 4) , where we examine the capacity of the studied learning algorithms to approximate a target automaton, we used *positive sampling*, where each learning string is produced by following a path in the graph, determined by selecting randomly an output edge in each step[3].

3 Phase Transitions : the FSA space and the generalization cone

This section investigates the percentage of coverage of deterministic and non-deterministic Finite-State Automata, either uniformly selected (section 3.2), or selected in the subspace actually investigated by grammatical inference algorithms, that is, the generalization cone (section 3.3). We first detail the experimental protocol used.

3.1 Experimental setting

For each setting of the order parameters, 100 independent problem instances are constructed. For each considered FSA (the sampling mechanisms are detailed below), the coverage rate is measured as the percentage of covered examples among 1,000 examples (strings of length ℓ, sampled along uniform distribution).

[3]The string is cut at the last accepting state met before arriving at length ℓ, if any ; otherwise it is rejected.

3.2 Phase Transition in the whole FSA space

The sampling mechanism on the whole deterministic FSA space (DFA) is defined as follows. Given the order parameter values (Q, B, L, a, Σ) :

- for every state q, (i) B output edges (q, q') are created, where q' is uniformly selected with no replacement among the Q states ; (ii) $L \times B$ distinct letters are uniformly selected in Σ ; and (iii) these letters are evenly distributed among the B edges above.
- every state q is turned into an accepting state with probability a.

The sampling mechanism for NFA differs from the above in a single respect : two edges with same origin state are not required to carry distinct letters.

Fig. 1 shows the average coverage in the (a, B) plane, for $|\Sigma| = 2$, $L = 1$ and $\ell = 10$, where the accepting rate a varies in $[0, 1]$ and the branching factor B varies in $\{1,2\}$. Each point reports the coverage of a sample string s by a FSA A, averaged over 100 FSA drawn with accepting rate a and branching factor B, times $1,000$ strings s of length ℓ.

These empirical results are analytically explained from the simple equations below, giving the probability that a string of length ℓ be accepted by a FSA defined on an alphabet of size $|\Sigma|$ with Q states, with a branching factor B and L letters on each edge, in the DFA and NFA cases.

$$P(\text{accept}) = \begin{cases} a \cdot (\frac{B \cdot L}{|\Sigma|})^{\ell} & \text{for a DFA} \\ a \cdot [1 - (1 - \frac{L}{|\Sigma|})^{B}]^{\ell} & \text{for a NFA} \end{cases}$$

The coverage of the FSA decreases as a and B decrease. The slope is more abrupt in the DFA case than in the NFA case ; still, there is clearly no phase transition here.

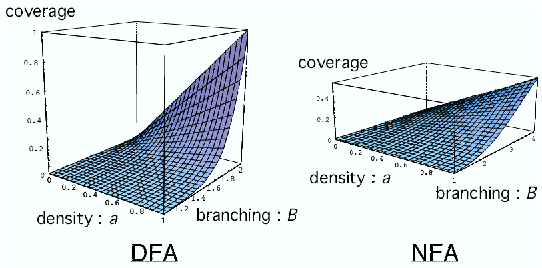


FIG. 1 – Coverage landscapes for Deterministic and Non-Deterministic FSA, for $|\Sigma|$=2, L=1 and ℓ=10. The density of accepting states a and the branching factor B respectively vary in $[0, 1]$ and $\{1, 2\}$.

3.3 PT in the Generalization Cone

The behaviour of the coverage displayed in Fig. 1 might lead to the impression that grammatical inference occurs in a well-behaved search space. However, grammatical inference algorithms do not explore the whole FSA space. Rather, as stated in section

2.1, the search is restricted to the generalization cone, the set of generalizations of the PTA formed from the set S^+ of the positive examples. The next aim is thus to considers the search space actually explored by GI algorithms.

A new sampling mechanism is defined to explore the DFA generalization cone :

1. N ($= 100$ in the experiments) examples of length ℓ are uniformly and independently sampled, and their PTA is constructed ;
2. K ($= 1000$ in the experiments) generalization paths, leading from the PTA to the most general FSA or Universal Acceptor (UA), are constructed ;
3. In each generalization path ($A_0 = PTA, A_1, \ldots, A_t = UA$), the i-th FSA A_i is constructed from A_{i-1} by merging two uniformly selected states in A_{i-1}, and subsequently applying the determinisation operator.
4. The sample of the generalization cone is made of all FSAs in all generalization paths.

The sampling mechanism on the non-deterministic generalisation cone differs from the above in a single respect : the determinisation operator is never applied

Fig. 2 shows the behaviour of the coverage in the DFA generalisation cone, gathered from 50 samples with $|\Sigma| = 4$ and $\ell = 8$. Each DFA A is depicted as a point with coordinates (Q, c), where Q is the number of states of A and c is its coverage (measured as in section 3.1).

Fig. 3 similarly shows the behaviour of the coverage in the NFA generalisation cone, gathered from 50 samples with $|\Sigma| = 4$ and $\ell = 16$.

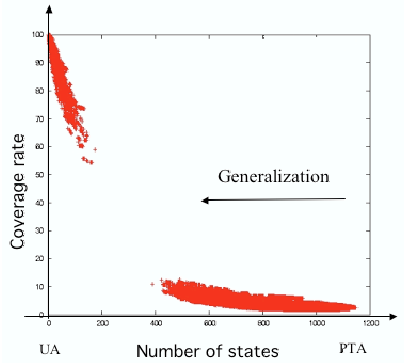


FIG. 2 – Coverage in the DFA generalization cone, here with $|\Sigma| = 4$, $\ell_{learn} \in [1, 8]$. At the far right stand the 50 PTA sampled, with circa 1150 states each. The generalization cone of each PTA includes 1,000 generalization paths, starting from the PTA and moving to the Universal Acceptor with one single state and coverage 1. Each point reports the coverage of a DFA on a generalization path, measured over 1,000 strings. This graph shows the existence of a large gap regarding both the number of states and the coverage of the DFAs that can be reached by generalization.

Fig 2, typical of all experimental results in the range of observation ($|\Sigma| = 2, 4, 8, 16$, and $\ell = 2, 4, 6, 8, 16, 17$) shows a clear-cut phase transition. Specifically, the coverage

abruptly jumps from circa 13% to 54% ; and this jump coincides with a gap in the number of states of the DFAs in the generalization cone : no DFA with state number in $[180, 420]$ was found.

Interestingly, the picture is much smoother in the non-deterministic case (fig. 3) ; although the coverage rapidly increases when the number of states decreases from 300 to 200, no gap can be seen, either in the number of states or in the coverage rate itself. [4]

In the following, we focus on the induction of DFAs.

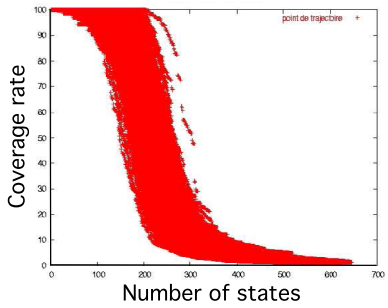


FIG. 3 – Typical coverage behaviour in the NFA generalization cone, with same order parameters as in Fig. 3.3 except $\ell = 16$.

4 Phase transition and search trajectories

The study of the coverage rate in the generalization cone shows that the density of hypotheses of coverage in between a large interval (typically between less than 20% to approximately 60%) falls abruptly. This means that a random exploration of the generalization cone would have great difficulties in finding an hypothesis in this region and would therefore likely return hypotheses of poor performance (overspecialized or overgeneralized) if the target concept had a coverage rate in this “no man’s land” interval.

It is consequently of central importance to examine the search heuristics that are used in the classical grammatical inference systems. First, are they able to thwart the a priori very low density of hypotheses in the gap ? Second, are they able to guide the search toward hypotheses that have a coverage rate correlated with the one of the target concept, specially if this coverage falls in the gap ?

We focused our study on two standard algorithms in grammatical inference, namely the RPNI and the RED-BLUE algorithms (Oncina & Garcia, 1992; Lang *et al.*, 1998).

[4] The difference with the DFA case is due to the determinisation process that forces further states merging when needed. We devised a chain-reaction like analytical model that accounts rather well with the observed behaviour, predicting the start of the gap (a chain reaction) with a 15% precision.

4.1 Experimental setting

By contrast to previous experiments, the learning behaviour is now studied with respect to a target concept. In particular, GI algorithms (and specifically RPNI and RED-BLUE) use negative examples in order to stop their generalization process.

In our first experiments, we tested whether heuristically guided inference algorithms can find good approximations of the target automata whatever its coverage rate. In particular, we considered target automata either of coverage rate of approximately 50% (in the middle of the "gap"[5]), or of coverage of approximately 5%.

For each target coverage rate, we followed the protocol described in (Lang *et al.*, 1998) in order to retain a certain number of target automata with a mean size of Q states ($Q = 50$, in our experiments). For each target automaton then, we generated N (=20) training sets of size $|S|$ (= 100) labeled according to the target automaton, with an equal number of positive and negative instances ($|S^+| = |S^-| = 50$) of length $\ell_{learn} \in [1, 14]$. The coverage rate was computed as before from the mean coverage on a randomly drawn test set of size 1000 with no intersection with the training set.

In a second set of experiments, we analyzed the learning trajectories with respect to test errors, both false positive and false negative, in a ROC like fashion.

In these experiments, we chose the type of target automata by setting the number of states Q and some predetermined structural properties (e.g. density of edges, rate of recursive connexions). We also set the structural completeness of the training set (defined as the percentage of the training set wrt. a structurally complete set of instances). The performance of the learning systems were measured on a test set of 1000 sequences of length $\ell \in [d, 2d]$ with d denoting the depth of the target automaton.

4.2 The heuristically guided search space

Due to space limitation, we report here only the graph obtained for the RPNI algorithm (see figure 4), but it is close to the one obtained with the RED-BLUE algorithm. For the sake of the clarity of the graph, we chose to report only 3 trajectories.

One immediate result is that both the RPNI and the EDSM heuristics manage to densely probe the "gap". This can explain why the gap phenomenon was not discovered before, and why the RED-BLUE algorithm for instance could solve some cases of the *Abbadingo* challenge where the target concepts have a coverage rate of approximately 50%. However, where RPNI tends to stop with automata of coverage less than the target coverage, RED-BLUE tends to overshoot the target coverage by 5% to 10%.

In order to test the capacity of the algorithms to return automata with a coverage rate close to the target coverage, we repeated these experiments with target automata of coverage rate of approximately 3%. Our results (see figure 5) shows that, in this case, RPNI ends with automata of coverage 4 to 6 times greater than the target coverage. The effect is even more pronounced with RED-BLUE which returns automata of average coverage rate around 30% !

[5] And also close to the coverage value of the target automata automatically generated in the *Abbadingo* challenge.

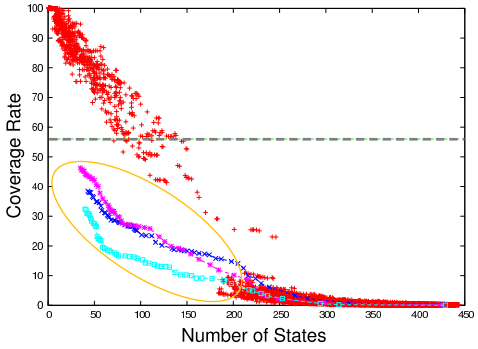


FIG. 4 – Three RPNI learning trajectories for a target concept of 56%. Their extremity is outlined in the oval on the left. The doted horizontal line corresponds to the coverage of the target concept. The cloud of points corresponds to random trajectories.

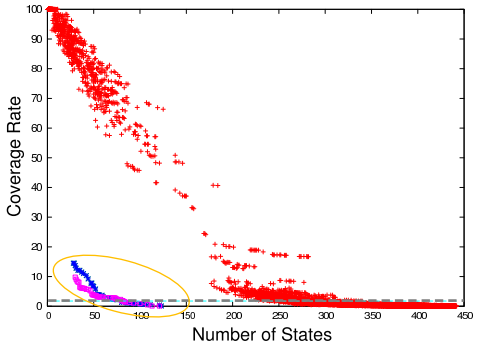


FIG. 5 – Same as in figure 4, except for the coverage of the target concept, here 3%.

4.3 The evolution of the learning performance on the guided trajectories

Since it appears that both RPNI and RED-BLUE tend to overgeneralize, specially when the target concept is of low coverage, it is important to examine more closely their behaviour with respect to positive and to negative test instances.

Table 1, obtained for different sizes of the target automata and for training sets of structural completeness above 40%, confirms that both RPNI and RED-BLUE return overgeneralized hypotheses. On one hand, their average coverage is vastly greater than the coverage of the target automata, on the other hand, they tend to cover only part of the positive test instances, while they cover a large proportion of the negative test instances. This shows that the heuristics used in both RPNI and RED-BLUE may be inadequate for target concepts of low coverage. Either the choice of the learning operators should be adapted, or the stopping criterion could incorporate some knowledge about the target

coverage.

Algo.	Q_c	$ucov_c$	Q_f	$ucov_f$	$pcov_f$	$ncov_f$
RB	15	5.97	10.38	33.81	60.93	34.69
RB	25	4.88	12.77	40.35	62.68	37.87
RB	50	4.2	14.23	45.38	66.14	42.23
RB	100	3.39	13.13	30.35	42.81	28.69
RPNI	15	5.95	5.14	22.9	57.51	26.99
RPNI	25	4.7	7.56	23.07	56.38	25.98
RPNI	50	3.87	14.08	23.45	51.89	24.42
RPNI	100	3.12	26.41	23.151	50.12	24.40

TAB. 1 – Results for target DFA of sizes Q = 15, 25, 50 and 100 states, *recursivity rate* = 50%, *edge density*=50% and training sets of structural completeness above 40%. Q_f, $ucov_f$, $pcov_f$ and $ncov_f$ respectively denote the average size of the learned automata, their average coverage, the average coverage restricted to the positive instances and the average coverage restricted to the negative instances.

5 Conclusion

This research has extended the Phase Transition-based methodology (Botta *et al.*, 2003) to the Grammatical Inference framework. Ample empirical evidence shows that the search landscape presents significant differences depending on the search operators that are considered.

A first result is that random search appears to be more difficult in the DFA generalization cone than in the whole search space : a large gap was found, in terms of hypothesis coverage and size. This explains why control heuristics are needed for the inference of DFA, particularly in the range of problems corresponding to the very influential *Abbadingo* challenge.

A second finding regards the limitations of the search operators in RPNI and RED-BLUE, especially outside the region of the *Abbadingo* target concepts. Experiments with artificial learning problems built from specific target concepts (coverage less than 10%) reveal that RPNI and RED-BLUE alike tend to learn overly general hypotheses (DFAs) ; with respect to both the size (lower number of states) and the coverage of the hypotheses (often larger by an order of magnitude than that of the target concept). What is even more worrying, is that this overgeneralization *does not imply* that the found hypotheses are complete : quite the contrary, the coverage of the positive examples remains below 65%, in all but one setting.

The presented study opens several perspectives for further research. First, it suggests that the learning search could be controlled using a hyper-parameter, the coverage rate of the target concept. This hyper-parameter might either be supplied by the expert, or estimated e.g. by cross-validation. In other words, the stopping criterion of the algorithms might be reconsidered. Second, more conservative generalisation operators will

be investigated ; preliminary experiments done with e.g. reverted generalisation (same operator as in RPNI, applied on the reverted example strings) show that such operators can delay the determinisation cascade, and offer a finer control of the final coverage rate of the hypotheses.

Finally, the main claim of the paper is that the phase transition framework can be used to deliver precise indications as to when heuristics are appropriate – hopefully leading to understand and ultimately alleviate their limitations.

Références

ANGLUIN D. (1988). Queries and concept learning. *Machine Learning journal*, **2**, 319–342.

BOTTA M., GIORDANA A., SAITTA L. & SEBAG M. (2003). Relational learning as search in a critical region. *Journal of Machine Learning Research*, **4**, 431–463.

CHEESEMAN P., KANEFSKY B. & TAYLOR W. M. (1991). Where the Really Hard Problems Are. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence, IJCAI-91, Sidney, Australia*, p. 331–337.

COSTE F. (1999). *State Merging Inference of Finite State Classifiers*. Rapport interne, Irisa.

DUPONT P. & MICLET L. (1998). *Inférence grammaticale régulière : fondements théoriques et principaux algorithmes*. Rapport interne, INRIA N^o 3449.

DUPONT P., MICLET L. & VIDAL E. (1994). What Is the Search Space of the Regular Inference ? In *Proceedings of the Second International Colloquium on Grammatical Inference and Applications, ICGI-94*, p. 25–37.

GIORDANA A. & SAITTA L. (2000). Phase transitions in relational learning. *Machine Learning*, **41**, 217–251.

GOLD E. M. (1967). Language identification in the limit. *Information and Control*, **10**, 447–474.

HOGG T., HUBBERMAN B. & WILLIAMS C. (1996). Phase transitions and the search problem. *Artificial Intelligence*, **81**, 1–15.

LANG K., PEARLMUTTER B. & PRICE R. (1998). Results of the abbadingo one dfa learning competition and a new evidence driven state merging algorithm. In *Fourth International Colloquium on Grammatical Inference (ICGI-98)*, volume LNCS-1433, p. 1–12 : Springer Verlag.

ONCINA J. & GARCIA P. (1992). Inferring regular languages in polynomial update time. *Pattern Recognition and Image Analysis*, p. 49–61.

PITT L. (1989). Inductive inference, dfas, and computational complexity. In *Proceedings of the Workshop on Analogical and Inductive Inference (AII-89)*, volume LNCS-397, p. 18–44 : Springer Verlag.

SAKAKIBARA Y. (1997). Recent advances of grammatical inferences. *Theoretical Computer Science*, **185**, 15–45.

VAPNIK V. (1995). *The Nature of Statistical Learning.* Springer.

Apprentissage par analogie et rapports de proportion: contributions méthodologiques et expérimentales

Nicolas Stroppa, François Yvon

GET/ENST et LTCI, UMR 5141
46, rue Barrault, 75013 Paris
{stroppa,yvon}@enst.fr

L'apprentissage par analogie (Gentner *et al.*, 2001) repose sur un mécanisme inductif en deux étapes : le premier temps consiste à construire un appariement structurel entre une nouvelle instance d'un problème et des instances déjà résolues du même problème ; une fois cet appariement établi, la solution de la nouvelle instance est élaborée à partir des solutions des instances analogues.

Dans le cas particulier du traitement des langues, la taille des bases de données manipulées, contenant des centaines de milliers d'instances, rend prohibitive la recherche d'appariements structurels complexes. En revanche, les entités étudiées dans ce domaine se décrivent à l'aide de formes bien identifiées : séquences (phonétiques, orthographiques), arbres (morphologiques, syntaxiques), structures de traits. L'exploitation d'appariements purement formels entre descriptions permet dans certaines situations de détecter des analogies plus profondes entre entités linguistiques. Ainsi, un appariement de forme entre $chanter$ et $chanteur$ permet de déceler un lien sémantique plus profond entre les entités linguistiques sous-jacentes. De telles ressemblances de forme peuvent évidemment se révéler trompeuses. Toutefois, l'exploitation de la redondance des données linguistiques permet en pratique de s'affranchir de ces limitations : l'appariement $chanter$-$chanteur$ est conforté par la présence de nombreuses paires similaires dans le lexique, à savoir $parler$-$parleur$, $râler$-$râleur$, etc. Pour définir à la fois l'appariement et la similarité de paires, nous utilisons la notion de *rapport de proportion* (ou proportion analogique) entre 4 objets x, y, z, t qui s'écrit $x : y :: z : t$ et qui se lit « x *est à* y *se que* z *est à* t ». Par exemple, $chanter : chanteur :: râler : râleur$.

La tâche d'apprentissage (supervisé) considérée dans cet article consiste, à partir d'une base d'apprentissage décrivant des entités connues, à inférer des propriétés inconnues d'entités nouvelles. Un exemple de tâche consiste à inférer les traits morpho-syntaxiques de la forme graphique $chanteur$ à partir d'instances figurant dans la base[1]
$\{(chanter : VI), (râler : VI), (râleur : NM), (parler : VI), (parleur : NM)\}$
contenant des formes graphiques associées à des traits morpho-syntaxiques.

Pour analyser une instance t dont certaines propriétés sont inconnues, nous procédons en deux temps. Tout d'abord, nous recherchons dans la base d'apprentissage des

[1] Pour les traits morpho-syntaxiques, *V=Verbe, I=Infinitif, N=Nom, M=Masculin.*

triplets d'instances formant un rapport de proportion avec t sur ses propriétés connues. Dans notre exemple, pour $chanteur$ en entrée, cette recherche fournit les deux triplets $(râler, râleur, chanter)$, $(parler, parleur, chanter)$. Pour chaque triplet x, y, z, nous résolvons ensuite l'*équation analogique* $x : y :: z :?$ sur les propriétés à apprendre. Résoudre l'équation *VI* : *NM* :: *VI* : ? permet ainsi d'inférer *NM* comme traits morpho-syntaxiques de $chanteur$. Dans ce modèle, les propriétés peuvent être décrites sous différentes formes (vecteurs de $\mathbb{R}^n$, séquences, arbres) ; la seule contrainte est de savoir définir (et calculer) le rapport de proportion entre objets représentés sous cette forme.

Dans ce contexte, la contribution de ce travail est double. D'une part, nous proposons une définition unifiée de la notion de rapport de proportion applicable aux structures algébriques courantes (ensemble de séquences, d'arbres, etc.), puis nous montrons que cette définition donne lieu à un calcul efficace de ces proportions, calcul réalisé à l'aide de transducteurs à états finis. Dans le cas des mots et des arbres, notre définition généralise celles données par (Lepage, 2003). Elle repose sur la notion de *décomposition* d'objets en termes plus petits *alternant deux à deux*. Ainsi, le rapport de proportion $chanter : chanteur :: râler : râleur$ fait intervenir les termes $chant$, $râl$, er, eur alternant deux à deux. Formellement, la définition proposée s'applique à tout semi-groupe, et a fortiori aux groupes, aux monoïdes et aux treillis (Stroppa & Yvon, 2005b).

D'autre part, nous fournissons des résultats expérimentaux originaux (Stroppa & Yvon, 2005a) issus de l'application de notre modèle d'inférence à l'apprentissage automatique de propriétés morphologiques. Dans ce cadre, deux types d'expérimentations ont été effectuées sur des lexiques du français, de l'anglais, de l'allemand et du néerlandais. La première tâche consiste à inférer des traits morpho-syntaxiques à partir d'une forme graphique (cf exemple ci-dessus). Les résultats obtenus sont encourageants puisque les taux de rappel et de précision dépassent tous deux 97% sur les catégories grammaticales visées (noms, verbes et adjectifs). Dans la deuxième tâche, on infère une décomposition hiérarchique (arbre) de la forme en entrée. Ce type d'application montre la flexibilité de notre apprenti, le seul, à notre connaissance, capable d'apprendre à apparier des séquences avec des arbres sans construire au préalable de modèle grammatical. Les résultats obtenus, bien que significativement moins bons sur cette deuxième tâche, permettent néanmoins de capturer les phénomènes de constructions les plus réguliers.

Des expériences complémentaires restent à conduire pour mieux apprécier les réelles forces et faiblesses du modèle. De manière plus générale, les expériences menées ont permis de confirmer la faisabilité et la pertinence d'une approche à base d'analogies pour capturer les régularités présentes dans la morphologie des langues européennes.

Références

D. GENTNER, K. J. HOLYOAK & B. N. KOKINOV, Eds. (2001). *The Analogical Mind.* Cambridge, MA : The MIT Press.

LEPAGE Y. (2003). De l'analogie rendant compte de la commutation en linguistique. Habilitation à diriger les recherches, Grenoble, France.

STROPPA N. & YVON F. (2005a). Apprentissage par analogie et rapports de proportion : contributions méthodologiques et expérimentales. In *Conférence d'Apprentissage*. Version longue.

STROPPA N. & YVON F. (2005b). *Formal models of analogical proportions*. Rapport interne, à paraître, ENST, Paris, France.

Inférence grammaticale et grammaires catégorielles : vers la Grande Unification !

Isabelle Tellier

GRAppA & Inria Futurs, Lille
MOSTRARE project
Université Lille 3
59653 Villeneuve d'Ascq
France
isabelle.tellier@univ-lille3.fr

Résumé : Dans cet article, nous proposons de comparer les techniques employées en inférence grammaticale de langages réguliers par exemples positifs avec celles employées pour l'inférence de grammaires catégorielles. Pour cela, nous commençons par étudier la traduction entre automates et grammaires catégorielles, et inversement. Nous montrons ensuite que l'opérateur de généralisation utilisé pour l'apprentissage de grammaires catégorielles est strictement plus puissant que celui de fusion d'états (usuel en inférence grammaticale régulière), puisqu'il fait parfois sortir de la classe des langages réguliers. Nous proposons un nouveau modèle génératif qui généralise les automates à états finis pour représenter le résultat de cet opérateur. Nous montrons que ce modèle a au moins la même expressivité que les grammaires catégorielles unidirectionnelles, c'est-à-dire celle des langages algébriques. Enfin, nous exhibons une sous-classe des grammaires catégorielles unidirectionnelles pour laquelle l'apprentissage à partir de textes n'est presque pas plus coûteux que l'apprentissage à partir de structures.

1 Introduction

En inférence grammaticale, on étudie comment apprendre une grammaire à partir d'exemples de phrases qu'elle engendre -et de phrases qu'elle ne reconnaît pas, si des exemples négatifs sont disponibles. Les résultats fondateurs du domaine concernent l'apprenabilité de sous-classes de grammaires régulières, généralement représentées par des automates à états finis (Angluin, 1982; Oncina & Garcia, 1992; Dupont *et al.*, 1994; Denis *et al.*, 2002).

L'inférence de grammaires algébriques est évidemment plus difficile. Elle a fait l'objet de nombreux travaux, mais généralement plutôt empiriques. Or, nous aimerions l'envisager dans le modèle d'apprentissage à la limite par exemples positifs de Gold (Gold, 1967). Les résultats théoriques qui nous semblent les plus avancés du domaine sont dûs à Kanazawa (Kanazawa, 1998), qui a prouvé dans ce modèle l'apprenabilité de sous-classes de grammaires catégorielles de type AB (un formalisme syntaxique qui a

la même expressivité que les grammaires algébriques). Ces résultats ont malheureusement peu de portée pratique, notamment parce que, à part dans des cas très particuliers, les algorithmes d'apprentissage auxquels ils donnent lieu sont très coûteux (Florêncio, 2001; Florêncio, 2002). Il semble que personne, jusqu'à présent, n'ait essayé de représenter par des grammaires catégorielles les langages réguliers, pour voir en quoi les résultats les concernant recoupent ou non ceux connus en inférence de langages réguliers. C'est le point de départ de notre travail.

La première partie de cet article, après avoir introduit les définitions nécessaires, expose donc en détail comment passer d'un automate à états finis à une grammaire catégorielle, et réciproquement. Dans un deuxième temps, on tente d'utiliser cette traduction pour déduire des résultats d'apprenabilité nouveaux ou rapprocher des résultats obtenus indépendamment dans chacun des deux contextes. Mais la voie qui se révèle la plus prometteuse consiste à comparer ce que réalisent les algorithmes d'apprentissage eux-mêmes. Nous montrons ainsi que l'opérateur de généralisation utilisé dans le cadre de l'apprentissage de grammaires catégorielles est strictement plus puissant que celui de "fusion d'états" utilisé en inférence grammaticale régulière, puisqu'il peut faire sortir de la classe des langages régulier. Son effet sur un automate à états finis reste toutefois interprétable, à condition d'étendre la définition des automates considérés. Cette piste donne lieu à la dernière partie de l'article et aboutit notamment à la caractérisation d'une nouvelle sous-classe de grammaires catégorielles, dont l'apprentissage est nettement moins coûteux que ce que les algorithmes de Kanazawa laissaient craindre.

Cet article propose en quelque sorte de rapprocher les connaissances et les techniques issues de deux communautés distinctes, en montrant qu'elles peuvent chacune tirer bénéfice des travaux de l'autre.

2 Automates à états finis et grammaires catégorielles

Dans cette section, nous introduisons les définitions nécessaires et nous explicitons les correspondances entre automates finis et grammaires catégorielles.

2.1 Automates à états finis et langages réguliers

Définition 1 (Automates à états finis (AF) et langages réguliers)
Un ***automate à états finis*** *(AF par la suite)* A *est un quintuplet* $A = \langle Q, \Sigma, \delta, q_0, F\rangle$ *où* Q *est l'ensemble fini des* états *de* A, Σ *est son* vocabulaire *fini, son* état initial *est* $q_0 \in Q$ *(nous nous restreignons ici aux automates avec un unique état initial) et* $F \subseteq Q$ *est l'ensemble de ses* états finaux*. Enfin,* δ *est la* fonction de transition *de* A*, définie de* $Q \times \Sigma$ *vers* 2^Q*.*

Le langage $L(A)$ *reconnu (ou engendré) par* A *est défini par :* $L(A) = \{w \in \Sigma^* | \delta^*(q_0, w) \cap F \neq \emptyset\}$*, où* δ^* *est l'extension naturelle de* δ *à* $Q \times \Sigma^*$ *telle que : pour tout* $a \in \Sigma$*, tout* $u \in \Sigma^*$ *et tout* $q \in Q$*,* $\delta^*(q, au) = \{\delta^*(q', u) | q' \in \delta(q, a)\}$*.*
L'ensemble de tous les langages reconnus par un AF est appelé l'ensemble des ***langages réguliers****.*

Exemple 1
La Figure 1 montre un AF A tel que $L(A) \overline{=} a^+b^+$.

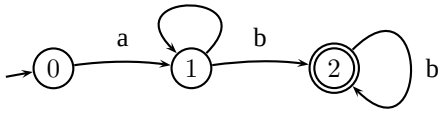


FIG. 1 – un automate à états finis

2.2 Grammaires formelles et grammaires catégorielles

Nous rappelons ici la définition classique des grammaires formelles ainsi que celle, moins connue, des grammaires catégorielles de type AB (en référence à leurs fondateurs, Adjukiewicz et Bar-Hillel), qui seront les seules considérées par la suite.

Définition 2 (Grammaires formelles et leur langage)
*Une **grammaire formelle** (ou tout simplement une grammaire) G est un quadruplet $G = \langle \Sigma, N, P, S \rangle$ où Σ est le* vocabulaire terminal *fini de G, N son* vocabulaire non terminal*, lui aussi fini, $P \subset (\Sigma \cup N)^+ \times (\Sigma \cup N)^*$ son ensemble fini de* règles de réécriture *et $S \in N$ son* axiome*.*

Le langage $L(G)$ reconnu (ou engendré) par G est défini par : $L(G) = \{w \in \Sigma^* | S \longrightarrow^* w\}$ où $\longrightarrow^*$ est la clôture réflexive et transitive de la relation définie par P.

Définition 3 (Catégories, grammaires catégorielles de type AB et leur langage)
Soit $\mathcal{B}$ un ensemble au plus dénombrable de catégories de base*, parmi lesquelles figure une* catégorie distinguée *$S \in \mathcal{B}$, appelée l'axiome. L'ensemble des **catégories fondées sur** $\mathcal{B}$, noté $Cat(\mathcal{B})$, est le plus petit ensemble tel que $\mathcal{B} \subset Cat(\mathcal{B})$ et pour tout $A, B \in Cat(\mathcal{B})$ on a : $A/B \in Cat(\mathcal{B})$ et $B\backslash A \in Cat(\mathcal{B})$.*

*Pour tout vocabulaire fini Σ (dont les membres seront appelés des mots) et pour tout ensemble $\mathcal{B}$ de catégories de base ($S \in \mathcal{B}$), une **grammaire catégorielle** G est une relation finie sur $\Sigma \times Cat(\mathcal{B})$. On note $\langle v, A \rangle \in G$ l'affectation de la catégorie $A \in Cat(\mathcal{B})$ au mot $v \in \Sigma$. Une* grammaire catégorielle de type AB *(ou simplement une GC par la suite) est une grammaire catégorielle dont les règles de réécriture sont réduites aux deux* schémas applicatifs *suivants : $\forall A, B \in Cat(\mathcal{B})$*

- *FA (Forward Application) : $A/B \;\; B \rightarrow A$*
- *BA (Backward Application) : $B \;\; B\backslash A \rightarrow A$*

Le langage $L(G)$ reconnu (ou engendré) par G est défini par : $L(G)$=$\{w = v_1 \dots v_n \in \Sigma^+ \mid \forall i \in \{1, \dots, n\}, \exists A_i \in Cat(\mathcal{B})$ tel que $\langle v_i, A_i \rangle \in G$ et $A_1 \; \dots \; A_n \rightarrow^* S\}$, où $\rightarrow^*$ est la clôture réflexive et transitive de la relation définie par FA et BA.

*Pour tout entier $k \geq 1$, l'ensemble des GCs qui affectent au plus k catégories distinctes à chacun des mots de leur vocabulaire est la classe des **GCs k-valuées**, et est notée $\mathcal{G}_k$. Les GCs de $\mathcal{G}_1$ sont aussi dîtes* rigides*.*

La particularité principale des GCs est qu'elles sont *lexicalisées*, au sens où la totalité de l'information syntaxique est associée aux mots du vocabulaire (puisque les règles sont, elles, définies une fois pour toutes et invariables d'une grammaire à une autre).

Exemple 2
Les GCs ont surtout été utilisées pour la modélisation des langues naturelles. Soit par exemple $\mathcal{B} = \{S, T, NC\}$ *(où* T *désigne la catégorie des "termes" et NC celle des "noms communs"),* $\Sigma = \{Paul, dort, aime, un, chat\}$ *et* G *la GC définie par :* $G = \{\langle Paul, T\rangle, \langle dort, T\backslash S\rangle, \langle aime, (T\backslash S)/T\rangle, \langle chat, NC\rangle, \langle un, T/NC\rangle, \}$. *G reconnaît des phrases comme "Paul dort", "Paul aime un chat", etc.*

Définition 4 (FA-Structures, exemple structuré, langage des structures)
Une ***FA-structure*** *(FA pour "Foncteur-Argument") sur un vocabulaire* Σ *est un arbre binaire dont les feuilles sont étiquetées par des éléments de* Σ *et dont chaque noeud est étiqueté soit par* BA *soit par* FA. *L'ensemble des FA-structures sur* Σ *est noté* Σ^F.
Pour toute GC $G \subset \Sigma \times Cat(\mathcal{B})$, *un* ***exemple structuré*** *pour* G *est un élément de* Σ^F *qui est obtenu à partir d'un arbre d'analyse syntaxique produit par* G *pour une phrase* $w \in L(G)$, *en effaçant dans cet arbre toutes les catégories de* $Cat(\mathcal{B})$. *Pour toute GC* G, *le* ***langage des structures*** *de* G, *noté* $FL(G)$ *est l'ensemble de ses exemples structurés.*

2.3 Des automates aux grammaires catégorielles et inversement

Les GCs peuvent engendrer tous les langages algébriques sans ϵ (où ϵ désigne le mot vide) (Bar-Hillel *et al.*, 1960). Elles peuvent donc aussi engendrer tous les langages réguliers sans ϵ. Les correspondances entre AFs et GCs sont faciles à définir.

Définition 5 (GCs régulières)
Nous appelons ***GC régulière*** *une CG* $G \subset \Sigma \times Cat(\mathcal{B})$ *qui ne contient que des affectations de la forme* $\langle v, A\rangle$ *ou* $\langle v, A/B\rangle$ *avec* $v \in \Sigma$ *et* $A, B \in \mathcal{B}$. *L'ensemble des GCs régulières est noté* $\mathcal{G}_r$.

Propriété 1 (Transformation d'un AF en GC)
Soit $A = \langle Q, \Sigma, \delta, q_0, F\rangle$ *un AF. Soit* $\mathcal{B} = (Q\backslash\{q_0\}) \cup \{S\}$ *(où* $S \notin Q$*). Il est possible de définir une GC régulière* $G \subset \Sigma \times Cat(\mathcal{B})$ *telle que* $L(G) = L(A)\backslash\{\epsilon\}$.

Preuve 1 (Preuve de la Propriété 1)
Cette propriété découle de la transformation classique d'un AF en une grammaire linéaire à gauche $G_1 = \langle \Sigma, Q, P_1, S\rangle$, *que l'on peut ensuite transformer en une GC* $G \in \mathcal{G}_r$. *On remplace* q_0 *par* S *puis* $\forall a \in \Sigma$ *et* $\forall q, q' \in Q$ *tels que* $q' \in \delta(q, a)$ *on fait :*
- *si* $q' \in F$ *alors*
  - $q \longrightarrow a$ *est une règle de* G_1 *(élément de* P_1*) et* $\langle a, q\rangle \in G$ *;*
  - *si* $\exists u \in \Sigma$ *tel que* $\exists q'' \in \delta(q', u)$ *alors* $q \longrightarrow aq'$ *est une règle de* G_1 *et* $\langle a, q/q'\rangle \in G$ *;*
- *sinon* $q \longrightarrow aq'$ *est une règle de* G_1 *et* $\langle a, q/q'\rangle \in G$.

$\epsilon \in L(A)$ *si et seulement si* $q_0 \in F$. *Cette situation amène à ajouter* $S \longrightarrow \epsilon$ *dans* P_1. *Mais, dans une GC, il est impossible d'affecter une catégorie à* ϵ. *Cette règle n'a donc pas de contrepartie dans* G *et nous avons :* $L(A)\backslash\{\epsilon\} = L(G_1)\backslash\{\epsilon\} = L(G)$.□

Exemple 3
En appliquant la propriété précédente à l'AF de l'Exemple 1, nous obtenons les règles de G_1 *suivantes (où l'état* i *est associé à un non-terminal noté* q_i, *avec* $q_0 = S$*) :*

$S \longrightarrow aq_1$, $q_1 \longrightarrow aq_1$, $q_1 \longrightarrow b$, $q_1 \longrightarrow bq_2$, $q_2 \longrightarrow b$, $q_2 \longrightarrow bq_2$. *La GC obtenue est :* $G = \{\langle a, S/q_1\rangle, \langle a, q_1/q_1\rangle, \langle b, q_1\rangle, \langle b, q_1/q_2\rangle, \langle b, q_2\rangle, \langle b, q_2/q_2\rangle\}$.

Les AF peuvent donc simplement être *lexicalisés*, sous la forme d'une GC. Remarquons que, dans les GCs régulières que nous avons définies, seul l'opérateur / est utilisé dans les catégories affectées aux mots du vocabulaire, et seul le schéma FA est utile (on aurait pu aussi se restreindre à l'opérateur $\backslash$ et au schéma BA en transformant l'AF en une grammaire linéaire à gauche). En fait, *la transformation précédente ne préserve pas seulement le langage final, mais aussi les structures d'analyse, qui sont des peignes*. Ainsi, les exemples structurés, tels que définis dans la Définition 4 sont en quelque sorte disponibles "gratuitement" à partir des phrases.

Exemple 4
La Figure 2 montre deux arbres d'analyse syntaxique produits par la CG de l'Exemple 3, et (à droite) les exemples structurés correspondant.

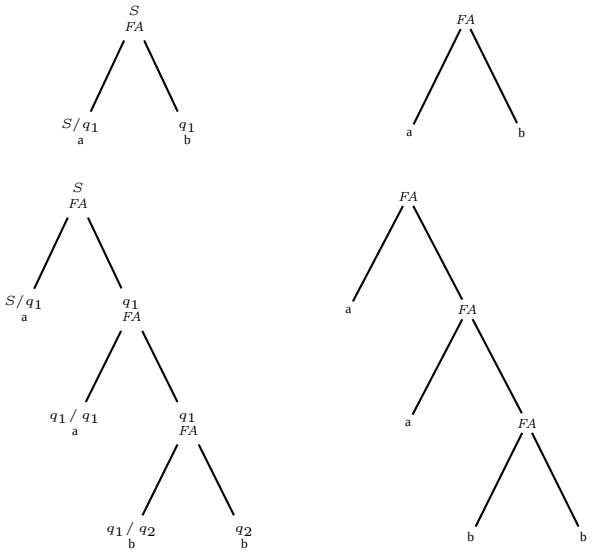


FIG. 2 – Deux analyses syntaxiques et les exemples structurés correspondant

Propriété 2 (Transformation d'une GC régulière en un AF)
Toute GC régulière $G \subset \Sigma \times Cat(\mathcal{B})$ *peut être transformée en un AF* $A = \langle Q, \Sigma, \delta, q_0, F\rangle$ *reconnaissant le même langage (sans* ϵ*).*

Preuve 2 (Schéma de preuve de la Propriété 2)
Cette transformation est l'inverse de celle exposée dans la Propriété 1 : le seul point notable est qu'il est plus facile d'ajouter un unique état final F_A *dans* A*. Soit donc* $Q = \mathcal{B} \cup \{F_A\}$ *avec* $F_A \notin \mathcal{B}$, $q_0 = S$ *et* $F = \{F_A\}$*. Chaque affectation* $\langle a, U/V\rangle \in G$ *correspond à une transition étiquetée par* a *entre les états* U *et* V *dans* A *(soit encore :*

$\delta(U, a) = V$) *et chaque affectation* $\langle a, U\rangle \in G$ *à une transition étiquetée par* a *entre* U *et* F_A *(*$\delta(U, a) = F_A$*).*□

Exemple 5
L'AF obtenu en appliquant cette opération à la GC de l'Exemple 3 est donné en Figure 3. Il ne coincide pas exactement avec celui de l'Exemple 1 à cause de l'état final ajouté à ceux provenant des catégories de base. Le résultat est, de ce fait, non déterministe.

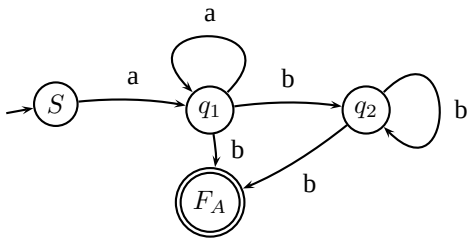


FIG. 3 – AF obtenu à partir d'une GC

Remarque 1
Les Propriétés 1 et 2 ne signifient pas que seules les GCs régulières génèrent des langages réguliers. Celle de l'Exemple 2 n'est pas régulière au sens de la Définition 5 mais elle génère un langage fini (donc régulier). Mais elle ne produit pas que des peignes.

Propriété 3 (Langage associé à une catégorie ou à un état)
Soit G *une GC régulière et* A *l'AF obtenu à partir de* G*. Alors, pour toute catégorie de base* $q \in \mathcal{B}$ *(correspondant à un état non final* $q \in Q$ *de* A*), nous avons deux façons distinctes de caractériser le langage* $L(q)$ *associé à* q *:*
$L(q) = \{w = v_1 \dots v_n \in \Sigma^+ \mid \forall i \in \{1, \dots, n\}\ \exists A_i \in Cat(\mathcal{B})$ *tel que* $\langle v_i, A_i\rangle \in G$ *et* $A_1\ \dots\ A_n \rightarrow^* q\}$ *et* $L(q) = \{w \in \Sigma^+ | F_A \in \delta^+(q, w)\}$*.*

Preuve 3 (Schéma de preuve de la Propriété 3)
Cette propriété est une conséquence triviale des Propriétés 1 et 2, où q *remplace* S*.*□

La seconde définition de $L(q)$ n'est correcte que parce que nous savons que $F_A \neq q$ est l'unique état final de A. Ainsi, les successions de mots (ou chaînes) auxquelles G associe la catégorie q sont celles qui correspondent dans A à un chemin qui part de l'état q et aboutit à l'état F_A. Bien sûr, si $q = q_0 = S$, on retrouve $L(S) = L(A) = L(G)$.

Exemple 6
Dans l'AF de la Figure 3, $L(q_1) = a^*b^+$ *et* $L(q_2) = b^+$*.*

3 Inférence de GCs par exemples positifs

L'étude de l'apprenabilité des GCs par exemples positifs seuls au sens de Gold (Gold, 1967) a donné lieu à de nombreux travaux récents, initiés par Kanazawa (Kanazawa,

1998). Maintenant que nous disposons d'un mécanisme de traduction d'une sous-classe des GCs (les GCs régulières) en AFs, il est naturel de se demander si des résultats concernant l'apprenabilité de ces GCs peuvent se traduire en résultats d'apprenabilité de langages réguliers.

3.1 Modèle de Gold

Définition 6 (Critère d'apprenabilité)
Soit $\mathcal{G}$ un ensemble de grammaires (par la suite, $\mathcal{G}$ sera une sous-classe de l'ensemble des GCs) sur un alphabet Σ et soit une fonction qui associe un langage à chaque grammaire. Cette fonction sera soit le langage des chaînes $L : \mathcal{G} \longrightarrow pow(\Sigma^*)$ (cf. Définition 3) soit le langage des structures $FL : \mathcal{G} \longrightarrow pow(\Sigma^F)$ (cf. Définition 4).

*Soit ϕ une fonction qui à tout échantillon fini de phrases de Σ^* (resp. d'exemples structurés de Σ^F) associe une grammaire de $\mathcal{G}$. On dit que cette fonction **converge** vers $G \in \mathcal{G}$ sur un échantillon $\langle s_i \rangle_{i\in\mathbb{N}}$ d'éléments de Σ^* (resp. de Σ^F) si $G_i = \phi(\langle s_0, ..., s_i \rangle)$ est défini et égal à G pour un nombre infini de valeurs de $i \in \mathbb{N}$ ou, ce qui revient au même, si $\exists n_0 \in \mathbb{N}$ tel que pour tout $i \geq n_0$, G_i et défini et égal à G.*

*On dit que ϕ **apprend** la classe $\mathcal{G}$ par exemples positifs (resp. par exemples structurés positifs) si pour tout langage L de $L(\mathcal{G}) = \{L(G)|G \in \mathcal{G}\}$ (resp. de $FL(\mathcal{G}) = \{FL(G)|G \in \mathcal{G}\}$) et pour tout séquence $\langle s_i \rangle_{i\in\mathbb{N}}$ qui énumère L, c'est-à-dire telle que $\{s_i|i \in \mathbb{N}\} = L$, il existe $G \in \mathcal{G}$ telle que $L = L(G)$ (resp. $L = FL(G)$) et ϕ converge vers G sur $\langle s_i \rangle_{i\in\mathbb{N}}$.*

*$\mathcal{G}$ est **apprenable** s'il existe une fonction ϕ calculable qui apprend $\mathcal{G}$.*

Théorème 1 (Apprenabilité de $\mathcal{G}_k$ (Kanazawa, 1998))
Pour tout entier $k \geq 1$, la classe des GCs k-valuées $\mathcal{G}_k$ est apprenable par exemples positifs et par exemples structurés positifs.

3.2 Apprentissage de GCs et inférence grammaticale régulière

Dans cette section, nous traduisons les résultats d'apprenabilité de Kanazawa dans la classe des GCs régulières, puis, nous rapprochons deux résultats connus.

Théorème 2 (Apprenabilité des GCs régulières k-valuées)
Pour tout entier $k \geq 1$, la classe des GCs régulières k-valuées $\mathcal{G}_k \cap \mathcal{G}_r$ est apprenable par exemples positifs et par exemples structurés positifs.

Preuve 4 (Schéma de preuve du Théorème 2)
Ce théorème est une conséquence du Théorème 1, restreint à la classe $\mathcal{G}_r$. Tout algorithme d'apprentissage de $\mathcal{G}_k$ à partir de structures peut être adapté pour devenir un algorithme d'apprentissage de $\mathcal{G}_k \cap \mathcal{G}_r$ à partir de chaînes, en associant des peignes avec noeuds FA aux chaînes et en ne conservant que les sorties de la fonction d'apprentissage qui sont isomorphes à des grammaires régulières (ce qui est décidable). □

Bien sûr, $\bigcup_{k\geq 1}\{L(G)|G \in \mathcal{G}_k \cap \mathcal{G}_r\}$ contient tous les langages réguliers. Mais le principal intérêt de cette classe est que, contrairement au cas général, elle tout aussi

facile à apprendre à partir de chaînes qu'à partir d'exemples structurés. Cette propriété sera étendue au delà des GCs régulières dans la section suivante.

Cependant le résultat d'apprenabilité lui-même n'est pas très surprenant. En effet, pour tout entier k, les AFs qui traduisent les GCs de $\mathcal{G}_k \cap \mathcal{G}_r$ (suivant le processus décrit dans la preuve de la Propriété 2) ont au plus k transitions étiquetées par le même mot du vocabulaire. Or, le nombre d'AFs vérifiant cette propriété est fini. L'apprentissage par exemples positifs d'un ensemble fini est toujours possible.

Néanmoins, cette classe d'automates a des caractéristiques originales. Elle inclut tout aussi bien des AFs déterministes que des AFs non-déterministes. Et, contrairement aux classes habituellement considérées en inférence grammaticale régulière, elle semble bien adaptée aux grands alphabets (à condition que k soit petit).

Pour compléter le rapprochement que nous avons commencé à établir entre AFs et GCs, nous rapprochons dans ce qui suit deux résultats connus.

Définition 7 (AFs 0-réversibles (Angluin, 1982))
Un AF est dit 0-réversible si et seulement si il est déterministe, et l'AF obtenu en inversant le sens de ses transitions, et en échangeant les rôles des états initiaux et finaux est déterministe. La classe des AFs 0-réversibles est apprenable par exemples positifs.

Définition 8 (CG réversible (Besombes & Marion, 2004))
Une GC est dite réversible si elle ne contient pas deux affectations de catégories pour un même mot du vocabulaire, qui ne sont distinctes que par une seule catégorie de base. La classe des GCs réversibles est apprenable à partir d'exemples structurés et de chaînes.

Théorème 3 (Equivalence entre ces deux notions de réversibilité)
Soit G une GC régulière et A l'AF obtenu à partir de G. A est 0-réversible dans le sens de la Définition 7 si et seulement si G est réversible dans le sens de la Définition 8.

Preuve 5 (Preuve du Théorème 3)
Par construction, A n'a qu'un seul état initial et aucune transition étiquetée par ϵ. Pour que A soit déterministe, il suffit donc qu'il n'existe pas deux transitions avec la même étiquette au départ d'un même état. Cette condition se traduit dans G par : $\forall a \in \Sigma$
- *$\forall Q_1, Q_2, Q_3 \in \mathcal{B}$: $\langle a, Q_1/Q_2\rangle \in G$ et $\langle a, Q_1/Q_3\rangle \in G \Leftrightarrow Q_2 = Q_3$.*
- *$\forall Q_1, Q_2 \in \mathcal{B}$: $\langle a, Q_1\rangle \in G$ et $\langle a, Q_1/Q_2\rangle \in G \Leftrightarrow Q_2 = F_A$ ($\langle a, Q_1\rangle$ joue en quelque sorte le rôle de $\langle a, Q_1/F_A\rangle$ et aucune transition ne part de F_A) ;*

De même, A n'a qu'un seul état final donc la condition pour que son inverse soit déterministe se traduit dans G de la façon suivante : $\forall a \in \Sigma$
- *$\forall Q_1, Q_2 \in \mathcal{B}$: $\langle a, Q_1\rangle \in G$ et $\langle a, Q_2\rangle \in G \Leftrightarrow Q_1 = Q_2$*
- *$\forall Q_1, Q_2, Q_3 \in \mathcal{B}$: $\langle a, Q_1/Q_2\rangle \in G$ et $\langle a, Q_3/Q_2\rangle \in G \Leftrightarrow Q_1 = Q_3$.*

Pour les GCs régulières, ces conditions coïncident avec celles de la Définition 8.□

Ainsi, le résultat d'apprenabilité de la classe des AFs 0-réversibles à partir de chaînes (Angluin, 1982) se déduit du résultat d'apprenabilité de la classe des GCs régulières réversibles par exemples structurés, qui lui-même découle de (Besombes & Marion, 2004). Là encore, ce résultat n'est pas très surprenant : (Besombes & Marion, 2004) se sont inspirés des grammaires algébriques réversibles de (Sakakibara, 1992) qui, lui-même, s'était inspiré de (Angluin, 1982).

3.3 Algorithme d'apprentissage

Les résultats d'apprenabilité du théorème 1 ne sont pas uniquement théoriques : ils s'accompagnent de la définition d'une fonction d'apprentissage originale. Elle est fondée sur un algorithme que nous appellerons BP, en hommage à ses inventeurs (Buszkowski & Penn, 1990), qui est capable, pour tout entier k et tout ensemble d'exemples structurés D, d'identifier l'ensemble des GCs k-valuées sans catégorie inutile (cf. définition 12 plus loin) compatibles avec D. Nous rappelons ici le principe de cet algorithme, en l'illustrant sur un exemple où les éléments de D sont des peignes avec FA pour noeuds internes. Pour tout élément de D, les premières étapes de BP consistent à :

1. introduire l'étiquette S à la racine de chaque exemple structuré ;
2. introduire une variable distincte x_i à chaque noeud argument (c'est-à-dire, au fils gauche de chaque noeud BA, et au fils droit de chaque noeud FA) ;
3. introduire à chaque autre noeud la catégorie qui rend possible l'application des schémas FA et BA qui étiquettent ces noeuds.

La GC définie en récoltant les affectations de catégories aux feuilles des éléments de D à la fin de ces étapes est appelée **forme générale** de D et notée $FG(D)$.

Exemple 7
Soit D défini comme l'ensemble des deux exemples structurés de l'Exemple 4. Les étapes 1 à 3 décrites précédemment aboutissent au résultat donné par la Figure 4, et $FG(D)$ est alors définie comme suit :
– *a :* $S/x_1,\ S/x_4,\ x_4/x_3$ *;*
– *b :* $x_1,\ x_3/x_2,\ x_2$.
*L'AF correspondant est donné en Figure 5 : il est presque identique à ce que, en inférence grammaticale, on appelle **automate canonique minimal** reconnaissant $\{ab, aabb\}$. La seule différence est que notre AF a un unique état initial et un unique état final.*

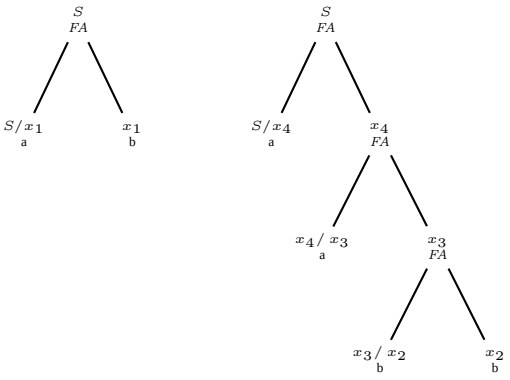


FIG. 4 – Résultat de l'application des 3 premières étapes de BP

La suite de l'algorithme BP consiste à chercher des *substitutions unificatrices* applicables sur $FG(D)$. On précise donc tout d'abord ce que sont ces substitutions.

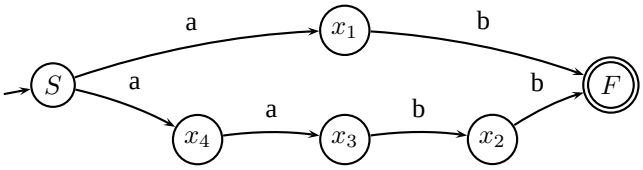


FIG. 5 – l'AF correspondant à $FG(D)$

Définition 9 (Catégories avec variables et substitutions)
*Soit χ un ensemble infini dénombrable de variables et soit $\mathcal{B} = \chi \cup \{S\}$. Une **substitution** est une fonction $\sigma : \chi \longrightarrow Cat(\mathcal{B})$ qui transforme une variable en une catégorie (par défaut, elle vaut l'identité sur χ). Une substitution est étendue par morphisme à une fonction de $Cat(\mathcal{B})$ dans $Cat(\mathcal{B})$ de la manière suivante : (i) $\sigma(S) = S$, (ii) $\sigma(A/B) = \sigma(A)/\sigma(B)$ et (iii) $\sigma(A\backslash B) = \sigma(A)\backslash\sigma(B)$ pour tout $A, B \in Cat(\mathcal{B})$. De même, une substitution peut être étendue à une GC quelconque G : $\sigma(G) = \{\langle v, \sigma(A)\rangle | \langle v, A\rangle \in G\}$. Pour toute GC G, une **substitution unificatrice** de G est une substitution qui unifie des catégories affectées à un même mot du vocabulaire de G.*

Propriété 4 (Propriété Fondamentale (Buszkowski & Penn, 1990))
Pour toute GC G, les propriétés suivantes sont équivalentes : (i) $D \subseteq FL(G)$ et (ii) $\exists\sigma$ telle que $\sigma[GF(D)] \subseteq G$.

En d'autres termes, les GCs compatibles avec un ensemble d'exemples structurés D sont celles qui incluent une substitution de $FG(D)$. Quand il cible des GCs k-valuées, l'étape 4 de l'algorithme BP consiste donc à chercher *toutes les substitutions unificatrices de $FG(D)$ qui sont dans $\mathcal{G}_k$*. Le résultat de BP est donc un *ensemble de GCs*. Si D est constitué d'exemples structurés et $k = 1$, la grammaire qui a produit D, si elle existe, est unique et isomorphe, à la limite, au résultat de BP : BP est alors un algorithme d'apprentissage efficace. Si $k > 1$, le résultat de BP n'est en général pas réduit à un singleton. Pour obtenir une fonction d'apprentissage au sens de Gold, il faut donc être capable de faire un choix parmi ses éléments. Cette étape nécessite de réaliser des tests d'inclusion et est très coûteuse en temps de calcul. Nous ne la détaillerons pas ici.

Appliquer une substitution à une GC est une *opération de généralisation*. En effet, on a la propriété suivante (Buszkowski & Penn, 1990) : $\sigma(G_1) \subseteq G_2 \Longrightarrow FL(G_1) \subseteq FL(G_2)$, qui implique que : $FL(G) \subseteq FL(\sigma(G))$. De même : $L(G) \subseteq L(\sigma(G))$. Cette opération a-t-elle un lien avec celle de fusion d'états, qui est utilisée en inférence grammaticale régulière (Angluin, 1982; Oncina & Garcia, 1992; Dupont *et al.*, 1994) ? C'est ce que nous allons voir maintenant. Dans le cas d'un ensemble D constitué de peignes avec noeuds internes FA, $FG(D)$ est toujours une GC régulière. Donc, pour unifier deux catégories de $FG(D)$, seuls deux cas peuvent se produire :

- des conditions de la forme $\sigma(x_i) = \sigma(x_j) = x_j$ pour x_i et x_j dans χ spécifient précisément une fusion des états x_i et x_j dans l'AF correspondant à $FG(D)$.
- des conditions de la forme $\sigma(x_i) = x_j/x_k$, pour $x_i \in \chi$ et $x_j, x_k \in \chi \cup \{S\}$ sont, au premier abord, plus difficiles à interpréter. Elles signifient en fait deux choses :
  - l'état x_i est renommé en x_j/x_k ;

– toutes les chaînes auxquelles $FG(D)$ affectait la catégorie x_i peuvent désormais être utilisées comme “transition” pour passer de l'état x_j à l'état x_k.

Exemple 8
Définissons une substitution unificatrice σ pour la $FG(D)$ obtenue dans l'Exemple 7 comme suit : $\sigma(x_4) = \sigma(x_1) = x_3/x_2$ (et σ est l'identité partout ailleurs). La GC $\sigma(GF(D))$ est alors définie par :
– *a : $S/(x_3/x_2),\ (x_3/x_2)/x_3$;*
– *b : $x_3/x_2,\ x_2$.*
Cette GC n'est plus régulière, à cause des catégories affectées à a. Néanmoins, on peut encore lui associer un automate en intégrant à celui-ci une “transition récursive”, c'est-à-dire une transition étiquetée non plus par un élément du vocabulaire, mais par un état (ici, x_3/x_2). L'automate obtenu est celui de la Figure 6.

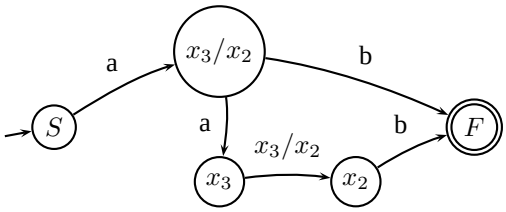


FIG. 6 – L'automate généralisé correspondant à $\sigma(FG(D))$

Dans cet automate, les états nommés x_1 et x_4 dans l'AF de la Figure 5 ont été fusionnés, sous l'effet de la condition $\sigma(x_4) = \sigma(x_1)$. La condition $\sigma(x_1) = x_3/x_2$ a, elle, eu pour effet de renommer cet état en x_3/x_2, et de remplacer la transition étiquetée par b entre x_3 et x_2 par une transition étiquetée par x_3/x_2 : c'est ce que nous appelons une transition récursive*. Pour la franchir, il faut produire une chaîne de catégorie x_3/x_2 c'est-à-dire, d'après la Propriété 3, une chaîne qui correspond à un chemin qui part de l'état x_3/x_2 et aboutit à l'état final F. Le premier exemple d'un tel chemin, c'est évidemment la transition étiquetée par b qui relie x_3/x_2 à F. Mais, partant de l'état x_3/x_2, il est aussi possible d'emprunter d'abord la transition étiquetée par a qui mène à x_3, avant de franchir une nouvelle fois la transition récursive. Une pile (implicite) est nécessaire pour enregistrer tous les appels récursifs successifs issus du franchissement de cette transition. Le langage reconnu par cet automate généralisé n'est autre que $a^n b^n$ qui, bien sûr, n'est pas régulier. On peut rapprocher ce dispositif de celui des Réseaux de Transitions Récursifs ou RTRs (Woods, 1970), mais réduit à un seul automate (dans les RTRs, il y a autant d'automates que de symboles non terminaux).*

L'arbre d'analyse syntaxique associé à la chaîne $aaabbb$ par $\sigma(FG(D))$ est donné Figure 7. Cet arbre n'est plus un peigne. Pour comprendre sa construction, le mieux est de se reporter à l'arbre d'analyse syntaxique associé à $aabb$ par $FG(D)$, en Figure 4. Dans ce dernier arbre, on voit que x_4 étiquette un noeud interne, tandis que x_3/x_2 étiquette une feuille. La condition $\sigma(x_4) = x_3/x_2$ ouvre donc la possibilité de substituer le sous-arbre de racine x_4 à la place de la feuille x_3/x_2. Cette opération, qui coïncide

exactement avec ce qui est appelé une ajonction *dans le formalisme des Tree Adjoining Grammars (Joshi & Schabes, 1997), produit comme résultat l'arbre de la Figure 7.*

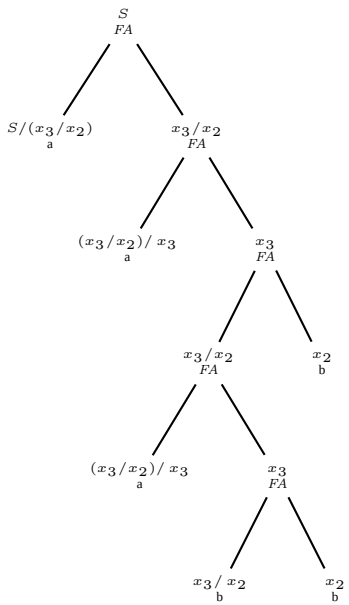


FIG. 7 – Arbre d'analyse syntaxique de $aaabbb$ par $\sigma(FG(D))$

4 Apprentissage de langages à partir de peignes

L'Exemple 8 suggère qu'il est possible de produire de vrais arbres par adjonctions de peignes, et de représenter les langages algébriques par des automates généralisés. Cette section est consacrée à la formalisation de ces idées, et à leur exploitation pour améliorer l'algorithme d'apprentissage de GCs à partir de chaînes de Kanazawa.

4.1 Automates récursifs et leur expressivité

Définition 10 (Automate récursif)
Un ***automate récursif*** *(AR par la suite)* R *est un quintuplet* $R = \langle Q, \Sigma, \gamma, q_0, F\rangle$ *où* Q *est l'ensemble fini des* états *de* R*,* Σ *son* vocabulaire *fini, son unique* état initial *est* $q_0 \in Q$ *et* $F \in Q$ *son (unique)* état final. γ *est la* fonction de transition *de* R*, définie de* $Q \times (\Sigma \cup Q)$ *vers* 2^Q.

La seule différence importante entre les ARs et les AFs est que, dans un AR, les transitions peuvent être étiquetées soit par un élément de Σ, soit par un élément de Q. Dans ce dernier cas, on parle de **transition récursive**. Pour franchir une transition récursive, il faut produire un élément du langage de l'état qui étiquette la transition.

Nous nous restreignons ici aux ARs qui ont un unique état initial et un unique état final, mais cela ne restreint pas l'expressivité du modèle. Comme nous nous intéresserons aux langages sans ϵ, on peut supposer de plus que $F \neq q_0$.

Définition 11 (Language Reconnu par un AR)
Le langage $L(R)$ reconnu (ou engendré) par un AR $R = \langle Q, \Sigma, \gamma, q_0, F\rangle$ est le plus petit ensemble défini par : $L(R) = \{w \in \Sigma^+ | F \in \gamma^+(q_0, w)\}$, où γ^+ est l'extension naturelle de γ à $Q \times \Sigma^+$. Pour pour tout $u \in \Sigma^+$, tout $v \in \Sigma^*$ et tout $q \in Q$, $\gamma^+(q, uv)$ est défini comme le plus petit sous-ensemble contenant $\{\gamma^*(q', v) | q' \in \gamma(q, u)\}$ si $u \in \Sigma$ et $\{\gamma^*(q', v) | \exists t \in Q$ tel que $q' \in \gamma(q, t)$ et $u \in L(t)\}$ sinon, où $L(t)$ est le langage de l'état t, défini comme en Définition 3 en remplaçant δ par γ. Cette définition de $L(G)$ est récursive : $L(G)$ est défini comme un plus petit point fixe, quand il existe.

Une transition récursive étiquetée par un état $q \in Q$ sera dite *vraiment récursive* s'il existe un chemin qui mène de l'état q à F en passant par cette transition. Les transitions *vraiment récursives* sont celles qui permettent des adjonctions et produisent donc des *structures* qui ne sont pas nécessairement des peignes.

Théorème 4 (Des GCs unidirectionnelles aux ARs)
Une GC unidirectionnelle n'affecte que des catégories qui sont soit de base, soit construites avec l'opérateur $/$ uniquement. L'ensemble des GCs unidirectionnelles sera noté $\mathcal{G}_/$. Pour toute GC dans $G \in \mathcal{G}_/$, on peut construire un AR fortement équivalent à G, c'est-à-dire produisant les mêmes structures.

Preuve 6 (Schéma de la preuve du Théorème 4)
Il est connu depuis longtemps que toute GC $G \subset \Sigma \times Cat(\mathcal{B})$ peut être transformée en une grammaire algébrique $H = \langle \Sigma, N, P, S\rangle$ sous forme normale de Chomsky fortement équivalent à G. H est construite de la façon suivante : N est l'ensemble de toutes les sous-catégories d'une catégorie présente dans les affectations de G (une catégorie est une sous-catégorie d'elle-même) et P contient toutes les règles de la forme $A \longrightarrow v$ pour tout $\langle v, A\rangle \in G$ et toutes les règles de la forme $A \longrightarrow A/B\ B$ pour toute catégorie A et B dans N (pour les GC unidirectionnelles, cela suffit). Pour construire un AR à partir de ces règles, il suffit de procéder exactement comme avec les règles régulières utilisées dans la Preuve de la Propriété 2.□

Corollaire 1 (Corollaire du Théorème 4)
Les grammaires de $\mathcal{G}_/$ peuvent produire tous les langages algébriques sans ϵ (Bar-Hillel et al.*, 1960). Donc, les ARs peuvent également produire tous ces langages.*

Exemple 9
La CG de $\mathcal{G}_/$ classique qui reconnaît $a^n b^n$, $n \geq 1$, est : $\{\langle a, S/B\rangle, \langle a, (S/B)/S\rangle, \langle b, B\rangle\}$. L'AR correspondant (distinct de celui de Example 8) est donné en Figure 8. Cet AR peut être simplifié : les transitions récursives qui ne sont pas vraiment récursives peuvent être lexicalisées. Ici, on peut effacer l'état $(S/B)/S$ et remplacer la transition récursive qui y fait référence entre S/B et S par a. Ce n'est possible pour aucun autre état.

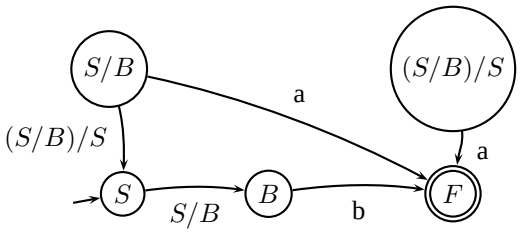


FIG. 8 – Un autre AR reconnaissant $a^n b^n$

Les ARs produisent les mêmes structures que les GCs unidirectionnelles, c'est-à-dire ne faisant appel qu'au schéma FA. On espère donc, comme dans l'Exemple 8, les apprendre à partir de peignes uniquement. Mais l'AR de la Figure 8 n'appartient à aucun espace de recherche qui partirait d'un ensemble de peignes, parce qu'il comprend des états qui ne sont pas accessibles à partir de l'été initial. On devra donc contraindre un peu plus nos ARs (ou nos GCs unidirectionnelles) pour espérer les apprendre ainsi.

4.2 Apprentissage de GCs unidirectionnelles à partir de peignes

Quand les données de départ sont des chaînes uniquement, la stratégie d'apprentissage employée par Kanazawa (Kanazawa, 1998) consiste à générer toutes les structures possibles compatibles avec ces chaînes, avant de lancer l'algorithme d'inférence à partir d'exemples structurés décrit en section 3.3. Nous proposons une nouvelle stratégie bien moins coûteuse, adaptée à de nouvelles sous-classes de GCs.

Définition 12 (Nouvelles sous-classes de GCs)
Une GC G est dite sans catégorie inutile *si toute affectation d'une catégorie à un mot du vocabulaire dans G est utilisée au moins une fois dans l'analyse syntaxique d'un élément de $L(G)$ (Kanazawa, 1998)). Soit :*
$\mathcal{G}_k^{FA} = \{\sigma(G) | G \in \mathcal{G}_k \cap \mathcal{G}_r$ *et G est sans catégorie inutile et σ est une substitution unificatrice pour $G\}$.*

Bien sûr, pour tout $k \geq 1$, $\mathcal{G}_k^{FA} \subset \mathcal{G}_k \cap \mathcal{G}_/$. De plus, $\bigcup_{k\geq 1}\{L(G) | G \in \mathcal{G}_k^{FA}\}$ contient tous les langages réguliers (cf. commentaires du Théorème 2, sachant que $\sigma = Id$ peut être considérée comme un cas particulier de substitution unificatrice) et certains langages algébriques (voir Example 8). Mais elle ne semble pas contenir tous les langages algébriques et nous ne savons pas encore caractériser précisément son expressivité. Le problème est qu'un état inutile parce que non accessible dans un AF (correspondant à une catégorie inutile dans une GC) peut devenir utile bien que toujours inaccessible après qu'une substitution ait été appliquée à l'AF pour le transformer en un AR. Le théorème suivant explicite la propriété fondamentale des éléments de $\mathcal{G}_k^{FA}$.

Théorème 5
Pour tout $k \geq 1$ et toute GC $G \in \mathcal{G}_k^{FA}$, il existe un ensemble fini $D \subset FL(G)$ constitué uniquement de peignes avec noeuds internes FA et il existe τ, une substitution unificatrice pour $FG(D)$, tels que $G = \tau(FG(D))$.

Preuve 7 (Preuve du Théorème 5)
Pour tout $G \in \mathcal{G}_k^{FA}$, par définition, $\exists G' \in \mathcal{G}_k \cap \mathcal{G}_r$ sans catégorie inutile et $\exists \sigma$ une substitution unificatrice pour G' telles que $G = \sigma(G')$. Nous savons, d'après le Théorème 2 que $\mathcal{G}_k \cap \mathcal{G}_r$ est apprenable par exemples structurés. Soit D un ensemble caractéristique d'exemples structurés pour G' (voir (Kanazawa, 1998)). G' est régulière, donc D n'est constitué que de peignes avec noeuds internes FA. G' est sans catégorie inutile, donc elle appartient au résultat de l'algorithme BP appliqué aux données k et D (Kanazawa, 1998). Cela signifie qu'il existe une substitution unificatrice ρ pour $FG(D)$ telle que $G' = \rho(FG(D))$. On a donc $G = \sigma(G') = \sigma(\rho(FG(D))$. On prend donc $\tau = \sigma \circ \rho$.□

Le Théorème 5 signifie que les membres de $\mathcal{G}_k^{FA}$ ont un ensemble caractéristique qui n'est constitué que de peignes avec noeuds internes FA, et qu'ils appartiennent donc au résultat de l'algorithme BP appliqué à cet ensemble. Cela suggère un nouvel algorithme pour apprendre la classe $\mathcal{G}_k^{FA}$ à partir de chaînes (voir l'Algorithme 1), qui n'a pas besoin de générer toutes les structures possibles associées à ces chaînes.

Algorithm 1 algorithme qui infère des GCs dans $\mathcal{G}_k^{FA}$ qui reconnaissent $\langle s_0, ..., s_i \rangle$

Require: $\langle s_0, ..., s_i \rangle$ où $\forall i$, $s_i \in \Sigma^+$ et k
1: $j \longleftarrow 0$
2: **repeat**
3: $C_j \longleftarrow \{s_0, ..., s_j\}$ \\ essayer C_j comme ensemble caractéristique
4: associer une structure de peignes avec noeuds FA à tous les éléments de C_j
5: appliquer l'algorithme BP à cet ensemble pour obtenir l'ensemble $R_{j,k} \subset \mathcal{G}_k$ des GCs sans catégorie inutile compatibles avec lui
6: supprimer les éléments de $R_{j,k}$ dont le langage de chaînes ne contient pas $\{s_{j+1}, ..., s_i\}$
7: $j \longleftarrow j + 1$
8: **until** $(j = i + 1)$ OR $(R_{j,k} \neq \emptyset)$
Ensure: $R_{j,k}$: un ensemble de GCs de $\mathcal{G}_k^{FA}$ qui reconnaissent $\langle s_0, ..., s_i \rangle$

Les GCs de $\mathcal{G}_k^{FA}$ sont au plus k-valuées. Mais la valeur de k requise par l'Algorithme 1 peut être plus grande que celle requise par l'algorithme BP. Les GCs de $\mathcal{G}_k^{FA}$ sont en quelque sorte sous une certaine forme normale : elles ne produisent que des peignes ou des adjonctions de peignes. Le Théorème 5 assure que, pour toute grammaire $G \in \mathcal{G}_k^{FA}$ produisant $L(G) = \langle s_i \rangle_{i \in \mathbb{N}}$, l'Algorithme 1 contient, à la limite, G parmi ses résultats.

5 Conclusion

Cette étude est un premier pas pour intégrer dans un même cadre les travaux réalisés en inférence grammaticale de langages réguliers et ceux réalisés en apprentissage de grammaires catégorielles. Les premiers bénéfices de ce rapprochement sont l'obtention, sans beaucoup d'efforts, de résultats originaux, et surtout une meilleure compréhension de la nature des opérations de généralisation utilisées dans chacune des deux approches. Une autre conséquence, sans doute moins attendue, de ce travail, est la définition d'une

nouvelle classe d'automates qui étend naturellement celle des automates finis tout en entretenant des liens forts avec les grammaires catégorielles unidirectionnelles.

Ce rapprochement montre surtout que le domaine de l'inférence grammaticale de langages algébriques n'est peut-être pas si différente qu'on le pensait de l'inférence de langages réguliers. Dans les langages réguliers, les structures se déduisent directement des chaînes. Quand, en revanche, il s'agit d'apprendre un langage algébrique à partir de chaînes, les structures possibles sont sous-déterminées. D'où l'idée, finalement assez naturelle, de chercher une forme normale pour représenter les langages algébriques qui contraigne le plus possible les structures sous-jacentes. Cette première approche demande bien sûr à être complétée et approfondie, notamment pour voir si elle recoupe ou non des travaux déjà effectués en inférence de grammaires algébriques.

Références

ANGLUIN D. (1982). Inference of reversible languages. *J. ACM*, **29**(3), 741–765.

BAR-HILLEL Y., GAIFMAN C. & SHAMIR E. (1960). On categorial and phrase structure grammars. *Bulletin of the Research Council of Israel*, **9F**.

BESOMBES J. & MARION J.-Y. (2004). Learning reversible categorial grammars from structures. In *Categorial Gramars*.

BUSZKOWSKI W. & PENN G. (1990). Categorial grammars determined from linguistic data by unification. *Studia Logica*, **49**, 431–454.

DENIS F., LEMAY A. & TERLUTTE A. (2002). Some language classes identifiable in the limit from positive data. In *ICGI 2002*, number 2484 in Lecture Notes in Artificial Intelligence, p. 63–76 : Springer Verlag.

DUPONT P., MICLET L. & VIDAL E. (1994). What is the search space of the regular inference. In L. N. IN ARTIFICIAL INTELLIGENCE, Ed., *ICGI'94 - Lectures Notes in Computer Science*, volume 862 - Grammatical Inference and Applications, p. 25–37, Heidelberg.

FLORÊNCIO C. C. (2002). Consistent identification in the limit of rigid grammars from strings is np-hard. In M. V. Z. P. ADRIAANS, H. FERNAU, Ed., *Grammatical Inference : Algorithms and Applications*, volume 2484 of *Lecture Notes in Artificial Intelligence*, p. 49–62 : Springer Verlag.

FLORÊNCIO C. C. (2001). Consistent identification in the limit of any of the classes k-valued is NP-hard. In *Logical Aspects of Computational Linguistics*, volume 2099 of *Lecture Notes in Artificial Intelligence*, p. 125–134 : Springer Verlag.

GOLD E. (1967). Language identification in the limit. *Inform. Control*, **10**, 447–474.

JOSHI A. & SCHABES Y. (1997). *Handbook of Formal Languages, vol3*, chapter Tree-Adjoining Grammars, p. 69–120. Springer Verlag.

KANAZAWA M. (1998). *Learnable Classes of Categorial Grammars*. The European Association for Logic, Language and Information. CLSI Publications.

ONCINA J. & GARCIA P. (1992). Inferring regular languages in polynomial update time. In *Pattern Recognition and Image Analysis*, p. 49–61.

SAKAKIBARA Y. (1992). Efficient learning of context-free grammars from positive structural examples. *Information and Computation*, **97**(1), 23–60.

WOODS W. (1970). Transition network grammars for natural language analysis. *Commun. ACM*, **10**, 591–606.

Séparateurs à Vaste Marge Optimisant la Fonction F_β

Jérôme Callut and Pierre Dupont

Department of Computing Science and Engineering, INGI
Université catholique de Louvain,
Place Sainte-Barbe 2
B-1348 Louvain-la-Neuve, Belgium
{jcal,pdupont}@info.ucl.ac.be

Abstract : Dans cet article, nous introduisons une nouvelle paramétrisation des Séparateurs à Vaste Marge (SVM) appelée F_β SVM. Cette dernière permet d'effectuer un apprentissage basé sur l'optimisation de la fonction F_β au lieu de l'erreur de classification habituelle. Les expériences montrent les avantages d'une telle démarche par rapport à la formulation *soft-margin* standard (avec les écarts à la marge au carré) lorsque l'on accorde une importance différente à la précision et au rappel. Une procédure automatique basée sur le score F_β de généralisation est ensuite introduite pour sélectionner les paramètres du modèle. Cette procédure repose sur les résultats de Chapelle, Vapnik et al. (Chapelle *et al.*, 2002) concernant l'utilisation de méthodes basées sur le gradient dans le cadre de la sélection de modèles. Les dérivées de la fonction de perte F_β par rapport à la constante de régularisation C et à la largeur σ d'un noyau gaussien sont définies formellement. A partir de là, les paramètres du modèle sont sélectionnés en effectuant une descente de gradient de la fonction de perte F_β dans l'espace des paramètres. Les expériences sur des données réelles montrent les bénéfices de cette approche lorsque l'on cherche à optimiser le critère F_β.

1 Introduction

Support Vector Machines (SVM) introduced by Vapnik (Vapnik, 1995) have been widely used in the field of pattern recognition for the last decade. The popularity of the method relies on its strong theoretical foundations as well as on its practical results. Performance of classifiers is usually assessed by means of classification error rate or by Information Retrieval (IR) measures such as precision, recall, F_β, breakeven-point and ROC curves. Unfortunately, there is no direct connection between these IR criteria and the SVM hyperparameters: the regularization constant C and the kernel parameters. In this paper, we propose a novel method allowing the user to specify his requirement in terms of the F_β criterion. First of all, the F_β measure is reviewed as a user specification criterion in section 2. A new SVM parametrization dealing with the β parameter is introduced in section 3. Afterwards, a procedure for automatic model selection according

to F_β is proposed in section 4. This procedure is a gradient-based technique derived from the results of Chapelle, Vapnik et al. (Chapelle *et al.*, 2002). Finally, experiments with artifical and real-life data are presented in section 5.

2 User specifications with the F_β criterion

Precision and recall are popular measures to assess classifiers performance in an information retrieval context (Sebastiani, 2002). Therefore, it would be convenient to use these evaluation criteria when formulating the user specifications. For instance, let us consider the design of a classifier used to retrieve documents according to topic. Some users prefer to receive a limited list of relevant documents even if this means losing some interesting ones. Others would not want to miss any relevant document at the cost of also receiving non-relevant ones. Those specifications correspond respectively to a high precision and a high recall.

The two previous measures can be combined in a unique F_β measure in which the paramater β specifies the relative importance of recall with respect to precision. Setting β equals to 0 would only consider precision whereas taking $\beta = \infty$ would only take recall into account. Moreover, precision and recall are of equal importance when using the F_1 measure. The contingency matrix and estimations of precision, recall and F_β are given hereafter.

	Target: +1	**Target: -1**
+1	True Pos. ($\#TP$)	False Pos. ($\#FP$)
-1	False Neg. ($\#FN$)	True Neg. ($\#TN$)

Precision π	$\frac{\#TP}{\#TP+\#FP}$
Recall ρ	$\frac{\#TP}{\#TP+\#FN}$
F_β	$\frac{(\beta^2+1)\pi\rho}{\beta^2\pi+\rho}$

3 F_β Support Vector Machines

In this section, we introduce a new parametrization of SVM allowing to formulate user specifications in terms of the F_β criterion. To do so, we establish a relation between the contingency matrix and the slack variables used in the soft-margin SVM setting. Based on this link, we devise a new optimization problem which maximizes an approximation of the F_β criterion regularized by the size of the margin.

3.1 Link between the contingency matrix and the slacks

Let us consider a binary classification task with a training set $Tr = \{(\boldsymbol{x_1}, y_1), \ldots, (\boldsymbol{x_n}, y_n)\}$ where $\boldsymbol{x_i}$ is an instance in some input space $\mathcal{X}$ and $y_i \in \{-1, +1\}$ represents its category. Let n^+ and n^- denote respectively the number of positive and negative examples. The soft-margin formulation of SVM allows examples to be missclassified or to lie inside the margin by the introduction of slack variables ξ in the problem constraints:

OP1 Minimize $W(\boldsymbol{w}, b, \boldsymbol{\xi}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C.\Phi(\boldsymbol{\xi})$

s.t. $$\begin{cases} y_i(\langle \boldsymbol{w}, \boldsymbol{x_i} \rangle + b) \geq 1 - \xi_i & \forall\, i = 1..n \\ \xi_i \geq 0 & \forall\, i = 1..n \end{cases}$$

where $\boldsymbol{w}$ and b are the parameters of the hyperplane.

The $\Phi(.)$ term introduced in the objective function is used to penalize solutions presenting many training errors. For any feasible solution $(\boldsymbol{w}, b, \boldsymbol{\xi})$, missclassified training examples have an associated slack value of at least 1. The situation is illustrated in figure 1. Hence, it seems natural to chose a function counting the number of slacks greater or equal to 1 as penalization function $\Phi(.)$. Unfortunately, the optimization of such a function combined with the margin criterion turns out to be a mixed-integer problem known to be NP-hard (Schölkopf & Smola, 2002). In fact, two approximations of the counting function are commonly used: $\Phi(\boldsymbol{\xi}) = \sum_{i=1}^{n} \xi_i$ (1-norm) and $\Phi(\boldsymbol{\xi}) = \sum_{i=1}^{n} \xi_i^2$ (2-norm). These approximations present two peculiarities: 1) The sum of slacks related to examples inside the margin might be considered as errors. 2) Examples with a slack value greater than 1 might contribute as more than one error. However, the use of these approximations is computationally attractive as the problem remains convex, quadratic and consequently solvable in polynomial time. In the sequel, we will focus on the 2-norm alternative.

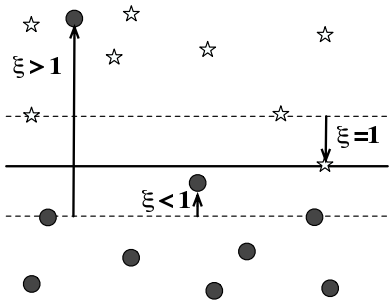


Figure 1: Soft-margin SVM and associated slacks

The computation of the preceding approximations separately for different class labels allows to bound the elements of the contingency matrix.

Proposition 1
Let ($\boldsymbol{w}$,b,$\boldsymbol{\xi}$) be a solution satisfying the constraints of OP1. The following bounds holds for the elements of the contingency matrix computed on the training set:

$$\bullet\ \#TP \geq n^+ - \sum_{\{i|y_i=+1\}} \xi_i^2 \qquad \bullet\ \#FP \leq \sum_{\{i|y_i=-1\}} \xi_i^2$$
$$\bullet\ \#FN \leq \sum_{\{i|y_i=+1\}} \xi_i^2 \qquad \bullet\ \#TN \geq n^- - \sum_{\{i|y_i=-1\}} \xi_i^2$$

These bounds will be called the slack estimates of the contingency matrix. It should be noted that they also could have been formulated using the 1-norm approximation.

3.2 The F_β parametrization

Let us introduce a parametrization of SVM in which a regularized F_β criterion is optimized. The F_β function can be expanded using the definition of precision and recall as:

$$F_\beta = \frac{(\beta^2+1)\pi\rho}{\beta^2\pi+\rho} = \frac{(\beta^2+1)\#TP}{(\beta^2+1)\#TP + \beta^2\#FN + \#FP}$$

The optimal value for F_β (≤ 1) is obtained by minimizing $\beta^2\#FN + \#FP$. Replacing $\#FN$ and $\#FP$ by their slack estimates and integrating this into the objective function leads to the following optimization problem:

OP2 Minimize $\quad W(\boldsymbol{w}, b, \boldsymbol{\xi}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C.[\beta^2. \sum_{\{i|y_i=+1\}} \xi_i^2 + \sum_{\{i|y_i=-1\}} \xi_i^2]$

s.t. $\quad \begin{cases} y_i(\langle \boldsymbol{w}, \boldsymbol{x_i} \rangle + b) \geq 1 - \xi_i & \forall\ i = 1..n \\ \xi_i \geq 0 & \forall\ i = 1..n \end{cases}$

The relative importance of the F_β criterion with respect to the margin can be tuned using the regularization constant C. Since the slack estimates for $\#FP$ and $\#FN$ are upper bounds, OP2 is based on a pessimistic estimation of the F_β. OP2 can be seen as an instance of the SVM parametrization considering two kinds of slacks with the associated regularization constants C^+ and C^- (Nello Critianini, 2002). In our case, the regularization constants derive from the β value, i.e. $C^+ = C\beta^2$ and $C^- = C$. It should be pointed out that when $\beta = 1$, OP2 is equivalent to the traditional 2-norm soft-margin SVM problem.

The optimization of the F_β criterion is closely related to the problem of training a SVM with an imbalanced dataset. When the prior of a class is by far larger than the prior of the other class, the classifier obtained by a standard SVM training is likely to act as the trivial acceptor/rejector (i.e. a classifier always predicting $+1$, respectively -1). To avoid this inconvenience, some authors (Veropoulos *et al.*, 1999) have introduced different penalities for the different classes using C^+ and C^-. This method has been applied in order to control the sensitivity[1] of the model. However, no automatic procedure has been proposed to choose the regularization constants with respect to the

[1]The sensitivity is the rate of true positive examples and is equivalent to recall.

user specifications. Recently, this technique has been improved by artificially oversampling the minority class (Akbani *et al.*, 2004). Other authors (Amerio *et al.*, 2004) have proposed to select a unique regularization constant C through a bootstrap procedure. This constant is then used as a starting point for tuning C^+ and C^- on a validation set.

4 Model selection according to F_β

In the preceding section, we proposed a parametrization of SVM enabling the user to formulate his specifications with the β parameter. In addition, the remaining hyperparameters, i.e. the regularization constant and the kernels parameters, must be selected. In the case of SVM, model selection can be made using the statistical properties of the optimal hyperplane, thus avoiding the need of performing cross-validation. Indeed, several bounds of the leave-one-out (loo) error rate can be directly derived from the parameters of the optimal hyperplane expressed in dual form (Vapnik & Chapelle, 2000; Schölkopf *et al.*, 1999; Joachims, 2000). A practical evaluation of several of these bounds has been recently proposed in (Duan *et al.*, 2003). Moreover, Chapelle, Vapnik et al. (Chapelle *et al.*, 2002) have shown that the hyperplane dual parameters are differentiable with respect to the hyperparameters. This allows the use of gradient-based techniques for model selection (Chapelle *et al.*, 2002; Chung *et al.*, 2003). In this section, we propose a gradient-based algorithm selecting automatically C and the width σ of a gaussian kernel[2] according to the generalization F_β score.

4.1 The generalization F_β loss function

It has been proved by Vapnik (Vapnik, 1998) that for an example $(\boldsymbol{x_i}, y_i)$ producing a loo error, $4\alpha_i R^2 \geq 1$ holds, where R is the radius of the smallest sphere enclosing all the training examples and α_i is the i-th dual parameter of the optimal hyperplane. This inequality was originally formulated for the hard-margin case. However, it can be applied to the 2-norm soft-margin SVM as the latter can be seen as a hard margin problem with a transformed kernel (Cortes & Vapnik, 1995; Nello Critianini, 2002). Using the preceding inequality, one can build an estimator of the generalization F_β score of a given model. Alternately, it is possible to formulate a loss function following the reasoning developed in section 3.2:

$$L_{F_\beta}(\boldsymbol{\alpha}, R) \triangleq 4R^2 \left(\beta^2 \sum_{\{i|y_i=+1\}} \alpha_i + \sum_{\{i|y_i=-1\}} \alpha_i \right)$$

4.2 The model selection algorithm

We introduce here an algorithm performing automatic model selection according to the F_β criterion. It selects the model by performing a gradient descent of the F_β loss

[2] $k(\boldsymbol{x_i}, \boldsymbol{x_j}) = exp(-\|\boldsymbol{x_i} - \boldsymbol{x_j}\|^2/2\sigma^2)$

function over the set of hyperparameters. For the sake of clarity, C and σ, are gathered in a single vector $\boldsymbol{\theta}$. The model selection algorithm is sketched hereafter.

Algorithm F_β MODELSELECTION
Input: Training set $Tr = (\boldsymbol{x_1}, y_1), \ldots, (\boldsymbol{x_n}, y_n)$
Initial values for the hyperparameters $\boldsymbol{\theta^0}$
Precision parameter ϵ
Output: Optimal hyperparameters $\boldsymbol{\theta^*}$
SVM optimal solution $\boldsymbol{\alpha^*}$ using $\boldsymbol{\theta^*}$

$\boldsymbol{\alpha}^0 \leftarrow \texttt{trainF}_\beta\texttt{SVM}(Tr, \boldsymbol{\theta^0})$;
$(R, \boldsymbol{\lambda})^0 \leftarrow \texttt{smallestSphereRadius}(Tr, \boldsymbol{\theta^0})$;

repeat

$\boldsymbol{\theta}^{t+1} \leftarrow \texttt{updateHyperparameters}(\boldsymbol{\theta}^t, \boldsymbol{\alpha}^t, R^t, \boldsymbol{\lambda}^t)$;
$\boldsymbol{\alpha}^{t+1} \leftarrow \texttt{trainF}_\beta\texttt{SVM}(Tr, \boldsymbol{\theta^{t+1}})$;
$(R, \boldsymbol{\lambda})^{t+1} \leftarrow \texttt{smallestSphereRadius}(Tr, \boldsymbol{\theta^{t+1}})$;
$t \leftarrow t + 1$;

until $|L_{F_\beta}(\boldsymbol{\alpha}^t, R^t) - L_{F_\beta}(\boldsymbol{\alpha}^{t-1}, R^{t-1})| < \epsilon$;
return $\{\boldsymbol{\theta^t}, \boldsymbol{\alpha^t}\}$

The $\texttt{trainF}_\beta\texttt{SVM}$ function solves OP3, the dual problem of OP2, which has the same form as the dual hard-margin problem (Schölkopf & Smola, 2002):

OP3 Maximize $W(\boldsymbol{\alpha}) = -\frac{1}{2}\sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j k'(\boldsymbol{x_i}, \boldsymbol{x_j}) + \sum_{i=1}^{n} \alpha_i$

s.t. $\begin{cases} \sum_{i=1}^{n} \alpha_i y_i = 0 \\ \alpha_i \geq 0 \quad \forall\, i = 1..n \end{cases}$

with a transformed kernel:

$$k'(\boldsymbol{x_i}, \boldsymbol{x_j}) = \begin{cases} k(\boldsymbol{x_i}, \boldsymbol{x_j}) + \delta_{ij} \cdot \frac{1}{C\beta^2} & \text{if } y_i = +1 \\ k(\boldsymbol{x_i}, \boldsymbol{x_j}) + \delta_{ij} \cdot \frac{1}{C} & \text{if } y_i = -1 \end{cases}$$

where δ_{ij} is the Kronecker delta and $k(.,.)$ is the original kernel function.

The radius of the smallest sphere enclosing all the examples computed by the `smallestSphereRadius` function is obtained by taking the square root of the objective function optimal value in the following optimization problem (Schölkopf & Smola, 2002):

OP4 Maximize $W(\boldsymbol{\lambda}) = \sum_{i=1}^{n} \lambda_i k'(\boldsymbol{x_i}, \boldsymbol{x_i}) - \sum_{i,j=1}^{n} \lambda_i \lambda_j k'(\boldsymbol{x_i}, \boldsymbol{x_j})$

s.t. $\begin{cases} \sum_{i=1}^{n} \lambda_i = 1 \\ \lambda_i \geq 0 \quad \forall\, i = 1..n \end{cases}$

The optimization problems OP3 and OP4 can be solved in polynomial time in n, *e.g.* using an interior point method (Vanderbei, 1994). Furthermore, the solution to OP3, respectively OP4, at a given iteration can be used as a good starting point for the next iteration.

At each iteration, the hyperparameters can be updated by means of a gradient step : $\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t - \eta . \partial L_{F_\beta} / \partial \boldsymbol{\theta}$ where $\eta > 0$ is the updating rate. However, second order methods often provide a faster convergence, which is valuable since two optimization problems have to be solved at each iteration. For this reason, the `updateHyperparameters` function relies on the BFGS algorithm (Fletcher & Powell, 1963), a quasi-Newton optimization technique. The time complexity of the `updateHyperparameters` function is $\mathcal{O}(n^3)$ since it is dominated by the inversion of a possibly $n \times n$ matrix (see section 4.3). The derivatives of the F_β loss function with respect to the hyperparameters are detailed in the next section. The algorithm is iterated until the F_β loss function no longer changes by more than ϵ.

4.3 Derivatives of the F_β loss function

The derivatives of the transformed kernel function with respect to the hyperparameters are given by:

$$\frac{\partial k'(\boldsymbol{x_i}, \boldsymbol{x_j})}{\partial C} = \begin{cases} -1/(C^2\beta^2) & \text{if } i = j \text{ and } y_i = +1 \\ -1/C^2 & \text{if } i = j \text{ and } y_i = -1 \\ 0 & \text{otherwise} \end{cases}$$

$$\frac{\partial k'(\boldsymbol{x_i}, \boldsymbol{x_j})}{\partial \sigma^2} = k(\boldsymbol{x_i}, \boldsymbol{x_j}) \frac{\|\boldsymbol{x_i} - \boldsymbol{x_j}\|^2}{2\sigma^4}$$

The derivatives of the squared radius can then be obtained applying the lemma 2 of Chapelle, Vapnik et al. (Chapelle *et al.*, 2002):

$$\frac{\partial R^2}{\partial \theta} = \sum_{i=1}^{n} \lambda_i \frac{\partial k'(\boldsymbol{x_i}, \boldsymbol{x_i})}{\partial \theta} - \sum_{i,j=1}^{n} \lambda_i \lambda_j \frac{\partial k'(\boldsymbol{x_i}, \boldsymbol{x_j})}{\partial \theta}$$

where $\theta \in \{C, \sigma^2\}$. The derivation of the hyperplane dual parameters proposed in (Chapelle *et al.*, 2002) follows:

$$\frac{\partial(\boldsymbol{\alpha}, b)}{\partial \theta} = -H^{-1} \frac{\partial H}{\partial \theta} (\boldsymbol{\alpha}, b)^T, \quad H = \begin{pmatrix} \boldsymbol{y}^T K \boldsymbol{y} & \boldsymbol{y} \\ \boldsymbol{y}^T & 0 \end{pmatrix}$$

where K is the kernel matrix and $\boldsymbol{y}$ is the vector of examples labels. The H matrix is derived by using the preceding kernel function derivatives. It should be stressed that only examples corresponding to support vectors have to be considered in the above formula. Finally, the derivative of $L_{F_\beta}(.,.)$ with respect to a hyperparameter θ is given by:

$$\begin{aligned}\frac{\partial L_{F_\beta}(\boldsymbol{\alpha}, R)}{\partial \theta} &= 4\frac{\partial R^2}{\partial \theta}\left(\beta^2 \sum_{\{i|y_i=+1\}} \alpha_i + \sum_{\{i|y_i=-1\}} \alpha_i\right) \\ &+ 4R^2\left(\beta^2 \sum_{\{i|y_i=+1\}} \frac{\partial \alpha_i}{\partial \theta} + \sum_{\{i|y_i=-1\}} \frac{\partial \alpha_i}{\partial \theta}\right)\end{aligned}$$

5 Experiments

We performed several experiments to assess the performance of the F_β parametrization and the model selection algorithm. First, the F_β parametrization was tested with positive and negative data in $\mathbb{R}^{10}$ drawn from two largely overlapping normal distributions. The priors for positive and negative classes were respectively 0.3 and 0.7. It is usually more difficult to obtain a good recall when data are unbalanced in this way. Experiments were carried out using training sets of 600 examples, a fixed test set of 1,000 examples and a linear kernel. A comparison between the F_β parametrization and the 2-norm soft-margin SVM with $C = 1$ is displayed in figure 2. For each β considered, the training data were resampled 10 times in order to produce averaged results. In this setting, our parametrization obtained better F_β scores than the standard soft-margin SVM, especially when a high recall was requested. The second part of the figure 2 presents the evolution of precision, recall and the F_β score for different β values.

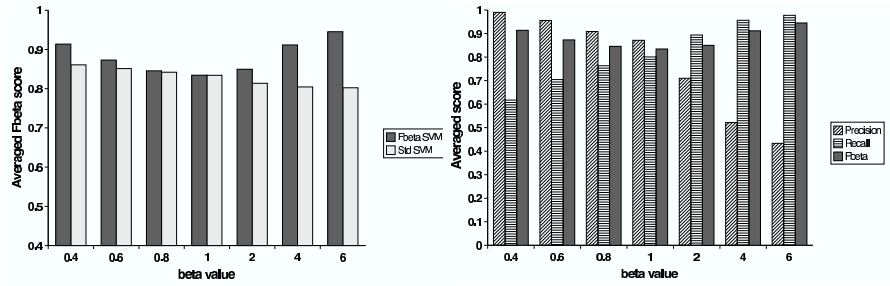


Figure 2: The F_β parametrization tested with artificially generated data. Left: comparison between the standard 2-norm soft-margin SVM and the F_β parametrization. Right: Evolution of precision, recall and of the F_β score accoring to different β values.

Afterwards, our parametrization was tested using several class priors. The experimental setup was unchanged except for the class priors while generating the training and test data. Figure 3 shows the evolution of the F_β score obtained by our parametrization and by the 2-norm soft-margin SVM using several class priors. For the standard 2-norm soft-margin SVM, one notes that the effect of the priors is particularly important when positive examples are few in numbers and that a high recall is requested. In this setting, our parametrization outperformed the standard 2-norm soft-margin SVM by more than 0.1.

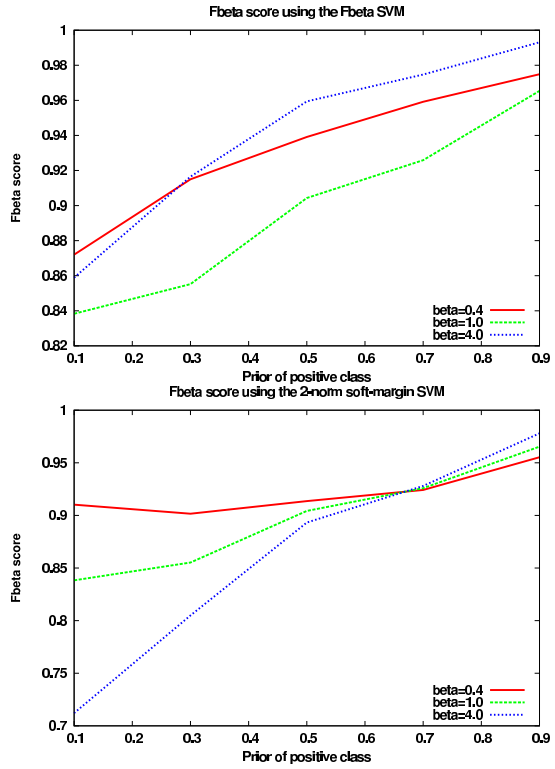


Figure 3: The F_β parametrization tested using artificially generated data with several class priors. Top: F_β scores obtained on the test set using the F_β parametrization. Bottom: F_β scores obtained on the test set using the standard 2-norm soft-margin SVM.

The model selection algorithm was first tested with data generated as in the previous paragraph. The hyperparameters C and σ were initialized to 1 and the precision parameter ϵ was set to 10^{-6}. Our objective was to investigate the relation between the minimization of the F_β loss function and the F_β score obtained on unknown test data. The figure 4 shows the evolution of the F_β loss function during the gradient descent, using $\beta = 2$. The associated precision, recall and F_β scores on test data are displayed in the bottom of the figure 4. Even if the optima of the F_β loss function and the F_β score do not match exactly, one can observe that good F_β scores were obtained when the F_β loss function is low. After 35 iterations, the classifier obtained a F_β score close to 0.9 with the hyperparameters $C = 4.33$ and $\sigma = 1.94$.

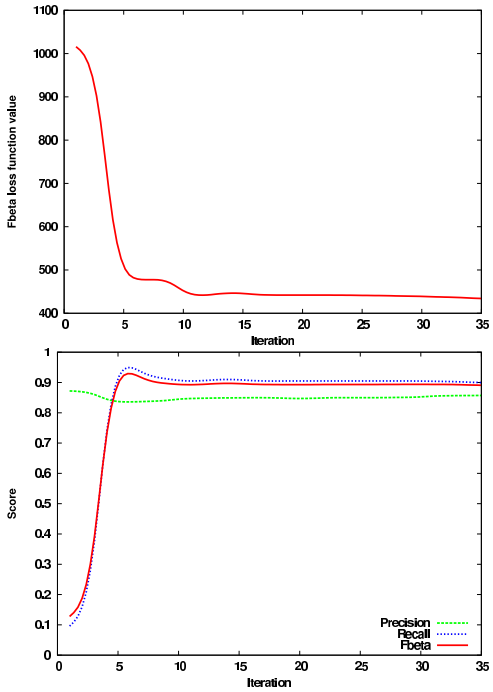


Figure 4: The F_β model selection algorithm tested with artificially generated data and with $\beta = 2$. Top: the evolution of the F_β loss function during the gradient descent. Bottom: the related values of precision, recall and F_β score on independent test data.

The model selection algorithm was then compared to the Radius-Margin (RM) based algorithm (Chapelle *et al.*, 2002) using the `Diabetes` dataset (Blake & Merz, 1998). This dataset contains 500 positive examples and 268 negative examples. It was randomly split into a training and a test set, each one containing 384 examples. In this setting, it is usually more difficult to obtain a classifier with a high precision. The same initial conditions as before were used. The RM based algorithm select the model parameters of the 2-norm soft-margin SVM according to the RM estimator of the gen-

eralization error rate. It should be pointed out that when $\beta = 1$, both methods are equivalent since the same function is optimized. The comparison is illustrated in the first part of the figure 5. As expected, our method provided better results when β moves far away from value 1. The influence of the β parameter on precision, recall and the F_β score can be observed in the second part of the figure 5.

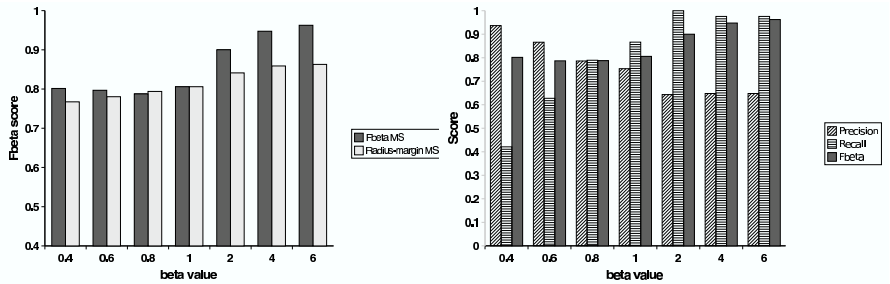


Figure 5: The F_β model selection algorithm tested with the `Diabetes` dataset. Left: Comparison between the F_β model selection algorithm and the radius-margin based method. Right: Evolution of precision, recall and of the F_β score accoring to different β values.

6 Conclusion

We introduced in this paper F_β SVMs, a new parametrization of support vector machines. It allows to formulate user specifications in terms of F_β, a classical IR measure. Experiments illustrates the benefits of this approach over a standard SVM when precision and recall are of unequal importance. Besides, we extended the results of Chapelle, Vapnik et al. (Chapelle *et al.*, 2002) based on the Radius-Margin (RM) bound in order to automatically select the model hyperparameters according to the generalization F_β score. We proposed an algorithm which performs a gradient descent of the F_β loss function over the set of hyperparameters. To do so, the partial derivatives of the F_β loss function with respect to these hyperparameters have been formally defined. Our experiments on real-life data show the advantages of this method compared to the RM based algorithm when the F_β evaluation criterion is considered.

Our future work includes improvements to the model selection algorithm in order to deal with larger training sets. Indeed, it is possible to use a sequential optimization method (Keerthi *et al.*, 2000) in the `smallestSphereRadius` function and chunking techniques (Joachims, 1998; Schölkopf & Smola, 2002) in the `trainF`$_\beta$`SVM` function. This typically allows to solve problems with more than 10^4 variables. Moreover, we believe that the inverse matrix H^{-1} can be computed incrementally during the chuncking iterations, using the Schur inversion formula for block matrices (Meyer, 2000).

Acknowledgment

This work is partially supported by the *Fonds pour la formation à la Recherche dans l'Industrie et dans l'Agriculture (F.R.I.A.)* under grant reference F3/5/5-MCF/FC-19271.

References

AKBANI R., KWEK S. & JAPKOWICZ N. (2004). Applying support vector machines to imbalanced datasets. In *Proceedings of the 15th European Conference on Machine Learning (ECML)*, p. 39–50, Pisa, Italy.

AMERIO S., ANGUITA D., LAZZIZZERA I., RIDELLA S., RIVIECCIO F. & ZUNINO R. (2004). Model selection in top quark tagging with a support vector classifier. In *Proceedings of International Joint Conference on Neural Networks (IJCNN 2004)*, Budapest, Hungary.

BLAKE C. & MERZ C. (1998). UCI repository of machine learning databases.

CHAPELLE O., VAPNIK V., BOUSQUET O. & MUKHERJEE S. (2002). Choosing multiple parameters for support vector machines. *Machine Learning*, **46**(1-3), 131–159.

CHUNG K.-M., KAO W.-C., SUN C.-L., WANG L.-L. & LIN C.-J. (2003). Radius margin bounds for support vector machines with the rbf kernel. *Neural Comput.*, **15**(11), 2643–2681.

CORTES C. & VAPNIK V. (1995). Support-vector networks. *Machine Learning*, **20**(3), 273–297.

DUAN K., KEERTHI S. S. & POO A. N. (2003). Evaluation of simple performance measures for tuning svm hyperparameters. *Neurocomputing*, **51**, 41–59.

FLETCHER R. & POWELL M. J. D. (1963). A rapidly convergent descent method for minimization. *Computer Journal*, **6**, 163–168.

JOACHIMS T. (1998). Making large-scale support vector machine learning practical. In A. S. B. SCHOLKOPF, C. BURGES, Ed., *Advances in Kernel Methods: Support Vector Machines*. MIT Press, Cambridge, MA.

JOACHIMS T. (2000). Estimating the generalization performance of a SVM efficiently. In P. LANGLEY, Ed., *Proceedings of ICML-00, 17th International Conference on Machine Learning*, p. 431–438, Stanford, US: Morgan Kaufmann Publishers, San Francisco, US.

KEERTHI S. S., SHEVADE S. K., BHATTACHARYYA C. & MURTHY K. R. K. (2000). A fast iterative nearest point algorithm for support vector machine classifier design. *IEEE-NN*, **11**(1), 124.

MEYER C. D. (2000). *Matrix analysis and applied linear algebra*. Society for Industrial and Applied Mathematics.

NELLO CRITIANINI J. S.-T. (2002). *An Introduction to Support Vector Machines*. The Press Syndicate of the University of Cambridge.

SCHÖLKOPF B., SHAWE-TAYLOR J., SMOLA A. J. & WILLIAMSON R. C. (1999). *Generalization Bounds via Eigenvalues of the Gram Matrix*. Rapport interne, Australian National University. Submitted to COLT99.

SCHÖLKOPF B. & SMOLA A. (2002). *Learning with Kernels*. Cambridge: MIT Press.

SEBASTIANI F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, **34**(1), 1–47.

VANDERBEI R. (1994). *LOQO: An Interior Point Code for Quadratic Programming*. Rapport interne SOR 94-15, Princeton University.

VAPNIK V. (1995). *The Nature of Statistical Learning Theory*. New York: Springer Verlag.

VAPNIK V. (1998). *Statistical Learning Theory*. New York: Wiley-Interscience.

VAPNIK V. & CHAPELLE O. (2000). Bounds on error expectation for support vector machines. *Neural Computation*, **12**(9).

VEROPOULOS K., CRISTIANINI N. & CAMPBELL C. (1999). Controlling the sensitivity of support vector machines. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI99)*, Stockholm, Sweden.

Kernel Basis Pursuit

Vincent Guigue, Alain Rakotomamonjy, Stéphane Canu[1]

Lab. Perception, Systèmes, Information - CNRS - FRE 2645
Avenue de l'Université, 76801 St Étienne du Rouvray
{Vincent.Guigue, Alain.Rakoto,
Stephane.Canu}@insa-rouen.fr

Résumé : Les méthodes à noyaux sont largement utilisées dans le domaine de la régression. Cependant, ce type de problème aboutit à deux questions récurrentes : comment optimiser le noyau et comment régler le compromis biais-variance ? L'utilisation de noyaux multiples et le calcul du chemin complet de régularisation permettent de faire face simplement et efficacement à ces deux tâches. L'introduction de noyaux multiples est également un moyen de fusionner des sources d'information hétérogènes.

Notre approche est inspirée de l'algorithme *Basis Pursuit* (Chen *et al.*, 1998). Nous avons suivi la méthode de Vincent et Bengio pour la non-linéarisation du *Basis Pursuit* (Vincent & Bengio, 2002).

Cet article présente une solution simple et parcimonieuse pour le problème de régression par méthode à noyaux multiples. Nous avons utilisé la formulation du LASSO (*Least Absolute Shrinkage and Selection Operator*) (Tibshirani, 1996), basée sur une régularisation $\mathcal{L}_1$, et l'algorithme du LARS (*Stepwise Least Angle Regression*) (Efron *et al.*, 2004) pour la résolution. La régularisation $\mathcal{L}_1$ est un gage de parcimonie tandis que le calcul du chemin complet de régularisation, via le LARS, permet de définir de nouveaux critères pour trouver le compromis biais-variance optimal. Nous présenterons également une heuristique pour le réglage des paramètres du noyau, afin de rendre la méthode complètement non paramétrique.

Mots-clés : Régression, Noyaux Multiples, LASSO, Méthode Non-Paramétrique

Abstract : Kernel methods have been widely used in the context of regression. But every problem leads to two major tasks: optimizing the kernel and setting the fitness-regularization compromise. Using multiple kernels and Basis Pursuit is a way to face easily and efficiently these two tasks. On top of that, it enables us to deal with multiple and heterogeneous sources of information.

Our approach is inspired by the Basis Pursuit algorithm (Chen *et al.*, 1998). We use Vincent and Bengio's method (Vincent & Bengio, 2002) to kernelize the Basis Pursuit and introduce the ability of mixing heterogeneous sources of information.

[1]This work was supported in part by the IST Program of the European Community, under the PASCAL Network of Excellence, IST-2002-506778. This publication only reflects the authors' views.

This article aims at presenting an easy, efficient and sparse solution to the multiple Kernel Basis Pursuit problem. We will use the Least Absolute Shrinkage and Selection Operator (LASSO) formulation (Tibshirani, 1996) ($\mathcal{L}_1$ regularization), and the Stepwise Least Angle Regression (LARS) algorithm (Efron *et al.*, 2004) as solver. The LARS provides a fast and sparse solution to the LASSO. The fact that it computes the optimal regularization path enables us to propose new auto-adaptive hyper-parameters for the fitness-regularization compromise. We will also propose some heuristics to choose the kernel parameters. Finally, we aim at proposing a parameter free, sparse and fast regression method.

Key words: Regression, Multiple Kernels, LASSO, Parameter Free.

1 Introduction

The context of our work is the following: we wish to estimate the functional dependency between an input x and an output y of a system given a set of examples $\{(x_i, y_i), x_i \in \mathcal{X}, y_i \in \mathcal{Y}, i = 1 \dots n\}$ which have been drawn i.i.d from an unknown probability law $P(X, Y)$. Thus, our aim is to recover the function f which minimizes the following risk

$$R[f] = \mathbb{E}\{(f(X) - Y)^2\} \tag{1}$$

but as $P(X, Y)$ is unknown, we have to look for the function f which minimizes the empirical risk :

$$R_{emp}[f] = \sum_{i=1}^{n} (f(x_i) - y_i)^2 \tag{2}$$

This problem is ill-posed and a classical way to turn it into a well-posed one is to use regularization theory (Tikhonov & Arsénin, 1977; Girosi *et al.*, 1995). In this context, the solution of the problem is the function $f \in \mathcal{H}$ that minimizes the regularized empirical risk :

$$R_{reg}[f] = \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \Omega(f) \tag{3}$$

where $\mathcal{H}$ is the hypothesis space, Ω is a functional which measures the smoothness of f and λ a regularization parameter (Wahba, 1990). Under general conditions on $\mathcal{H}$ (Reproducing Kernel Hilbert Space) (Kimeldorf & Wahba, 1971), the solution of this minimization problem is of the form:

$$f(x) = \sum_{i=1}^{n} \beta_i K(x_i, x) \tag{4}$$

where K is the reproducing kernel of $\mathcal{H}$.

The objective of this paper is two-fold: to propose a method to build a sparse kernel-based solution for this regression problem and to introduce new solutions for the bias-variance compromise problem. The question of the sparsity of the solution f can be addressed in two different ways. The first approach is to use a regularization term in equation 3 that imposes sparsity of β whereas the second one is based on stepwise

method consisting in adding functions from a dictionary. The bias-variance problem involves several parameters, especially the kernel parameters and the hyper-parameter trading between goodness-of-fit and regularization.

Our solution is based on ℓ_1 regularization, we use $\Omega = \|\beta\|_{\ell_1}$ in equation 3. This formulation is called the Least Absolute Shrinkage and Selection Operator (LASSO) (Tibshirani, 1996), it will enable us to improve sparsity. Our solver relies on the Stepwise Least Angle Regression (LARS) algorithm (Efron *et al.*, 2004), which is an iterative forward algorithm. Thus, the sparsity of the solution is closely linked to the efficiency of the method. We use Vincent and Bengio's strategy (Vincent & Bengio, 2002) to kernelize the resulting method. Finally, we end at the Kernel Basis Pursuit algorithm.

Associated with this learning problem, there are two major tasks to build a good regression function with kernel: optimizing the kernel and choosing a good compromise between fitness and regularization. The use of multiple kernels is a way to make the first task easier. We will use the optimal path regularization properties of the LARS to propose new heuristics, in order to set dynamically the fitness-regularization compromise.

In section 2, we will compare two approaches to the question of sparsity: the Matching Pursuit and the Basis Pursuit. We will explain the building and the use of the multiple kernels, combined with the LARS in section 3. Our results on synthetic and real data are presented in section 4. Section 5 gives our conclusions and perspectives on this work.

2 Basis vs Matching Pursuit

Two common strategies are available to face the problem of building a sparse regression function f. The first one relies on an iterative building of f. At each step k, the comparison between the target y and the function f_k leads to add a new source of information to build f_{k+1}. This approach is fast but it is greedy and thus sub-optimal. The second solution consists in solving a learning problem, by minimizing the regularized empirical risk of equation 3.

Mallat and Zhang introduced the Matching Pursuit algorithm (Mallat & Zhang, 1993): they proposed to construct a regression function f as a linear combination of elementary functions g picked from a finite redundant dictionary $\mathcal{D}$. This algorithm is iterative and one new function g is introduced at each step, associated with a weight β. At step k, we get the following approximation of f:

$$f_k = \sum_{i=1}^{k} \beta_i g_i \tag{5}$$

Given R_k, the residue generated by f_k, the function g_{k+1} and its associated weight β_{k+1} are selected according to:

$$(g_{k+1}, \alpha_{k+1}) = \text{argmin}_{g \in \mathcal{D}, \beta \in \mathbb{R}} \|R_k - \beta g\|^2 \tag{6}$$

The improvements described by Pati et al. (Orthogonal Matching Pursuit algorithm) (Pati *et al.*, 1993) keep the same framework, but optimize all the weights β_i at each

step. A third algorithm called pre-fitting (Vincent & Bengio, 2002) enables us to choose (g_{k+1}, β_{k+1}) according to R_{k+1}.

All those methods are iterative and greedy. The different variations improve the weights or the choice of the function g but the main characteristic remains unchanged. Matchin Pursuit does not allow to get rid of a previous source of information, which means that its solution is sub-optimal. The approach of Chen et al. (Chen *et al.*, 1998) is really different: they consider the whole dictionary of functions and look for the best linear solution (equation 5) to estimate y, namely, the solution which minimizes the regularized empirical risk. Using $\Omega = \|\beta\|_{\ell_1}$ leads to the LASSO formulation. Such a formulation requires costly and complex linear programming (Chen, 1995) or modified EM implementation (Grandvalet, 1998) to be solved. Finally it enables them to find an exact solution to the regularized learning problem.

The Stepwise Least Angle Regression (LARS) (Efron *et al.*, 2004) offers new opportunities, by combining an iterative and efficient approach with the exact solution of the LASSO. The fact that the LARS begins with an empty set of variables, combined with the sparsity of the solution explains the efficiency of the method. The ability of deleting dynamically useless variables enables the method to converge to the exact solution of the LASSO problem.

3 Learning with multiple kernels

3.1 Building a multiple kernel regression function

Vincent and Bengio (Vincent & Bengio, 2002) propose to treat the kernel K exactly in the same way as the matrix $\mathcal{X}$. Each column of K is then a source of information that can be added to the linear regression model f. Given an input vector x and a parametric mapping function Φ_θ defined by

$$\begin{array}{rccc} \Phi_\theta : & \mathbb{R}^d & & \mathcal{F} \\ & x & \rightarrow & \Phi_\theta(x) = K_\theta(x, \cdot) \end{array} \tag{7}$$

where $\mathcal{F}$ is the spanned feature space, we consider $K_\theta(x, .)$ as a source of information.

It becomes easy to deal with multiple mapping functions Φ_i. The multiple resulting kernels K_i are placed side by side in a big matrix K:

$$K = \left[\; K_1 \ldots K_i \ldots K_N \; \right] \tag{8}$$

N is the number of kernels. In this situation, each source of information $K_i(x_j, \cdot)$ is characterized by a point x_j of the learning set and a kernel parameter i. The number of information sources is then $s = nN$ and $K \in \mathbb{R}^{n \times s}$.

The learning problem becomes a variable selection problem where the β_i coefficients can be seen as the weights of the sources of information. We simplify the notations:

$$f = \sum_{i=1}^{N} \sum_{j=1}^{n} \beta_{ij} K_i(x_j, \cdot) = \sum_{i=1}^{s} \beta_i K(i, \cdot) = K\beta \tag{9}$$

It is important to note that no assumption is made on the kernel K_θ which can be non-positive. K can associate kernels of the same type (e.g. Gaussian) with different parameter values as well as different types of kernels (e.g. Gaussian and polynomial). The resulting matrix K is neither positive definite or square.

3.2 LARS

The LARS (Efron *et al.*, 2004) is a stepwise iterative algorithm which provides an exact to minimization of the regularized empirical risk (equation 3) with $\Omega = \|\beta\|_{\ell_1}$. We use the following formulation, which is equivalent to the LASSO:

$$\begin{array}{c} \min_\beta \|y - K\beta\|^2 \\ \text{With respect to: } \|\beta\|_{\ell_1} \leq t \end{array} \tag{10}$$

We denote by β_i the regression coefficient associated to the i^{th} source of information and by $\hat{y}^{(j)} = K\beta^{(j)}$ the regression function at step j. More generally, we will use exponent to characterize the iteration. LARS is made of the following main steps:

1. Initialization: the active set of information source $\mathcal{A}$ is empty, all β coefficients are set to zero.
2. Computation of the correlation between the sources of information and the residue. The residue R is defined by $R = y - \hat{y}$.
3. The most correlated source is added to the active set.

$$\mathcal{A} = \{\mathcal{A} \ \arg\max_{i,\theta}(|K_\theta^T(x_i, \cdot)R|)\} \tag{11}$$

4. Definition of the best direction in the active set: $\overrightarrow{u_\mathcal{A}}$ This is the most expensive part of the algorithm in time computation since it requires the inversion of the matrix $K_\mathcal{A}^T K_\mathcal{A}$.
5. The original part of the algorithm resides in the computation of the step γ. The idea is to compute γ such as two functions are equi-correlated with the residue (cf Fig. 1) whereas Ordinary Least Square (OLS) algorithm defines γ such as $\overrightarrow{u_\mathcal{A}}$ and $\overrightarrow{\hat{y}^{(j+1)}}, \overrightarrow{y}$ become orthogonal.
6. The regression function is updated:

$$\hat{y}^{(j+1)} = \hat{y}^{(j)} + \gamma\overrightarrow{u_\mathcal{A}} \tag{12}$$

It is necessary to introduce the ability of suppressing a function from the active set to fit the LASSO solution, namely to turn the forward algorithm into a stepwise method. When the sign of a β_i changes during the update (equation (12), the step γ is reduced so that this β_i becomes zero. Then, the corresponding source is removed from the active set and an optimization is performed over the new active set.

Solving the LASSO is really fast with this method, due to the fact that it is both forward and sparse. The first steps are not expensive, because of the small size of the

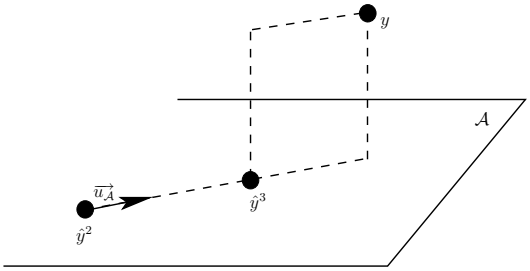


Figure 1: Computation of the step γ between $\hat{y}^{(2)}$ and $\hat{y}^{(3)}$. The plane denoted $\mathcal{A}$ represents the space spanned by the active set.

active set, then it becomes more and more time consuming with iterations. But the sparsity of ℓ_1 regularization limits the number of required iterations. LARS begins with an empty active set whereas linear programming and other backward methods begin with all functions and require to solve high dimensional linear system to put irrelevant coefficients to zero. Given the fact that only one point is added (or removed) during an iteration, it is possible to update the inverted matrix of step four instead of fully computing it. This leads to a simple-LARS algorithm, similarly to the simple-SVM formulation (Loosli *et al.*, 2004), which also increases the speed of the method.

3.3 Optimization of regularization parameter

One of the most interesting property of the LARS is the fact that it computes the whole regularization path. The regularization parameter λ of equation 3 is equivalent to the bound t of equation 10. At each step, the introduction of a new source of information leads to an optimal solution, corresponding to a given value of t. In the other classical algorithms, λ is set *a priori* and optimized by cross-validation. The LARS enables us to compute a set of optimal solutions corresponding to different values of t, with only one learning stage. It also enables us to optimize the value of t dynamically, during the learning stage.

Finding a good setting for t is very important: when t becomes too large, the resulting regression function is the same as the Ordinary Least Square (OLS) regression function. Hence, it requires the resolution of linear system of size $s \times s$. Early stopping should enable us to decrease the time computation (which is linked to the sparsity of the solution) as well as to improve the generalization of the learning (by regularizing).

3.3.1 Different compromise parameters

The computation of the complete regularization path offers the opportunity to set the compromise parameter dynamically (Bach *et al.*, 2004). The first step is to look for different expressions of the regularization parameter t of equation (10). The aim is to find the most meaningful one, namely the easiest way to set this parameter.

- The original formulation of the LARS relies on the compromise parameter t which is a bound on the sum of the absolute values of the β coefficients. t is difficult to set because it is somewhat meaningless.

- It is possible to apply Ljung criterion (Ljung, 1987) on the autocorrelation of the residue. The parameter is then a threshold which decides when the residue can be considered as white noise.

- Another solution consists in the study of the evolution of loss function $\ell(y_i, f_\theta^j(x_i))$ with regards to the step j. The criterion is a bound on the variation of this cost.

- ν-LARS. It is possible to define a criterion on the number of support vectors or on the rate of support vectors among the learning set. It is important to note that the ν threshold is then a hard threshold, whereas in the ν-SVM method where ν can be seen as an upper bound on the rate of support vectors (Schölkopf & Smola, 2002).

However, all these methods require the setting of a parameter *a priori*. The value of this parameter is estimated by cross-validation.

3.3.2 Trap source

We propose another method based on a trap parameter. The idea is to introduce one or many sources of information that we do not want to use. When the most correlated source with the residue belongs to the trap set, the learning procedure is stopped.

The trap sources of information can be built on different heuristics:

- according to the original signal noise when there exists prior knowledge on the data,

- with regards to the distribution of the learning points, to prevent overfitting (cf Fig. 2), in this case, a Gaussian kernel $K = K_{\sigma_{of}}$ is added to the information sources, with σ_{of} very small,

- by adding random variables among the sources of information (with Gaussian or uniform distribution). This kind of heuristic has already been used in variable selection (Bi *et al.*, 2003).

The use of a trap scale is closely linked to the way that LARS selects the sources of information. As seen in section 3.2, the selected source of information at a given iteration is the most correlated with the residue. Those heuristics are based on the meaning of the trap scale: the learning stage should be stopped when the residue is most correlated respectively with the noise, with only one source of information or with an independent random variable generated according to the uniform distribution. This means that no more relevant information is present in the sources that are not in the active set.

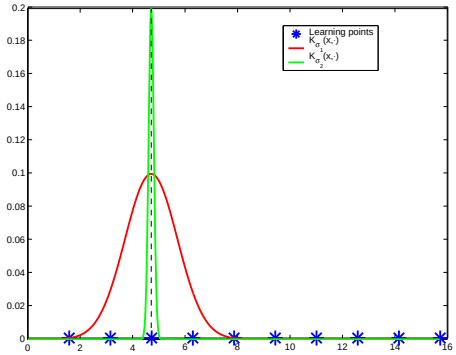


Figure 2: Illustration of a trap scale based on overfitting heuristic (Gaussian kernel). When $K_{\sigma_2}(x, \cdot)$ is the most correlated source of information with the residue, it means that the error is caused by only one point, it is a way to detect the beginning of overfitting.

3.4 Optimizing kernel parameters

Instead of searching an optimal parameter (or parameter vector) for the problem, we propose to find a key scale to fit the regions where there are the highest point density in the input space. We aim at finding a reference Gaussian parameter so that the two nearest points in the input space have few influence on each other. This reference parameter represents the smallest bandwidth which can be interesting for a given problem. Then, we propose to build a series of bigger Gaussian parameters from this scale to fit the different densities of points that can append in the whole input space.

A one nearest neighbors is performed on the training data. To describe high density regions, we focus on the shortest distances between neighbors. The key distance D_k is defined as the distance between x_i and x_j, the two nearest points in the input space.The corresponding key Gaussian parameter σ_k is defined so that:

$$K(x_i, x_j) = \frac{1}{\sqrt{2\pi}\sigma_k} \exp\left(-\frac{D_k^2}{2\sigma_k^2}\right) = 0.1 \tag{13}$$

That is to say, the bandwidth σ_k is designed so that a learning point has few influences on its neighbors in high density regions. For more robustness, it is recommended to use an improved definition of D_k. Given $\mathcal{S}$ the set of the one-nearest-neighbor distances. We define D_k as the mean of the 0.01 quantile of $\mathcal{S}$.

Then, a series of bandwidth is build as follow:

$$\sigma = \{\sigma_k, \sigma_k p, \sigma_k p^2, \sigma_k p^3, \sigma_k p^4, \sigma_k p^5\} \tag{14}$$

We choose to set the cardinal of σ to six, given the fact that experimental results are not improved beyond this value. Cross validations over synthetic data lead to set $p = 3.5$.

Another advantage of multiple kernels is that the LARS will optimize the scale for each point dynamically in the training stage. It offers the opportunity to adapt to the local density of points of the input space.

4 Experiments

We illustrate the efficiency of the methodology on synthetic and real data. Tables 1 and 2 present the results with two different algorithms: the SVM and the LARS. We use four strategies to stop the learning stage of the LARS.

- LARS- $\sum_i |\beta_i|$ is the classical method where a bound is defined on the sum of the regression coefficient. This bound is estimated by cross validation.
- ν-LARS is based on the fraction of support vectors. ν is also estimated by cross validation.
- LARS-RV relies on the introduction of random variables as sources of information. The learning stage is stopped when one of these sources is picked up as most correlated with the residue.
- LARS-σ_s relies also on a trap scale, but this scale is built according to the distribution the learning set. Selecting a source in this trap scale can be seen as overfitting. We use $\sigma_s = \sigma_k$ of equation (13).

To validate this approach, we compare the results with classical Gaussian ϵ-SVM regression, Parameters ϵ, C and σ are optimized by cross validation. In order to distinguish the benefit of the LARS from the benefits of the multiple kernel learning, we also give the results of LARS algorithm combined with single kernel.

4.1 Synthetic data

The learning of $\cos(\exp(\omega t))$ regression function, with random sampling show the multiple kernel interest. We try to identify:

$$f(t) = \cos(\exp(\omega t)) + b(t) \tag{15}$$

where $b(t)$ is a Gaussian white noise of variance $\sigma_b^2 = 0.4$. $t \in [0, 2]$ is drawn according to a uniform distribution, $\omega = 2.4$. We also tested the method over classical synthetic data described by Donoho and Johnstone (Donoho & Johnstone, 1994). For those signals, we took $t \in [0, 1]$, drawn according to a uniform distribution.

We use 200 points for the learning set and 1000 points for the testing set. The noise is added only on the learning set. Parameters (ν, $\sum_i |\beta_i|$...) are computed by cross validation on the learning set. Table 1 presents the results over 30 runs for each data base.

These results point out the sparsity and the efficiency of LARS solutions. Figure 3 illustrates how multiple kernel learning enables the regression function to fit the local frequency of the model. It also shows that selected points belong higher and higher scales with iterations. Indeed, the correlation with the residue can be seen as an energetic criterion: when the amplitude of the signal remain constant, there is more energy in the low frequency part of the signal. That is why the first selected sources of information describe those parts of the signal. The results with different Donoho's synthetic signals enable us to distinguish the benefits of the LARS method from the benefits of

Nb kernel	1			
Algorithm	ϵ - SVM	LARS- $\sum_i \lvert\beta_i\rvert$	ν-LARS	LARS-RV
$\cos(\exp(t))$	0.16 ± 0.016	0.16 ± 0.014	0.17 ± 0.015	0.17 ± 0.015
	155.4	130.3	120	141.3
	0	0	0	0
Doppler	0.045 ± 0.0062	0.041 ± 0.0068	0.043 ± 0.0060	0.039 ± 0.0059
	59.33	37.10	34	32.80
	0	0	0	0
Blocks	1.18 ± 0.20	1.03 ± 0.27	1.19 ± 0.17	1.07 ± 0.22
	45.30	25.02	21	27.35
	0	2	0	0
Ramp	$\mathbf{0.026 \pm 0.0055}$	0.028 ± 0.0070	0.029 ± 0.0060	0.028 ± 0.0065
	44.16	25.63	23	33.93
	22	4	0	3
HeaviSine	$\mathbf{0.48 \pm 0.12}$	0.48 ± 0.13	0.49 ± 0.12	0.51 ± 0.11
	51.43	37.25	40	42.11
	11	**11**	4	0

Nb kernel	6 (Multiple Kernels)			
Algorithm	LARS- $\sum_i \lvert\beta_i\rvert$	ν-LARS	LARS-RV	LARS-σ_s
$\cos(\exp(t))$	$\mathbf{0.13 \pm 0.014}$	$\mathbf{0.13 \pm 0.014}$	0.13 ± 0.016	0.14 ± 0.016
	122.4	120	121.4	127.8
	17	3	7	3
Doppler	0.033 ± 0.0060	0.035 ± 0.0062	$\mathbf{0.033 \pm 0.0059}$	0.035 ± 0.0075
	44.13	46	47.70	49.80
	12	0	**18**	0
Blocks	$\mathbf{0.95 \pm 0.34}$	0.97 ± 0.25	0.96 ± 0.25	0.99 ± 0.33
	34.02	38	42.35	35.87
	13	0	6	9
Ramp	0.031 ± 0.0080	0.032 ± 0.0050	0.031 ± 0.0058	0.033 ± 0.0059
	15.13	27	22.50	27.33
	1	0	0	0
HeaviSine	0.50 ± 0.14	0.51 ± 0.14	0.49 ± 0.15	0.50 ± 0.15
	51.30	49	48.22	49.03
	0	0	4	0

Table 1: Results of SVM and LARS for the $\cos(\exp(t))$ and Donoho's classical functions estimation. Mean and standard deviation of MSE on the test set (30 runs), number of support vectors used for each solution, number of best performances.

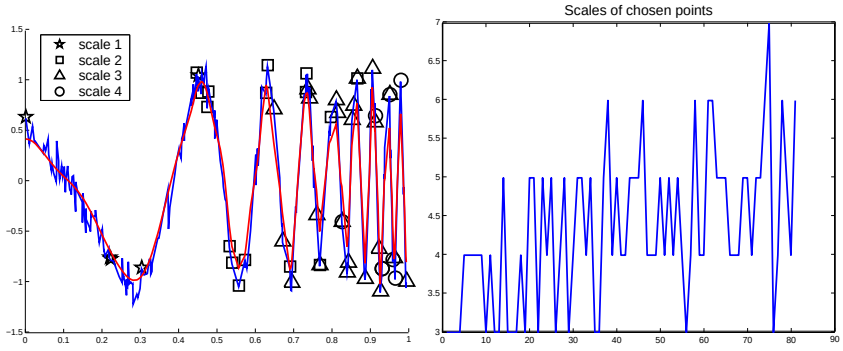


Figure 3: These illustrating figures explain the learning of the $\cos(\exp(\omega t))$ function with low noise level. Selected learning points belong to different scales, depending on the local frequency of the function to learn. Right figure shows that selected points belong higher and higher scales with iterations, namely, the sources of information correlated with the residue are more and more local with iterations.

the multiple kernels. The LARS improves the sparsity of the solution, whereas the multiple kernels improve the results on signals that require a multiple scale approach.

ϵ-SVM achieves the best results for Ramp and HeaviSine signals. This can be explained by the fact that the Ramp and HeaviSine signals are almost uniform in term of frequency. The ϵ tube algorithm of the SVM regression is especially efficient on this kind of problem.

It is important to note that LARS-RV and LARS-σ_s are parameter free methods when combined with the heuristic described in section 3.4. Best results are achieved with LARS-$\sum_i |\beta_i|$, however, LARS-RV results are almost equivalent without any parameters.

4.2 Real data

Experiments are carried out over regression data bases pyrim and triazines available in the UCI repository (Blake & Merz, 1998). We compare our results with (Chang & Lin, 2005).

The experimental procedure for real data is the following one: Thirty training/testing set are produced randomly, table 2 presents mean and standard deviation of MSE (mean square error) on the test set. 80% of the points are used for training and the remaining paramters (ν, $\sum_i |\beta_i|$...) are computed by cross validation on the learning set.

The results obtained with LARS algorithm are either equivalent to Chang and Lin's ones or better. ϵ-SVM solution is not really competitive but it gives an interesting information on the number of support vectors required for each solution. LARS-RV and LARS-σ_s results are very interesting: they are parameter free using the heuristic describe in section 3.4, moreover the LARS-RV achieves the best results for pyrim.

Nb kernel	1			
Algo	SVM		LARS	
	(Chang & Lin, 2005)	ϵ-SVM	$\sum_i \lvert\beta_i\rvert$	RV
pyrim	0.007 ± 0.007	0.009 ± 0.016	0.010 ± 0.011	0.011 ± 0.009
	–	37.11	29.67	31.20
	–	0	0	0
triazines	0.021 ± 0.005	0.022 ± 0.006	0.022 ± 0.006	0.022 ± 0.005
	–	80.80	37.00	42.40
	–	0	0	0

Nb kernel	6 (Multiple Kernels)		
Algo	LARS		
	$\sum_i \lvert\beta_i\rvert$	RV	σ_s
pyrim	0.007 ± 0.008	$\mathbf{0.007 \pm 0.006}$	0.008 ± 0.009
	37.03	38.08	39.26
	13	**17**	0
triazines	$\mathbf{0.019 \pm 0.006}$	0.020 ± 0.005	0.022 ± 0.008
	37.00	52.03	31.86
	22	8	0

Table 2: Results of SVM and LARS for the different regression database. Mean and standard deviation of MSE on the test set (30 runs), number of support vectors used for each solution, number of best performances.

5 Conclusion

This paper enables us to meet two objectives: proposing a sparse kernel-based solution for the regression problem and introducing new solutions for the bias-variance compromise problem.

The LARS offers opportunities for both problems. It gives an exact solution to the LASSO problem, which is sparse due to ℓ_1 regularization. The ability of dealing with multiple kernels allows rough setting for the kernel parameters. Then, LARS algorithm optimizes the parameters at each iteration, selecting a new point in the optimal scale. The fact that the LARS computes the regularization path offers efficient and non parametric settings for the compromise parameter.

This methodology gives good results on synthetic and real data. In the meantime, the required time computation is reduced compared with SVM, due to the sparsity of the obtained solutions.

The perspectives of this work are threefold. We have to test LARS-methods on more databases to evaluate all properties. We also want to improve the multiple kernel building. Indeed, the use of the current σ_k often leads to a slight overfitting and to less sparse solutions. Finally, we will analyze the LARS-RV results deeper, to explain the good results and possibly improve them.

References

BACH F., THIBAUX R. & JORDAN M. (2004). Computing regularization paths for learning

multiple kernels. In *Advances in Neural Information Processing Systems*, volume 17.

BI J., BENNETT K., EMBRECHTS M., BRENEMAN C. & SONG M. (2003). Dimensionality reduction via sparse support vector machines. *Journal of Machine Learning Research*, **3**, 1229–1243.

BLAKE C. & MERZ C. (1998). UCI rep. of machine learning databases.

CHANG M. & LIN C. (2005). Leave-one-out bounds for support vector regression model selection. *Neural Computation*.

CHEN S. (1995). *Basis Pursuit*. PhD thesis, Department of Statistics, Stanford University.

CHEN S., DONOHO D. & SAUNDERS M. (1998). Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, **20**(1), 33–61.

DONOHO D. & JOHNSTONE I. (1994). Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, **81**, 425–455.

EFRON B., HASTIE T., JOHNSTONE I. & TIBSHIRANI R. (2004). Least angle regression. *Annals of statistics*, **32**(2), 407–499.

GIROSI F., JONES M. & POGGIO T. (1995). Regularization theory and neural networks architectures. *Neural Computation*, **7**(2), 219–269.

GRANDVALET Y. (1998). Least absolute shrinkage is equivalent to quadratic penalization. In *ICANN*, p. 201–206.

KIMELDORF G. & WAHBA G. (1971). Some results on Tchebycheffian spline functions. *J. Math. Anal. Applic.*, **33**, 82–95.

LJUNG L. (1987). *System Identification - Theory for the User*.

LOOSLI G., CANU S., VISHWANATHAN S., SMOLA A. J. & CHATTOPADHYAY M. (2004). Une boîte à outils rapide et simple pour les svm. In *CAp*.

MALLAT S. & ZHANG Z. (1993). Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, **41**(12), 3397–3415.

PATI Y. C., REZAIIFAR R. & KRISHNAPRASAD P. S. (1993). Orthogonal matching pursuits : recursive function approximation with applications to wavelet decomposition. In *Proceedings of the 27th Asilomar Conference in Signals, Systems, and Computers*.

SCHÖLKOPF B. & SMOLA A. (2002). *Learning with kernels*.

TIBSHIRANI R. (1996). Regression shrinkage and selection via the lasso. *J. Royal. Statist.*, **58**(1), 267–288.

TIKHONOV A. & ARSÉNIN V. (1977). *Solutions of ill-posed problems*. W.H. Winston.

VINCENT P. & BENGIO Y. (2002). Kernel matching pursuit. *Machine Learning Journal*, **48**(1), 165–187.

WAHBA G. (1990). *Spline Models for Observational Data*. Series in Applied Mathematics, Vol. 59, SIAM.

Méthodologie de sélection de caractéristiques pour la classification d'images satellitaires

Marine Campedel et Eric Moulines

Ecole Nationale Supérieure des Télécommunications
Laboratoire de Traitement du Signal et des Images
46, rue Barrault, 75013 Paris
marine.campedel@enst.fr, eric.moulines@enst.fr

Résumé :
Choisir les descripteurs d'une image en vue de son indexation n'est pas aisé, du fait de la variété des choix présentés dans la littérature. Nous développons à cet effet une méthodologie permettant de comparer différents ensembles de caractéristiques extraits d'une même base d'images. Cette méthodologie repose sur des algorithmes supervisés et non supervisés de sélection de caractéristiques. Elle est appliquée à une base d'images satellitaires dont sont extraites des caractéristiques texturales variées.[1].
Mots-clés : Sélection de caractéristiques, classification, apprentissage supervisé et non supervisé, machine à vecteurs supports, textures, pertinence.

1 Problème et méthodologie

Le travail présenté s'inscrit dans un vaste projet d'indexation d'images satellitaires. Ces images, de plus en plus nombreuses, sont très mal exploitées du fait de leur diversité et de leurs grandes tailles. Nous nous intéressons à la détermination du meilleur ensemble de caractéristiques à extraire pour leur indexation. Nous prolongeons nos travaux précédents (Campedel & Moulines, 2004), en introduisant des algorithmes de sélection de caractéristiques non supervisés[2].
L'idée fondamentale de nos méthodes non supervisées, qui peuvent être qualifiées de *filtre*, consiste à exploiter une mesure de similarité des caractéristiques par l'intermédiaire d'une clusterisation. Il s'agit, à l'instar des auteurs de (Mitra *et al.*, 2002), de regrouper les attributs similaires puis de choisir des représentants pour chacun des groupes produits. L'exploration méthodique de l'espace des sous-ensembles d'attributs est remplacée (et donc simplifiée) par l'opération de clusterisation. Nous exploitons ainsi deux algorithmes de clusterisation, SVC (Ben-Hur *et al.*, 2001) et kMeans, sous le nom de SVC-FS et kMeans-FS (FS pour feature Selection).

[1]Cette étude est financée par le Centre National d'Etudes Spatiales (CNES)

[2]En effet l'étiquetage manuel peut ne pas être envisagé du fait de la taille de la base, mais aussi parce qu'il n'aurait pas de sens envers l'application finale. En particulier, les images satellitaires sont exploitées différemment par les spécialistes de la géologie, de l'agriculture ou de l'urbanisme.

2 Expérimentations et conclusion

Les sélections produites par les différents algorithmes sont comparées à l'aide de l'entropie de représentation (H) et de performances de classification (k-plus-proches-voisins, classificateur de Fisher et SVM) évaluées par validation croisée.
La base de données est constituée de 600 vecteurs de caractéristiques (les 78 coefficients d'Haralick (Haralick *et al.*, 1973) dans le cas présenté), extraits d'imagettes 64 $\times$ 64 issues de scènes SPOT 5, à raison de 100 imagettes par classe de texture (nuage, mer, désert, forêt, ville, champs). Les résultats majeurs, présentés dans le tableau 1, sont :

- Le nombre de caractéristiques peut être réduit d'un facteur 4 sans accroissement de l'erreur de classification ;
- Les méthodes de sélection non supervisées sont aussi performantes que les méthodes supervisées et souvent plus rapides ;
- L'entropie de représentation associée aux méthodes supervisées est plus importante, ce qui est significatif de leur capacité à choisir des sous-ensembles moins redondants d'attributs.

Satellite	600 exemples - 6 classes D = 78, 5 validations croisées			
	kPPV(k=8)	Fisher	SVM	H
d = D	13.8±2.2	30.8±4.3	7.7±2.0	
d = 20				
Fisher	14.8±1.9	**30.0±3.0**	9.0±2.7	0.51
Relieff	18.0±2.0	30.7±3.6	14.8±4.0	0.19
SVM-RFE	16.0±2.7	33.3±3.7	12.3±3.4	0.48
l_2-AROM	15.8±2.3	33.2±4.7	11.0±1.5	0.88
MIC	14.8±1.6	34.0±2.7	8.8±1.5	**1.09**
kMeans-FS	**13.5±2.8**	**30.3±5.2**	**6.7±1.0**	**1.00**
SVC-FS	14.8±2.6	31.7±4.2	11.0±2.3	**1.13**

TAB. 1 – Sélection de 20 attributs parmi 78. Les méthodes de sélection non supervisées sont MIC, kMeans-FS et SVC-FS.

Références

BEN-HUR A., HORN D., SIEGELMANN H. & VAPNIK V. (2001). Support vector clustering. *Journal of Machine Learning Research*, **2**, 125–137.

CAMPEDEL M. & MOULINES E. (2004). Modélisation de textures par sélection de caractéristiques. In *CAp04*, p. 1–16, Montpellier.

HARALICK R. M., SHANMUGAN K. & DINSTEIN I. (1973). Textural features for image classification. *IEEE Transactions on Systems, Man and Cybernetics*, **3(6)**, 610–621.

MITRA P., MURTHY C. & PAL S. (2002). Unsupervised feature selection using feature similarity. *IEEE Trans. Pattern Anal. Mach. Intell.*, **24**(3), 301–312.

Semantic Learning Methods: Application to Image Retrieval

Philippe H. Gosselin and Matthieu Cord

ETIS / CNRS UMR 8051
6, avenue du Ponceau, 95014 Cergy-Pontoise, France

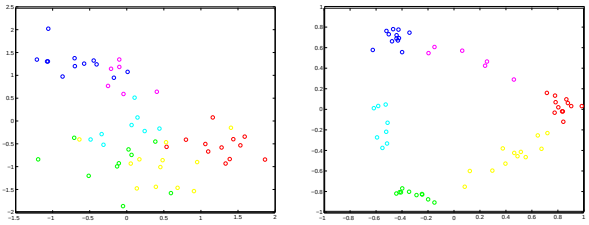
Introduction. Indexing, retrieval and classification tools are usefull to process large digital document collections. Digital documents can be automatically gathered into *clusters* or *concepts*. For instance, concepts like keywords are useful for text database organization. Concepts do not necessarily form a clustering, *i.e.* a document can belong to several concepts. Documents are usually represented by low-level features, computed automatically, and learning techniques based on relevance feedback are used to retrieve the concepts (Tong & Koller, 2000; Gosselin & Cord, 2004).

According to an incomplete set of partial labels, we propose two semantic learning methods to improve the representation of the document collection, even if the size, the number and the structure of the concepts are unknown. These methods may learn a lot of concepts with many mixed information. We build these methods in a general framework, thus powerful learning or semi-supervised learning methods may be used to retrieve, classify, or browse data.

Challenge. Suppose that we have a set of documents, each of them represented by a vector $\mathbf{x}_i \in \mathbb{R}^{N_d}$ of $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_{N_x}\}$, and a set of labels $Y = \{\mathbf{y}_1, \ldots, \mathbf{y}_{N_y}\}$. For instance, X can be the set of feature vectors of an image database, and each $\mathbf{y}_p$ contains the labels provided by a user during the retrieval session p. We suppose that labels are sampled from a hidden set of concepts. The documents are gathered in a finite (but unknown) number N_c of concepts, and these concepts do not necessarily form a clustering. Thus, a document represented by a vector $\mathbf{x}_i$ can belong to several concepts. For instance on an image database, one can find buildings, cars, houses, or landscapes, but also cars in front of a building or a house, or houses in a landscape.

A vector $\mathbf{y}_p \in [-1, 1]^{N_x}$ is a partial labeling of the set X, according to one of the concepts. Every positive value y_{ip} means that the document represented by $\mathbf{x}_i$ is in this concept, as much as y_{ip} is close to 1. Every negative value y_{ip} means that the document represented by $\mathbf{x}_i$ is not in this concept, as much as y_{ip} is close to -1. Every value y_{ip} close to zero means that there is no information for $\mathbf{x}_i$ about this concept. We also suppose that the number of non-zero values in $\mathbf{y}_p$ is small against the size of the concept. Thus, from only one $\mathbf{y}_p$, it is impossible to build the corresponding concept. The challenge is to use this set of partial labeling in order to learn the concepts.

Vector-based update. The repartition of the centers in space is important in the case of mixed concepts. As we wish to represent any possible combination of memberships, centers should be at the same distance from each other. The building of equidistant cen-



(a) (b)

FIG. 1 – Toy example with 3 mixed concepts, blue = concept 1, red = concept 2, green = concept 3, magenta = concept 1 and 2, cyan = concept 1 and 3, yellow = concept 2 and 3. (a) Initial set. (b) Concept Vector Learning method.

ters has implications on the dimension N_d of vectors. A theorem also shows, for N_c concepts in a space of $N_c - 1$ dimensions, that distance between equidistant concept centers is unique, modulo a scaling. It is easy to see that in higher dimension, this property is no longer true. In the computation of possible centers, we exploit this property in order to get equidistant centers.

Kernel-based update (Gosselin & Cord, 2005). The knowledge contained in the matrix Y can be used to update the similarity matrix. The similarity matrix is the matrix of all similarities between image pairs. The strategy is based on a kernel matrix adaptation, and is designed to model mixed concepts. We also manage the complexity constraint using efficient eigenvalue matrix decomposition ; the method has a $O(N_x)$ complexity and memory need, and so it is applicable to large databases.

Experiments. Tests are carried out on the generalist COREL photo database. Results show that the proposed method, and especialy the vector-based one, increase significally the performances of the system in comparison to distance learning methods.

Conclusion. We introduced a feature-based and a kernel-based semantic learning methods, to improve the performances of an image retrieval system. These methods deal with the constraints of the CBIR framework, and are able to enhance the database representation with a partial and incomplete knowledge about the structure of the database. Tests carried out on a generalist database show that the data representation may be improved by these learning strategies. Using the proposed learning protocol, the vector-based technique gives the best results.

Références

GOSSELIN P. & CORD M. (2004). RETIN AL : An active learning strategy for image category retrieval. In *IEEE International Conference on Image Processing*, Singapore.

GOSSELIN P. & CORD M. (2005). Semantic kernel learning for interactive image retrieval. In *IEEE International Conference on Image Processing*, Genova, Italy.

TONG S. & KOLLER D. (2000). Support vector machine active learning with applications to text classification. *International Conference on Machine Learning*, p. 999–1006.

Détection de contexte par l'apprentissage*

Gaëlle Loosli[1], Sang-Goog Lee[2], Stéphane Canu[1]

[1] PSI, CNRS FRE2645, INSA de Rouen, FRANCE
gaelle.loosli@insa-rouen.fr and
http://asi.insa-rouen.fr/∼gloosli

[2] Interaction Lab./Context Awareness TG,
Samsung Advanced Institute of Technology, Korea
sglee@samsung.com

Abstract : Nos travaux s'intéressent à la détection et l'identification de contexte d'un être humain. Ce domaine, aussi appelé « affective computing », requiert la définition de contextes et d'émotions, d'états affectifs ou émotionnels. Ces définitions étant particulièrement délicates à déterminer, nous pensons qu'une approche ascendante (regarder ce qu'il est possible de voir dans les données et en déduire des contextes) est plus abordable que l'approche descendante (définir et caractériser les états et les rechercher dans les données). Nous présentons donc une méthode de segmentation de signaux physiologiques à l'aide de méthodes d'apprentissage non paramétriques, à savoir les SVM à une classe. **Mots-clés** : OC-SVM, Segmentation, Détection de rupture, Context-Aware, Affective Computing

Introduction

Dans un monde où les machines sont de plus en plus présentes, on constate de plus en plus de situations qui provoquent la frustration des utilisateurs. Avoir des machines aptes à prendre en compte ces humeurs est un objectif récurrent de beaucoup de travaux. A cette fin nous proposons une approche basée sur un système d'acquisition de données physiologiques, système porté par l'utilisateur. La capacité « d'apprendre » est essentielle pour cette tâche car chaque personne a un fonctionnement qui lui est propre et on ne peut se contenter de connaître les caractéristiques générales à priori pour reconnaître un état affectif.

Segmentation des signaux par OC-SVM

Le OC-SVM (SVM à une classe) est un algorithme qui vise à déterminer le contour d'une classe et qui permet de détecter les points qui ne sont pas de la classe concernée. Nous pouvons utiliser cet algorithme pour apprendre la classe courante des

*This work was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778. This publication only reflects the authors' views.

signaux et regarder si l'estimation de cette classe permet d'expliquer les données futures. L'adéquation entre l'estimation d'un OC-SVM sur les données passées et d'un autre OC-SVM sur les données futures est un test statistique approximant le test GLR (generalized likelyhood ratio), optimum sous les conditions de Neyman-Pearson. Les ruptures ainsi détectées permettent de segmenter le signal en une suite d'états qui pourront par la suite être identifiés (soit comme état connu soit comme nouvel état).

Résultats expérimentaux

Nos expériences utilisent des capteurs physiologiques (pression sanguine, rythme respiratoire, température périphérique, conductivité de la peau, activité musculaire). Nous monitorons un utilisateur équipé des capteurs dans diverses activités et notons les instants des événements (changement d'activité par exemple). Ce sont ces moments que nous espérons retrouver dans les signaux. Les taux de détection vont de 68 à 92% suivant les activités monitorées. Une observation intéressante est qu'avec la détection automatique nous trouvons des ruptures dans les signaux qui ne sont pas sont pas notées par l'observateur et qui correspondent à des changements cachés (essoufflement par exemple). Un dernier point important sur l'implémentation de la méthode est l'utilisation de OC-SimpleSVM, un algorithme qui permet de faire du traitement en ligne et de gérer la mise à jour de la solution optimale à l'arrivée d'un nouveau point [1].

Perspectives

Nous montrons le lien entre les tests statistiques et OC-SVM et nous illustrons l'intérêt d'une méthode non-paramétrique dans la segmentation de signaux dans le cadre de la détection de contexte. Nos résultats confirment également notre hypothèse, à savoir qu'il ne faut pas chercher à dresser la liste caractérisée des états attendus, mais partir des données et en tirer les états. La suite d'états automatiquement découpée doit être maintenant être traitée pour être étiqueter. Cette tâche se doit être semi-supervisée. En effet nous pouvons étiqueter un certain nombre de situation et en même temps être capables de gérer les situation nouvelles.

References

BASSEVILLE M. & NIKIFOROV I. V. (1993). *Detection of Abrupt Changes - Theory and Application*. Prentice-Hall.

DAVY M. & GODSILL S. (2002). Detection of abrupt spectral changes using support vector machines. In *Proc. IEEE ICASSP-02*.

LIEBERMAN H. & SELKER T. (2000). Out of context: Computer systems that adapt to, and learn from, context. **39**.

LOOSLI G. (2004). Fast svm toolbox in Matlab based on SimpleSVM algorithm. `http://asi.insa-rouen.fr/~gloosli/simpleSVM.html`.

PICARD R. W., PAPERT S., BENDER W., BLUMBERG B., BREAZEAL C., CAVALLO D., MACHOVER T., RESNICK M., ROY D. & STROHECKER C. (2004). Affective learning : A manifesto. *BT Technology Journal*, **22**(4), 253–269.

[1] disponible sur `http://asi.insa-rouen.fr/~gloosli/simpleSVM.html`

Modèles markoviens pour l'organisation spatiale de descripteurs d'images.

J.Blanchet, F.Forbes, C.Schmid

INRIA Rhône-Alpes
ZIRST-655 avenue de l'Europe
38330 Montbonnot Saint Martin

Résumé : Ce papier décrit une nouvelle approche probabiliste pour la reconnaissance de textures. Une image est décrite à l'aide de descripteurs locaux, ainsi que par des relations spatiales entre ces descripteurs. On peut alors associer une image à un graphe : les nœuds sont les points d'intérêt de l'image correspondant à des régions caractéristiques et les arêtes relient des régions voisines. Ajouter une telle information de voisinage permet d'améliorer les résultats de reconnaissance. Les approches actuelles consistent à modéliser les descripteurs comme des variables indépendantes, puis à rajouter l'information spatiale par le biais de poids, sans modéliser explicitement ces dépendances. Nous proposons d'introduire un modèle statistique rendant compte directement de cette dépendance entre descripteurs, par l'utilisation de champs de Markov cachés. L'estimation des paramètres de tels modèles étant en pratique difficile, nous utilisons des procédures d'estimation récentes basées sur le principe du champ moyen de la physique statistique. Nous illustrons notre méthode sur la reconnaissance d'images uni et multi-textures. Les résultats obtenus sont prometteurs.

Mots-clés : Champ de Markov, Algorithme de type EM, Relaxation, Classification, Apprentissage statistique, Reconnaissance de textures.

1 Introduction

Une notion clé en vision par ordinateur est celle de descripteurs, caractérisations locales d'une image. De manière générale, un bon descripteur se doit d'être résistant aux occlusions, ainsi qu'invariant à diverses transformations géométriques de l'image. La recherche de "bons" descripteurs a déjà fait l'objet de nombreuses études, alors que la prise en compte de leur organisation spatiale reste un problème très ouvert. Nous proposons donc de coupler l'utilisation d'outils performants de vision et de statistique, dans le but de modéliser l'organisation spatiale de tels descripteurs.
Une tentative de prise en compte du caractère spatial des données pour la reconnaissance de textures a déjà été effectuée dans (Lazebnik *et al.*, 2003a) : dans ce travail, lors de la phase de reconnaissance, les probabilités *a posteriori* d'appartenance aux différentes classes de textures sont raffinées par l'algorithme de relaxation proposé dans (Rosenfeld *et al.*, 1976). Cependant, le voisinage n'y est pris en compte que par un

terme de poids, sans modèle explicite. Nous proposons de modéliser les descripteurs comme des variables statistiques liées, et par conséquent d'utiliser un modèle statistique paramétrique rendant compte explicitement de ces dépendances. Le modèle que nous avons choisi est celui du champ de Markov caché. L'estimation des paramètres d'un tel modèle étant difficile, nous utilisons des procédures d'estimation récentes (algorithme de type EM), basées sur l'algorithme d'*Expectation-Maximisation* (EM) et sur le principe du champ moyen issu de la physique statistique (Chandler, 1987).
La base d'apprentissage considérée dans notre présentation n'est composée que d'images uni-texturées. Cependant, les algorithmes de type EM utilisés pour l'apprentissage pourraient tout à fait être généralisés à des images multi-textures pourvu que soit déclaré l'ensemble des textures qu'elles contiennent. On pourra se référer à (Lazebnik *et al.*, 2003a) ou (Nigam *et al.*, 2000) pour plus de détails. Pour ce qui est de la reconnaissance, la méthode ne se restreint pas aux images uni-textures : chaque vecteur caractéristique est classé individuellement, si bien que les images test peuvent être multi-textures (voir Section 6.2).
Dans la Section 2, nous présentons brièvement la procédure d'extraction de caractéristiques, et la façon dont est construit le graphe de voisinage. Le modèle probabiliste utilisé (champs de Markov cachés) pour les textures est explicité dans la Section 3. Les différentes phases d'apprentissage et de reconnaissance sont décrits dans la Section 4. Nous présentons pour comparaison en Section 5 deux autres algorithmes de classification de textures : la relaxation et l'algorithme NEM. Des expériences effectuées sur des images de scènes d'intérieur ainsi qu'une discussion achèvent ce papier.

2 Extraction de caractéristiques et graphe de voisinage

Pour la phase d'extraction de caractéristiques, nous suivons la méthode décrite dans (Lazebnik *et al.*, 2003a) pour ses performances en comparaison à d'autres méthodes récentes (Bradshaw *et al.*, 2001; Kumar & Hebert, 2003; Malik *et al.*, 2001; Schmid, 2001). Le détecteur utilisé est le *Laplacien avec adaptation affine*. Brièvement, les points d'interêt détectés correspondent à des maximum locaux dans l'espace échelle du laplacien normalisé de l'intensité. A ces points sont associés des cercles de rayon l'échelle correspondante, cercles transformés en ellipse par le processus d'*adaptation affine* (Lindeberg & Garding, 1997). Outre ses performances (notamment l'invariance aux transformations affines), l'intérêt d'un tel détecteur est de permettre la définition naturelle d'un graphe de voisinage (Lazebnik *et al.*, 2003a). Un point i sera dit voisin de j s'il appartient à l'ellipse centrée sur j, grossie d'un certain nombre de pixels (15 dans notre présentation) le long de chacun de ses axes (ceci pour éviter que les points associés à de petites régions aient trop peu de voisins). Notons qu'un tel voisinage est non symétrique. On peut alors voir une image comme un graphe orienté, chaque arcs reliant un point d'interêt détecté à un autre point voisin. Dans l'optique d'utiliser les champs de Markov, définis sur un graphe non orienté, nous avons symétrisé ce voisinage, en remplaçant les arcs par des arêtes (voir figure 1). A chacune des régions (ellipses) détectées, est associé un vecteur caractéristique (descripteur). Les descripteurs utilisés sont des vecteurs de dimension 80 obtenus à partir de *spin images* (Lazebnik *et al.*, 2003b) , obtenues de la manière suivante : chaque ellipse détectée est transformée

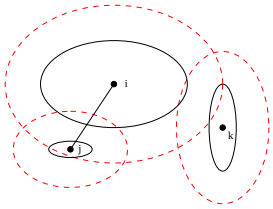


FIG. 1 – Graphe de voisinage symétrique : les points i et j sont voisins

en cercle unité, sur lequel on calcule un histogramme à 2 dimensions : une tranche de la spin image correspondant à la distance d est l'histogramme de l'intensité des pixels situés à la distance d du centre du cercle. Pour notre expérimentation, nous avons pris des spin images de taille 5×16. D'autres descripteurs sont envisageables et la méthode que nous proposons s'applique de même. A chaque point d'intérêt détecté est donc associé un vecteur caractéristique de dimension 80.

3 Modélisation des textures

L'hypothèse sur laquelle se fonde nos travaux est que les descripteurs sont des variables aléatoires dépendantes, de loi de probabilité spécifique pour chaque texture. Dans (Lazebnik *et al.*, 2003a), la distribution des descripteurs issus d'une texture donnée est modélisée par un mélange de gaussiennes, ce qui suppose que les descripteurs sont des variables indépendantes. Il existe pourtant de manière évidente une forte dépendance spatiale entre les vecteurs caractéristiques d'une même image. Afin de prendre en compte cette dépendance, nous proposons de modéliser leur distribution par un champ de Markov caché, de paramètres inconnus.

Soit $\mathbf{x} = (x_1, \ldots, x_n)$ les n descripteurs (vecteurs de dimension 80) extraits d'une image (ou partie d'image) de la texture m ($1 \leq m \leq M$). On suppose que chaque texture est composée de K sous-classes $c_{m1} \ldots c_{mK}$. Nous prendrons $K = 10$ dans nos expérimentations, indépendemment de la texture, mais la méthode resterait tout à fait identique pour un $K(m)$ dépendant de m. Pour $i = 1, \ldots, n$, on modélise la probabilité d'observer le descripteur x_i pour une image de la texture m par :

$$P(x_i|\mathbf{\Psi}_m) \;=\; \sum_{k=1}^{K} P(Z_i = c_{mk}|\mathbf{\Delta}_m)\; f(x_i|\mathbf{\Theta}_{mk}), \qquad (1)$$

où $f(x_i|\mathbf{\Theta}_{mk})$ dénote la distribution gaussienne multivariée de paramètres $\mathbf{\Theta}_{mk}$ (la moyenne μ_{mk} et la matrice de covariance Σ_{mk}). La variable aléatoire Z_i représente la sous-classe à laquelle appartient le descripteur x_i ; elle peut prendre les valeurs $\{c_{m1}, \ldots, c_{mK}\}$, et sa loi est paramétrée par $\mathbf{\Delta}_m$. $\mathbf{\Psi}_m = (\mathbf{\Delta}_m, (\mathbf{\Theta}_{mk})_{1 \leq k \leq K})$ dénote l'ensemble des paramètres du modèle pour la texture m.

Dans (Lazebnik *et al.*, 2003a), le modèle utilisé pour $P(x_i|\mathbf{\Psi}_m)$ est le mélange gaus-

sien, ce qui revient à supposer que les Z_i sont des variables *indépendantes* et que $P(Z_i = c_{mk}|\mathbf{\Delta}_m) = p_{mk}$ indépendemment du site i. $\mathbf{\Delta}_m$ est alors l'ensemble des proportions $(p_{mk})_{1\leq k\leq K}$ du mélange pour la texture m. Cela implique que les decripteurs sont également des variables indépendantes.
C'est cette limitation que nous nous proposons de surmonter, en supposant que les descripteurs sont des variables *dépendantes*. Plus précisément, les dépendances entre descripteurs voisins sont modélisées en considérant que la loi jointe des variables $Z_1, \ldots, Z_n$ est un champ de Markov discret sur le graphe défini à la section 2. Soit $\mathbf{z} = (z_1, \ldots, z_n)$ des réalisations des Z_i. On définit :

$$P(\mathbf{z}|\mathbf{\Delta}_m) \quad = \quad W(\mathbf{\Delta}_m)^{-1} \exp(-H(\mathbf{z}, \mathbf{\Delta}_m)) \tag{2}$$

où $W(\mathbf{\Delta}_m)$ est une constante de normalisation et H une fonction énergie supposée être de la forme (nous nous limitons aux interactions entre paires) :

$$H(\mathbf{z}, \mathbf{\Delta}_m) = \sum_i V_i(z_i, \boldsymbol{\alpha}_m) + \sum_{i\sim j} V_{ij}(z_i, z_j, \mathbb{B}_m)$$

(la notation $i \sim j$ signifie que les sites i et j sont voisins ; la somme de droite ne porte donc que sur des sites voisins).

Les fonctions V_i et V_{ij} se rapportent respectivement aux potentiels sur les singletons et sur les paires, de paramètres respectifs $\boldsymbol{\alpha}_m$ et $\mathbb{B}_m$. Il s'en suit $\mathbf{\Delta}_m = (\boldsymbol{\alpha}_m, \mathbb{B}_m)$.
Nous supposons que les potentiels sur les singletons ne dépendent que de la valeur z_i (et non de i), c'est à dire :

$$V_i(z_i, \boldsymbol{\alpha}_m) \quad = \quad -\alpha_m(k) \quad \text{si } z_i = c_{mk}$$

Les potentiels sur les singletons sont donc caractérisés par K poids $\boldsymbol{\alpha}_m = (\alpha_m(k))_{1\leq k\leq K}$ associés aux K sous-classes de la texture m. Dans le cas particulier où les V_{ij} sont nuls (ce qui revient à supposer qu'il n'y a pas d'interactions entre les points), $\mathbf{\Delta}_m$ se résume à $\boldsymbol{\alpha}_m$ et d'après (2),

$$P(Z_i = c_{mk}|\boldsymbol{\alpha}_m) = \frac{e^{\alpha_m(k)}}{\sum_{l=1}^{K} e^{\alpha_m(l)}}$$

si bien que $\boldsymbol{\alpha}_m$ pondère l'importance relative des différentes sous-classes de la texture m.
De même, les potentiels sur les paires V_{ij} sont supposés ne dépendre que de z_i et z_j, soit

$$V_{ij}(z_i, z_j, \mathbb{B}_m) \quad = \quad -B_m(k, l) \quad \text{si } z_i = c_{mk}, z_j = c_{ml}$$

Ils sont donc caractérisés par la matrice $\mathbb{B}_m = (B_m(k, l))_{1\leq k,l\leq K}$. Notons que si $\mathbb{B}_m = \beta_m \times Id$, le paramètre spatial $\mathbb{B}_m$ se réduit à un scalaire β_m et nous retombons sur le modèle de Potts traditionnel utilisé en segmentation d'images.
La texture m est donc représentée par un champ de Markov caché paramétré par $\mathbf{\Psi}_m = (\boldsymbol{\alpha}_m, \mathbb{B}_m, (\mathbf{\Theta}_{mk})_{1\leq k\leq K})$.

4 Apprentissage et classification

Dans la modélisation précédente, les paramètres $\mathbf{\Psi}_m = (\mathbf{\Delta}_m, (\mathbf{\Theta}_{mk})_{1\leq k\leq K})$ sont inconnus et doivent être estimés pour chaque texture m.

4.1 Apprentissage des paramètres inconnus

Pour apprendre le modèle associé à chacune des textures, on suppose que l'on dispose d'une base d'apprentissage constituée d'images identifiées comme appartenant à l'une des textures. Chaque texture va alors être apprise successivement sur les images correspondantes. Pour la texture m, doivent être estimés les paramètres $(\mu_{mk}, \Sigma_{mk})_{1\leq k\leq K}$ des lois gaussiennes ainsi que les paramètres du champ spatial, *ie* le vecteur $\boldsymbol{\alpha}_m = (\alpha_m(k))_{1\leq k\leq K}$ de K poids et la matrice d'interaction spatiale $\mathbb{B}_m$ de dimension $K \times K$. L'algorithme EM (Dempster *et al.*, 1977) est couramment utilisé pour l'estimation de paramètres dans le cas de données cachées, et en particulier pour l'estimation de mélanges indépendants. Pour de tels modèles, l'hypothèse d'indépendance mène à une implémentation facile de l'algorithme. Pour les champs de Markov cachés, du fait de la dépendance des données, l'algorithme n'est pas utilisable en pratique, et des approximations sont donc nécessaires. Dans ce papier, nous utilisons une des approximations de (Celeux *et al.*, 2003), basée sur le principe du champ moyen. Les procédures qui en découlent ont la particularité de prendre en compte la structure markovienne tout en préservant les bonnes qualités de EM. L'algorithme EM variationnel de (Zhang, 1992) en est un cas particulier. Nous utiliserons cependant plutôt les algorithmes de *type champ moyen* (et en particulier l'algorithme en *champ simulé*) pour leurs performances dans le cadre de la segmentation (Celeux *et al.*, 2003). Notons cependant qu'il est nécessaire de généraliser ces algorithmes pour permettre l'estimation de la matrice $\mathbb{B}_m$ (et non seulement d'un scalaire β_m comme dans le modèle de Potts originel).
Dans le cas du champ de Markov caché de paramètre $\mathbf{\Psi}_m$, deux difficultés apparaissent lors de la mise en œuvre de l'algorithme :

- la constante de normalisation $W(\mathbf{\Delta}_m)$ de (2)
- les probabilités conditionnelles $P(z_i|\mathbf{x}, \mathbf{\Psi}_m)$ et $P(z_i, z_j, j \in \mathcal{V}(i)|\mathbf{x}, \mathbf{\Psi}_m)$, où $\mathcal{V}(i)$ désigne l'ensemble des voisins de i

ne peuvent être calculées de manière exacte. Le principe du champ moyen consiste à se ramener à un système de particules indépendantes (sur lequel l'algorithme EM pourra être appliqué) en négligeant, pour un site i, les fluctuations de ses voisins autour de leur moyenne (*ie* en fixant $\forall j \in \mathcal{V}(i),\ z_j = I\!E(Z_j)$). Plus généralement, on parle d'approximation de type champ moyen lorsque, pour un site i, ses voisins sont fixés à des constantes. Ce champ de constantes $\tilde{z}_1, \ldots, \tilde{z}_n$ n'est pas arbitraire, il doit satisfaire certaines conditions (Celeux *et al.*, 2003). La distribution markovienne (2) peut alors être approximée par :

$$P(\mathbf{z}|\mathbf{\Delta}_m) \simeq \prod_{i=1}^{n} P(z_i|\tilde{z}_j, j \in \mathcal{V}(i), \mathbf{\Psi}_m)$$

De même, la loi jointe $P(\mathbf{x}, \mathbf{z}|\mathbf{\Psi}_m)$ et la loi markovienne $P(\mathbf{z}|\mathbf{x}, \mathbf{\Psi}_m)$ se trouvent approximées par des distributions factorisées. L'utilisation de telles approximations mène

à des algorithmes itératifs à deux étapes **(a)** et **(b)** : à l'itération (q),

(a) Créer, à partir des observations $\mathbf{x}$ et de l'estimation courante des paramètres $\mathbf{\Psi}_m^{(q)}$, une configuration $\tilde{z}_1^{(q)}, \ldots \tilde{z}_n^{(q)}$, *i.e.* des valeurs pour les Z_i.

De part les approximations précédentes, les deux problèmes rencontrés lors de la mise en œuvre de EM disparaissent, et par suite,

(b) Appliquer l'algorithme EM sur ce modèle factorisé, afin d'obtenir, à partir du paramètre courant $\mathbf{\Psi}_m^{(q-1)}$, une nouvelle estimation $\mathbf{\Psi}_m^{(q)}$.

En particulier, l'algorithme en *champ moyen* consiste, à partir de la distribution conditionnelle $P(\mathbf{z}|\mathbf{x}, \mathbf{\Psi}_m^{(q-1)})$, à fixer les $\tilde{z}_i^{(q)}$ à l'estimation de leur moyenne, l'algorithme en *champ modal* à l'estimation de leur mode et l'algorithme en *champ simulé* à les simuler. En pratique, pour l'étape **(b)**, une seule itération de l'algorithme EM est suffisante. Dans ce cas, l'algorithme en champ moyen est l'algorithme *EM variationnel* de (Zhang, 1992).

L'utilisation d'un de ces algorithmes nous permet, pour chaque texture m, d'obtenir des estimateurs $(\hat{\mu}_{mk}, \hat{\Sigma}_{mk})_{1 \leq k \leq K}$ des lois gaussiennes, ainsi que des estimateurs $\hat{\mathbb{B}}_m$ et $\hat{\boldsymbol{\alpha}}_m$ des paramètres du champ caché. Cet ensemble de paramètres $\hat{\mathbf{\Psi}}_m$ va ensuite être utilisé pour classer les régions d'une image test dans une des textures apprises.

4.2 Classification d'une image test

L'objectif est de classer individuellement chacune des régions dans une des M textures. Notons encore $\mathbf{x} = (x_1, \ldots, x_n)$ les n descripteurs extraits d'une image (ou partie d'image) de texture inconnue (image test). Chaque descripteur est susceptible d'être issu d'une des M textures possibles, et donc d'une des MK sous-classes possibles. Il est alors naturel de modéliser le champ caché par un champ de Markov discret, pouvant prendre les valeurs $\{c_{mk}, m \in [1, M], k \in [1, K]\}$. Pour $i = 1, \ldots, n$, on modélise donc la probabilité d'observer le descripteur x_i par :

$$P(x_i|\mathbf{\Psi}) \;=\; \sum_{m=1}^{M} \sum_{k=1}^{K} P(Z_i = c_{mk}|\mathbf{\Delta})\; f(x_i|\mathbf{\Theta}_{mk}), \tag{3}$$

Comme dans la section 3, la fonction d'énergie du champ spatial Z est supposée se décomposer en fonctions potentielles sur les singletons et sur les paires. Son paramètre noté $\mathbf{\Delta}$ s'écrit alors sous la forme $\mathbf{\Delta} = (\boldsymbol{\alpha}, \mathbb{B})$, où $\boldsymbol{\alpha}$ est un vecteur de dimension MK pondèrant l'importance relative des différentes sous-classes, et $\mathbb{B}$ est une matrice $MK \times MK$ modélisant les interactions entre sous-classes associées à des sites voisins. Il est alors naturel de fixer $\boldsymbol{\alpha}$ égal aux potentiels $(\hat{\boldsymbol{\alpha}}_m)_{1 \leq m \leq M}$ appris pendant la phase d'apprentissage. Notons que cela revient à supposer que les différentes textures sont *a priori* équiprobables. De même, les termes de la matrice $\mathbb{B}$ correspondant aux potentiels d'interaction entre sous-classes c_{mk} et c_{ml} d'une même texture m sont fixés aux $\hat{B}_m(k, l)$ appris. Les autres termes, au contraire, concernent les interactions entre sous-classes associées à des textures différentes. Une possibilité pour les estimer serait d'effectuer l'apprentissage sur des images multi-textures, mais comme souligné dans (Lazebnik *et al.*, 2003a), l'estimation de ces termes serait mauvaise du fait du faible nombre d'arcs entre sous-classes de textures différentes. En pratique ils sont fixés à

une valeur constante qui peut varier selon le degré d'interaction que l'on souhaite (-10 pour nos expérimentations). La matrice $\hat{\mathbb{B}}$ est donc construite de la manière suivante : sur la diagonale, les matrices $\hat{\mathbb{B}}_m$, et en dehors, la constante choisie. Enfin, il est logique de fixer les $\mathbf{\Theta}_{mk}$ aux $\hat{\mathbf{\Theta}}_{mk}$ obtenus par l'apprentissage. Au final, on obtient donc une valeur du paramètre $\hat{\mathbf{\Psi}} = (\hat{\boldsymbol{\alpha}}, \hat{\mathbb{B}}, \hat{\mathbf{\Theta}})$ de la loi d'une image quelconque.
Une texture m étant composée des K sous-classes $c_{m1}, \ldots, c_{mK}$, il est naturel de classer un descripteur x_i dans la texture m maximisant $\sum_{k=1}^{K} P(Z_i = c_{mk}|x_i, \hat{\mathbf{\Psi}})$ et donc maximisant $\sum_{k=1}^{K} P(Z_i = c_{mk}|\hat{\mathbf{\Delta}})\ f(x_i|\hat{\mathbf{\Theta}}_{mk})$. Cependant, la loi markovienne de Z_i fait intervenir la classification inconnue z_j des sites j voisins de i, et n'est donc pas calculable directement. On peut cependant réappliquer le principe du champ moyen sur les paramètres $\hat{\mathbf{\Psi}}$ laissés fixes. Pour toute sous-classe c, notons $t_{i,c}^{(q-1)}$ une approximation à l'itération $(q-1)$ de $t_{i,c} \equiv P(Z_i = c|x_i, \hat{\mathbf{\Psi}})$. A l'itération (q) pour le site i, un nouveau champ de voisins $\tilde{z}_1^{(q)}, \ldots, \tilde{z}_n^{(q)}$ est créé à l' étape **(a)** à partir des paramètres $\hat{\mathbf{\Psi}}$ et des $t_{i,c}^{(q-1)}$. Le principe du champ moyen conduit à réestimer à l'étape **(b)** les $t_{i,c}^{(q)}$ par :

$$t_{i,c}^{(q)} \propto P(Z_i = c|(\tilde{z}_j^{(q)})_{j\in\mathcal{V}(i)}, \hat{\mathbf{\Delta}})\ f(x_i|\hat{\mathbf{\Theta}}_c)$$

c'est à dire par :

$$t_{i,c}^{(q)} \propto \exp(\hat{\boldsymbol{\alpha}}_c + \sum_{j\in\mathcal{V}(i)} B(c, \tilde{z}_j^{(q)}))\ f(x_i|\hat{\mathbf{\Theta}}_c).$$

Pour rendre la comparaison avec d'autres algorithmes (voir Section 5) plus claire, on utilisera également la notation $\tilde{z}_{j,c'}^{(q)}$ qui vaut 1 si $\tilde{z}_j$ appartient à la sous-classe c', et 0 sinon. La formule précédente s'écrit alors :

$$t_{i,c}^{(q)} \propto \exp(\hat{\boldsymbol{\alpha}}_c)\ f(x_i|\hat{\mathbf{\Theta}}_c)\ \exp(\sum_{j\in\mathcal{V}(i)} \sum_{c'} B(c, c')\tilde{z}_{j,c'}^{(q)}). \quad (4)$$

5 Autres algorithmes de classification de textures

Pour comparaison, nous considérons également d'autres algorithmes pour l'apprentissage de textures : la relaxation (Rosenfeld *et al.*, 1976), et l'algorithme Neighborhood EM (Ambroise *et al.*, 1997).

5.1 Relaxation

Il s'agit d'une méthode courante en vision, notamment utilisée dans (Lazebnik *et al.*, 2003a), pour ajouter simplement du spatial, par un terme de poids. Le principe consiste à raffiner les probabilités $t_{i,c} = P(Z_i = c|x_i, \mathbf{\Psi})$ utilisées dans la règle de classification. A l'itération (q), les nouvelles probabilités $t_{i,c}^{(q)}$ sont réestimées à partir des $t_{i,c}^{(q-1)}$ par :

$$t_{i,c}^{(q)} \propto t_{i,c}^{(q-1)}(1 + \sum_{j\in\mathcal{V}(i)} \sum_{c'} B(c, c')\ t_{j,c'}^{(q-1)}) \quad (5)$$

où le terme $B(c, c')$ exprime à quel point les sous-classes c et c' sont compatibles. Dans (Lazebnik *et al.*, 2003a), ce terme est calculé comme une co-occurence pour deux sous-classes c et c' associées à une même texture, et fixé à une constante négative sinon. Notons que la matrice $B(c, c')$ ainsi définie joue un rôle comparable à la matrice $\mathbb{B}$ du champ de Markov : elle traduit la force d'interaction entre les différentes sous-classes.
Dans (Lazebnik *et al.*, 2003a), la méthode de classification de textures est la suivante : l'algorithme est initialisé par

$$t_{i,c}^{(0)} = P(Z_i = c|x_i, \mathbf{\Psi}^{\text{EM}})$$

où $\mathbf{\Psi}^{\text{EM}}$ désigne une estimateur de $\mathbf{\Psi}$ obtenu par application de l'algorithme EM pour chacune des textures (ce qui suppose que les descripteurs sont indépendants). Notons que dans ce cas, $\mathbf{\Psi}$ correpond aux proportions $\{p_c\} = \{p_{mk}, m \in [1, M], k \in [1, K]\}$ du mélange, ainsi qu'aux paramètres $\{\mathbf{\Theta}_c\} = \{\mathbf{\Theta}_{mk}, m \in [1, M], k \in [1, K]\}$ des gaussiennes. L'algorithme de relaxation ne convergeant pas, un certain nombre d'itérations sont effectuées (N=200 en pratique). Un descripteur x_i est enfin classé en fonction des $t_{i,c}^{(N)}$, selon la même règle que dans la Section 4.2.

Notons qu'une classification de base obtenue à l'aide des $t_{i,c}^{(0)}$ consiste à classer un descripteur x_i dans la texture de plus grande vraisemblance relativement à x_i pour un modèle de mélange indépendant (c'est la méthode que nous appellerons "Maximum de vraisemblance" dans la Section 6).

5.2 Algorithme Neighborhood EM

L'algorithme *Neighborhood EM* ou NEM (Ambroise *et al.*, 1997) est une pénalisation de l'algorithme EM, permettant d'ajouter de l'information spatiale pour des données modélisées par un mélange indépendant. S'inspirant de la Section 4, une méthode de classification de textures basées sur cet algorithme est la suivante : les paramètres $\{p_c^{\text{NEM}}\}$ (les proportions du mélange) et $\{\mathbf{\Theta}_c^{\text{NEM}}\}$ (les paramètres des gaussiennes) des différentes textures sont appris en appliquant NEM. Dans la phase de test, comme souligné en Section 4.2, la classification pourrait consiter à classer un descripteur dans la texture m maximisant $\sum_k P(Z_i = c_{mk}|x_i, \hat{\mathbf{\Psi}})$, mais en pratique les résultats sont grandement améliorés en réappliquant NEM au mélange avec paramètres fixés $\hat{\mathbf{\Psi}}$. Dans ce cas, les probabilités $t_{i,c}$ sont réestimées à l'itération (q) par :

$$t_{i,c}^{(q)} \propto p_c^{\text{NEM}}\ f(x_i|\mathbf{\Theta}_c^{\text{NEM}})\ \exp\Big(\sum_{j \in \mathcal{V}(i)} \sum_{c'} B(c, c')\ t_{j,c'}^{(q-1)}\Big) \tag{6}$$

où le terme $B(c, c')$ a la même fonction et est estimé de la même manière que pour la relaxation.
Remarquons que dans l'algorithme NEM, à chaque itération, un modèle de mélange est sous-jacent, ce qui se note dans la formule (6) par la présence du terme $p_c^{\text{NEM}}\ f(x_i|\mathbf{\Theta}_c^{\text{NEM}})$. Dans la relaxation au contraire, la formule (5) nous montre qu'aucun modèle sous-jacent n'est supposé. Dans la formule (4) des algorithmes de type champ moyen, comme souligné en Section 3, le terme $\exp(\hat{\boldsymbol{\alpha}}_c)$ peut être comparé aux proportions p_c^{NEM} du mélange. En fait, dans le cas particulier où les paramètres sont fixés, l'algorithme en

champ moyen est équivalent à l'algorithme NEM. Pour les algorithmes en champs modal et simulé, le terme dans l'exponentielle diffère : $t_{j,c'}^{(q-1)}$ est un réel estimant $P(Z_i = c'|x_i, \hat{\mathbf{\Psi}})$, alors que $\tilde{z}_{j,c'}^{(q)}$ prend la valeur 0 ou 1 selon que Z_j ait été affecté à la classe c' ou simulé dans c' par la loi $P(.|x_i, (\tilde{z}_j)_{j\in\mathcal{V}(i)}, \hat{\mathbf{\Psi}})$.

6 Résultats expérimentaux

Les expériences sont effectuées sur des images de 7 textures différentes dont un échantillon se trouve dans la Figure 2.

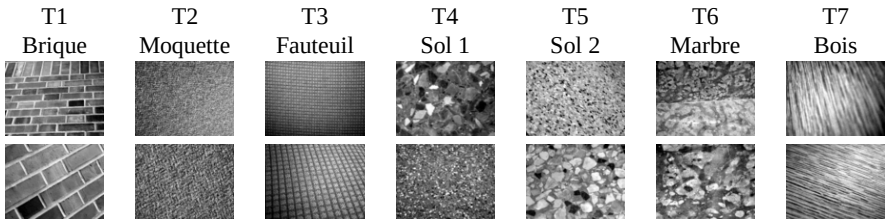


FIG. 2 – Echantillon des 7 textures utilisées pour les expériences.

La base d'apprentissage est composée de 10 images uni-texture pour chacune des 7 textures (soit un total de 70 images). Pour simplifier, nous supposons que $K = 10$ pour chacune des textures (en fait, la selection de K par l'application du Critère d'Information Bayésien (BIC) de Schwartz (Celeux *et al.*, 2003) ne semble pas apporter d'amélioration significative). De même, nous nous sommes restreints à des matrices de covariances diagonales. Parmi ces modèles, pour chaque texture m, BIC sélectionne celles de la forme $\Sigma_{mk} = \sigma_m^2\ I$ pour tout $k = 1 \ldots K$ (encore une fois, un modèle de la forme $\Sigma_{mk} = \sigma_{mk}^2\ I$ n'améliore que très peu les résultats, tout en augmentant très significativement les temps de calcul).

6.1 Reconnaissance d'images uni-texture

Nous testons notre méthode sur une base de test comprenant 10 images de chaque texture non utilisées dans la phase d'apprentissage. Le Tableau 1 donne les taux de bonne classification de l'ensemble des régions, c'est à dire, pour chaque texture m, le pourcentage de descripteurs de la texture m effectivement classés dans la texture m. Les différentes lignes se réfèrent aux algorithmes vus précédemment (les résultats correspondant aux champs moyen et modal ne sont pas reportés, car au mieux équivalents à ceux du champ simulé).

Les résultats de la Table 1 confirment l'intérêt de prendre en compte l'organisation spatiale des descripteurs : la classification par maximum de vraisemblance donne des taux de classification significativement inférieure pour les textures 1 à 5 (de -20 à -34%). Pour les textures 6 et 7, l'amélioration est moins notable. De manière générale, toutes les méthodes semblent avoir du mal à apprendre les textures 6 et 7. Ceci s'explique par

Texture	T1	T2	T3	T4	T5	T6	T7
Max. vraisemblance	48	77	52	56	51	17	30
Relaxation	78	96	72	86	80	19	42
NEM	82	98	78	88	80	20	43
Champ simulé	81	97	77	80	86	26	46

TAB. 1 – % de bonne classification des régions sur des images uni-textures.

le fait que ces deux textures contiennent des images avec de très brusques changements de luminosité, rendant l'apprentissage d'autant plus difficile.
NEM et l'algorithme en champ simulé améliorent les taux de classification pour chacune des textures par rapport à la relaxation. En effet, dans l'algorithme de relaxation, aucun modèle n'est supposé et les itérations sont indépendantes du modèle utilisé pour les données. Au contraire, NEM est originellement fait pour un modèle de mélange, si bien que ce modèle de mélange est pris en compte à chaque itération. La méthode utilisant les champs de Markov est quant à elle la seule où les descripteurs sont modélisés comme des variables statistiques dépendantes.
Le taux de bonne classification de NEM est plus élevé que celui du champ simulé pour les textures 1 à 4, mais plus faible pour les textures 5 à 7. En moyenne, c'est l'algorithme en champ simulé qui apparait le plus performant (70.43% de reconnaissance pour l'algorithme en champ simulé contre 69.86% pour NEM).
Ces premières expériences montrent qu'il y a un apport significatif à ajouter de l'information spatiale entre les descripteurs. Il apparait en outre qu'il est plus judicieux d'utiliser un modèle paramétrique, comme le modèle de mélange (NEM) ou son extension aux champs de Markov (algorithme de type champ moyen), pour l'apprentissage comme pour le test.
Notons toutefois les bonnes performances de l'algorithme de relaxation malgré l'absence des données dans l'itération (5). Ces dernières ne sont prises en compte que dans l'initialisation, ce qui correspond à un principe de maximum a priori et non a posteriori comme le recommande la théorie statistique. Une explication de ces bonnes performances est sans doute la robustesse d'un tel algorithme vis à vis de la mauvaise adéquation du modèle de mélange aux données. Il n'existe pas à ce jour de test statistique pour tester l'hypothèse de mélange mais une voie d'approche est envisageable, basée sur un prétraitement des données visant à les rendre plus proches du modèle choisi.

6.2 Sur des images multi-textures

Les différents algorithmes sont également testés sur 62 images multi-textures, dont 5 artificiellement créées. Des exemples de classifications obtenues sont présentés dans les Figures 3 à 8.
La Figure 3 est un exemple de segmentation obtenue par les 4 algorithmes étudiés précédemment : le maximum de vraisemblance, la relaxation, NEM et le champ simulé.
Il en ressort clairement que la prise en compte de l'information spatiale (par la re-

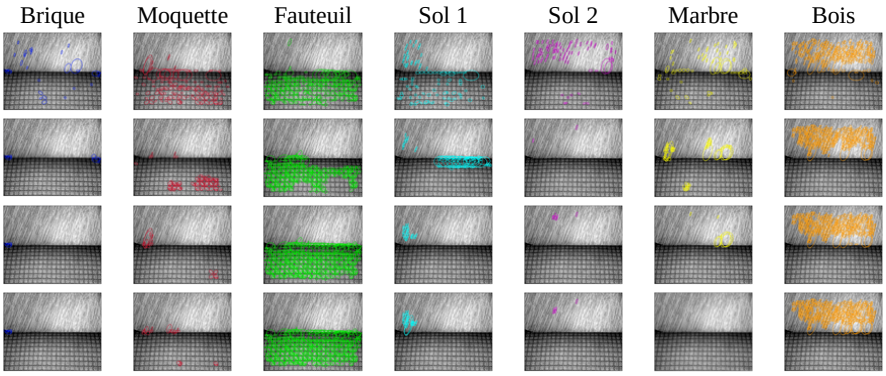


FIG. 3 – De haut en bas : classification obtenue par le maximum de vraisemblance, la relaxation, NEM et l'algorithme en champ simulé, sur une image composée de fauteuil et de bois

laxation, NEM, ou le champ simulé), améliore la classification. De plus on observe que, parmi ces algorithmes spatiaux, c'est celui utilisant une modélisation explicite des dépendances (champ de Markov caché estimé par l'algorithme en champ simulé) qui donne les meilleurs résultats. En outre, on observe que la moquette est très bien reconnue, alors que quelques erreurs subsistent avec le bois. Ces résultats vont dans le sens de ceux obtenus sur des images uni-textures, le bois ayant été mal appris.

Un autre exemple illustrant l'intérêt de la prise en compte de l'organisation spatiale des descripteurs est donné Figure 4 : l'algorithme en champ simulé permet une bien meilleure segmentation que le maximum de vraisemblance.

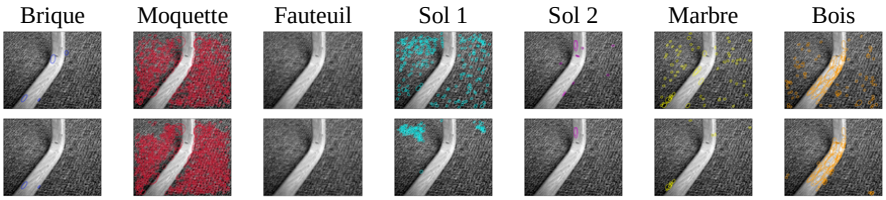


FIG. 4 – Classification obtenue par le maximum de vraisemblance (en haut) et l'algorithme en champ simulé (en bas), sur une image composée de moquette et de bois

Les Figures 5 à 8 sont des exemples de classifications obtenues par l'algorithme en champ simulé. On notera (Figure 5) les performances de cet algorithme spatial sur une image artificielle, constituée de 4 bouts d'images.

En Figure 6, on observe clairement le comportement de l'algorithme en champ simulé : le marbre est mal reconnu (ce qui était un résultat prévisible au vu du Tableau 1), mais c'est surtout avec la brique et le bois qu'il est confondu. En effet, cet algorithme

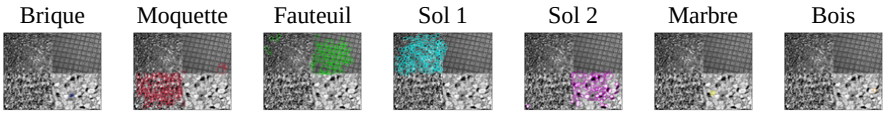


FIG. 5 – Classification obtenue par l'algorithme en champ simulé, sur une image artificielle composée de sol 1, de fauteuil, de moquette et de sol 2.

spatial tend à regrouper les sites voisins dans un même classe, et va donc, de proche en proche, classer une partie des sites correspondant au marbre dans l'une des deux textures présentes aux alentours, la brique et le bois.

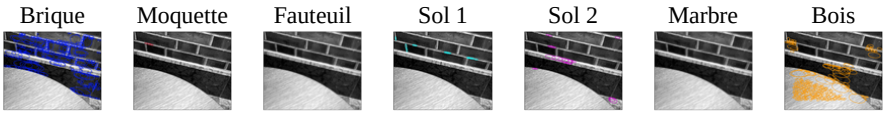


FIG. 6 – Classification obtenue par l'algorithme en champ simulé, sur une image composée de briques, de marbre et de bois.

Enfin, les Figures 7 et 8 mettent en relief les problèmes de classification lorsque la qualité de l'image diminue. En effet, en Figure 7, l'arrière plan (le bois) est flou, et les résultats obtenus sont médiocres (noter que sur l'image nette de la Figure 3, ils sont bien meilleurs). En Figure 8, une partie de l'image (en l'occurence les briques) est mal éclairée, si bien que les sites correspondants sont relativement mal classés, alors que la texture brique a été bien apprise (81% de bonne classification sur les images uni-textures). Plus qu'une limitation de nos algorithmes, c'est plutôt la qualité du descripteur que doit être mis en défaut. La possibilité d'utiliser d'autres descripteurs plus invariants au problèmes de luminosité est à envisager.
De manière générale, notons que la plupart de erreurs ont lieu près des bords, ou aux frontières entre textures différentes. Ceci suggère que le graphe de voisinage pourrait être repensé pour limiter ce phénomène.

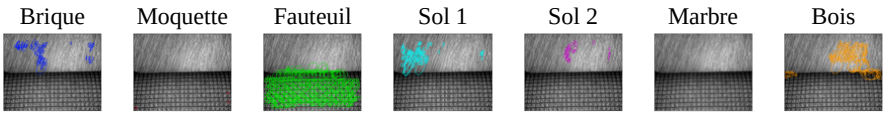


FIG. 7 – Classification obtenue par l'algorithme en champ simulé, sur une image de qualité moindre composée de fauteuil et de bois (partie bois floue).

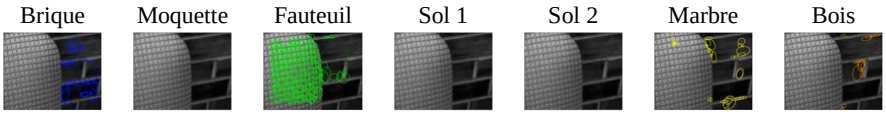


FIG. 8 – Classification obtenue par l'algorithme en champ simulé, sur une image avec mauvais éclairage composée de fauteuil et de briques.

7 Conclusion et perspectives

Notre travail s'est basé sur des techniques récentes de description d'image à l'aide de descripteurs calculés en un certain nombre de points caractéristiques de l'image. Notre objectif était de montrer que des modèles statistiques paramétriques pouvaient être introduits pour rendre compte de l'organisation spatiale et géométrique de ces descripteurs. Les champs de Markov cachés étaient des candidats naturels, que nous avons expérimentés dans le cadre de la reconnaissance de textures. L'utilisation de modèles markoviens pour la ségmentation d'images caractérisées par une grille régulière de pixels est standard, mais leur introduction en reconnaissance pour modéliser des vecteurs caractéristiques irrégulièrement espacés est nouvelle. Dans ce contexte, les paramètres de ces modèles ont une interprétation naturelle : certains (les α_{mk}) peuvent être assimilés à des proportions sur les textures, alors que d'autres (la matrice $\mathbb{B}$) à des intéractions spatiales. Dans notre méthode, ces paramètres sont estimés, ou bien fixés pour incorporer de la connaissance *a priori* sur les textures.
Des résultats obtenus sur des images uni ou multi textures sont prometteurs. Ils mettent notamment à jour l'intérêt de prendre en compte l'organisation spatiale des descripteurs. De plus, l'utilisation d'un modèle rendant explicitement compte de ces dépendances (champ markovien) améliore les taux de reconnaissance. Notons que les résultats pourraient sans doute encore être améliorés en effectuant sur les données des transformations préliminaires visant à les rendre plus proche du modèle utilisé (un mélange gaussien). Plus précisément, on pourra envisager de coupler des transformations de type Box-Cox avec des tests de gaussianité en grande dimension.
Par ailleurs, le formalisme général présenté dans ce papier pourrait être envisagé dans d'autres contextes, en reconnaissance d'objets par exemple. Mais avant cela, une étude plus spécifique du choix de la structure de voisinage serait nécessaire, en particulier concernant la définition d'un graphe de voisinage préservant au maximum l'invariance affine. Enfin, la méthodologie pourrait être testée en utilisant d'autres techniques de description d'images.

Références

AMBROISE C., DANG V. M. & GOVAERT G. (1997). Clustering of spatial data by the EM algorithm. In K. A. P. DORDRENCHT, Ed., *geoENV I- Geostatistics for Environmental Applications,Quantitative Geology and Geostatistics*, volume 9.

BRADSHAW B., SCHOLKOPF B. & PLATT J. (2001). Kernel methods for extracting local image semantics. In *Microsoft Research Technical Report, MSR-TR-2001-99*.

CELEUX G., FORBES F. & PEYRARD N. (2003). EM procedures using mean field-like approximations for Markov model-based image segmentation. In *Pattern Recognition*, volume 36(1), p. 131–144.

CHANDLER D. (1987). Introduction to modern statistical mechanics.

DEMPSTER A., LAIRD N. & RUBIN D. (1977). Maximum Likelihood from incomplete data via the EM algorithm.

KUMAR S. & HEBERT M. (2003). Man-made structure detection in natural images using a causal multiscale random field. In *Proc. CVPR*.

LAZEBNIK S., SCHMID C. & PONCE J. (2003a). Affine-invariant local descriptors and neighborhood statistics for texture recognition. In *Proc. ICCV*.

LAZEBNIK S., SCHMID C. & PONCE J. (2003b). Sparse texture representation using affine-invariant regions. In *Proc. CVPR*.

LINDEBERG T. & GARDING J. (1997). Shape-adapted smoothing in estimation of 3-d depth cues from affine distorsions of local 2-d brightness structure. In *Image and Vision Computing*, volume 15.

MALIK J., BELONGIE S., LEUNG T. & SHI J. (2001). Contour and texture analysis for image segmentation. In *IJCV*, volume 43(1).

NIGAM K., MCCALLUM A., THRUN S. & MITCHELL T. (2000). Text classification from labeled and unlabeled documents using EM. In *Machine Learning*, volume 39 (2/3).

ROSENFELD A., HUMMEL R. & ZUCKER S. (1976). Scene labeling by relaxation operations. In *IEEE Trans. Systems, Man, and Cybernetics*, volume 6(6).

SCHMID C. (2001). Constructing models for content-based image retrieval. In *Proc. CVPR*.

ZHANG J. (1992). The Mean Field Theory in EM Procedures for Markov Random Fields. In *IEEE trans. Signal Proc.*, volume 40(10).

Planification robuste avec (L)RTDP

Olivier Buffet et Douglas Aberdeen

National ICT Australia &
The Australian National University
{olivier.buffet,douglas.aberdeen}@nicta.com.au
http://rsise.anu.edu.au/∼{buffet,daa}

Résumé : Les problèmes de chemin le plus court stochastique (SSP : Stochastic Shortest Path problem), un sous-ensemble des problèmes de décision markoviens (MDPs), peuvent être efficacement traîtés en utilisant l'algorithme *Real-Time Dynamic Programming* (RTDP). Toutefois, les modèles des MDPs sont souvent incertains (obtenus à l'aide de statistiques ou par intuition). Une approche usuelle est alors la planification robuste : chercher la meilleure politique sous le pire modèle. Cet article montre comment RTDP peut être rendu robuste dans le cas commun où l'on sait que les probabilités de transition se trouvent dans un intervalle donné. Cela permet d'effectuer une planification en tenant compte de l'incertitude d'un modèle appris alors que les approches classiques font l'hypothèse d'un modèle "moyen".

1 Introduction

Pour la planification dans le cadre de la théorie de la décision, les problèmes de décision markoviens (Bertsekas & Tsitsiklis, 1996) sont d'un intérêt majeur quand un modèle probabilitiste du domaine est disponible. Divers algorithmes permettent de trouver un plan (une politique) optimisant l'espérance de l'utilité à long terme. Toutefois, les résultats de convergence vers la politique optimale dépendent tous de l'hypothèse que le modèle probabiliste du domaine est précis.

Malheureusement, un grand nombre de modèles de MDPs sont basés sur des probabilités (et récompenses) incertaines. Nombre d'entre elles dépendent de modèles statistiques de systèmes physiques ou naturels, tels que pour le contrôle d'usines ou l'analyse de comportements d'animaux. Ces modèles statistiques sont parfois basés sur des simulations (elles-mêmes étant des modèles mathématiques), des observations d'un système réel ou une expertise humaine.

Travailler avec des modèles incertains requiert d'abord de répondre à deux questions étroitement liées : 1– comment modéliser l'incertitude, et 2– comment utiliser le modèle résultant. Les travaux existants montrent que l'incertitude est parfois représentée à travers un ensemble de modèles possibles, à chacun étant assigné une probabilité (Munos, 2001). L'exemple le plus simple est celui d'un ensemble de modèles possibles que l'on considère d'égales probabilités (Bagnell *et al.*, 2001; Nilim & Ghaoui, 2004). Mais

plutôt que de construire un ensemble éventuellement infini de modèles, nous choisissons de représenter l'incertitude sur le modèle en définissant chaque probabilité à l'intérieur du modèle comme se trouvant dans un intervalle donné (Givan *et al.*, 2000; Hosaka *et al.*, 2001).

Les probabilités incertaines ont été étudiées dans des problèmes d'allocation de ressources pour trouver le modèle le plus adapté (Munos, 2001) :

– ressource temporelle : comment explorer efficacement (Strehl & Littman, 2004), et
– ressource spatiale : comment aggréger des états (Givan *et al.*, 2000) ;

et dans le but de trouver des politiques robustes (Bagnell *et al.*, 2001; Hosaka *et al.*, 2001; Nilim & Ghaoui, 2004). Nous nous concentrons sur ces derniers, considérant un jeu à deux joueurs où l'adversaire choisi parmi les modèles possibles celui qui dégrade le plus l'utilité à long-terme.

Notre principal objectif est de développer un planificateur efficace pour un sous-ensemble commun de MDPs pour lesquels toutes les politiques optimales ont la garantie de s'arrêter dans un état terminal : les problèmes de chemin le plus court stochastique (SSP : Stochastic Shortest Path). L'algorithme glouton *Real-Time Dynamic Programming* (RTDP) (Barto *et al.*, 1995) est particulièrement adapté aux SSPs, trouvant de bonnes politiques rapidement et ne nécessitant pas une exploration complète de l'espace d'états.

Cet article montre que RTDP peut être rendu robuste, permettant ainsi une planification plus adaptée à un modèle incertain parce qu'appris par expérimentations, voire par intuition. En section 2, nous présentons les SSPs, RTDP et la robustesse. Puis la section 3 explique comment RTDP peut être transformé en un algorithme robuste. Finalement, des expérimentations sont présentées pour analyser le comportement de l'algorithme obtenu, avant une discussion et conclusion.[1]

2 Contexte

2.1 Chemin le plus court stochastique

Un problème de chemin le plus court stochastique (Bertsekas & Tsitsiklis, 1996) est défini ici par un uplet $\langle S, s_0, G, A, T, c\rangle$. Il décrit un problème de contrôle où S est l'ensemble fini des **états** du système, $s_0 \in S$ est un état de départ, et $G \subseteq S$ est un ensemble d'états buts. A est l'ensemble fini des **actions** possibles. Les actions contrôlent les transitions d'un état s à un autre s' selon la dynamique probabiliste du système, décrite par la **fonction de transition** T définie par $T(s, a, s') = Pr(s_{t+1} = s' | s_t = s, a_t = a)$. L'objectif est d'optimiser une mesure de performance basée sur la **fonction de coût** $c : S \times A \times S \to \mathbb{R}^+$.[2]

Les SSP requièrent l'hypothèse qu'il existe une politique propre, c'est-à-dire pour laquelle un état but est accessible depuis tout état dans S, de sorte qu'il n'est pas possible de rester bloqué dans un sous-ensemble d'états. On fait de plus l'hypothèse qu'une

[1] Ce travail est présenté plus en détails dans (Buffet & Aberdeen, 2004).

[2] Le modèle n'étant pas certain, nous ne faisons pas l'hypothèse usuelle $c(s, a) = \mathbb{E}_{s'}[c(s, a, s')]$.

politique impropre conduit à un coût à long terme infini pour au moins un état. Un algorithme de résolution d'un SSP doit trouver une **politique** associant à chaque état une distribution de probabilité sur les actions $\pi : S \rightarrow \Pi(A)$ qui optimise le **coût à long terme** J défini comme l'espérance de la somme des *coûts* pour atteindre un état but.

Dans cet article, nous considérons des SSPs à des fins de planification, avec connaissance complète de l'uplet définissant le problème : $\langle S, s_0, G, A, T, c\rangle$. Dans ce cadre, des algorithmes de programmation dynamique stochastique biens connus tels que *Value Iteration* (VI) permettent de trouver une politique déterministe optimale. Value Iteration fonctionne en calculant la fonction $J^*(s)$ qui donne l'espérance de coût à long terme (finie avec l'hypothèse faite d'existence d'une politique propre) des politiques optimales. C'est le point fixe solution (unique) de l'équation de Bellman :

$$J(s) \quad = \quad \min_{a \in A} \sum_{s' \in S} T(s,a,s') \left[c(s,a,s') + J(s')\right]. \tag{1}$$

Mettre à jour J par cette formule entraîne la convergence asymptotique vers J^*. Pour des raisons pratiques, nous introduisons aussi la Q-valeur :

$$Q(s,a) = \sum_{s' \in S} T(s,a,s')[c(s,a,s') + J(s')].$$

Les SSPs peuvent facilement être vus comme des problèmes de chemin le plus court dans lesquels choisir un chemin ne mène que de manière probabiliste vers la destination espérée. Ils peuvent représenter un sous-ensemble très utile des MDPs, puisqu'il s'agit essentiellement de MDPs à horizon finis.

2.2 RTDP

L'algorithme *Trial based*[3] *Real-Time Dynamic Programming* (RTDP), introduit dans (Barto *et al.*, 1995), utilise le fait que les coûts du SSP sont positifs et l'hypothèse supplémentaire que chaque essai (parcours depuis l'état de départ) atteindra un état but avec une probabilité 1. Ainsi, avec une initialisation nulle de la fonction de coût à long terme J, J comme les Q-valeurs croissent de manière monotone durant leur calcul itératif.

L'idée derrière RTDP (algorithme 1) est de suivre des chemins depuis l'état de départ s_0 en choisissant toujours de manière gloutonne des actions associées au coût à long terme le plus bas, et en mettant à jour $Q(s,a)$ au fur et à mesure que les états s sont rencontrés. En d'autres termes, l'action choisie est celle dont on espère qu'elle mènera aux coûts futurs les plus bas, jusqu'à ce que les calculs itératifs montrent qu'une autre action semble pouvoir faire mieux.

RTDP a l'avantage de vite éviter les plans qui conduiraient à des coûts élevés. Ainsi, l'exploration regarde principalement un sous-ensemble prometteur de l'espace d'états. Toutefois, parce que l'algorithme suit les chemins en suivant la dynamique du système, les transitions rares ne sont prises en compte que rarement. L'utilisation de la simulation permet d'obtenir de bonnes politiques tôt, mais au prix d'une convergence finale lente, du fait de la mauvaise fréquence de mise à jour des transitions rares.

[3]On considèrera toujours la version *trial based* de RTDP.

Algorithme 1 Algorithme RTDP pour SSPs

RTDP(s : état) // $s = s_0$
répéter
  ESSAIRTDP(s)
jusqu'à // *pas de condition d'arrêt*

ESSAIRTDP(s : état)
tant que ¬BUT(s) **faire**
  a =ACTIONGLOUTONNE(s)
  J(s)= $Q(s, a)$
  s =CHOISIRETATSUIVANT(s, a)
fin tant que

2.3 Robust Value Iteration

Pessimisme et optimisme

Nous passons maintenant au problème de tenir compte de l'incertitude du modèle lors de la recherche d'une "meilleure" politique. L'ensemble (potentiellement infini) des modèles possibles est noté $\mathcal{M}$.

Une approche simpliste est de calculer le modèle moyen sur $\mathcal{M}$, ou le modèle le plus probable, puis d'utiliser des méthodes d'optimisation standard pour SSPs. De telles approches ne garantissent rien sur le coût à long terme de la politique si le vrai modèle diffère de celui choisi pour l'optimisation.

Nous suivons l'approche décrite dans (Bagnell *et al.*, 2001), laquelle consiste à trouver une politique se comportant bien face au pire modèle possible. Cela revient à considérer un jeu à deux joueurs et à somme nulle, i.e. où le gain d'un joueur est la perte de l'autre. Le joueur choisit une politique sur les actions (dans l'espace de politiques stochastiques Π_A) alors que son adversaire "perturbateur" choisit simultanément une politique sur les modèles (dans l'espace $\Pi_{\mathcal{M}}$). Comme c'est un jeu simultané, les politiques optimales peuvent être stochastiques. Cela mène à un algorithme de type max-min :[4]

$$\max_{\pi_{\mathcal{M}} \in \Pi_{\mathcal{M}}} \min_{\pi_A \in \Pi_A} J_{\pi_{\mathcal{M}}, \pi_A}(s_0).$$

Dans ce jeu SSP, Value Iteration converge vers une solution fixe (Patek & Bertsekas, 1999).

Il est aussi possible d'être optimiste, considérant que les deux joueurs collaborent (du fait qu'ils endurent les mêmes coûts), ce qui transforme le $\max$ en un $\min$ dans la formule précédente. Ce second cas est équivalent à un SSP classique où une décision consiste en le choix d'une action **et** d'un modèle local.

[4] Le jeu étant simultané, l'ordre entre $\max$ et $\min$ est sans importance.

Localité

Un tel algorithme max-min serait particulièrement coûteux à implémenter. Même en restreignant la recherche à une politique déterministe sur les modèles, il faudrait calculer la fonction de coût à long terme optimale pour chaque modèle avant de choisir le pire modèle et la politique optimale associée. Toutefois, un processus plus simple peut être utilisé pour calculer J en cherchant en même temps le pire modèle. Il faut pour cela faire l'hypothèse que les distributions $T(s, a, \cdot)$ sont indépendantes d'une paire état-action (s, a) à l'autre. Cette hypothèse n'est pas toujours valide, mais rend les choses plus faciles pour l'adversaire puisqu'il peut ainsi choisir à travers un ensemble de modèles élargi. On ne risque alors que d'avoir des politiques "trop robustes" (parce que trop pessimistes).

Parce que nous faisons l'hypothèse d'une indépendance au niveau "état-action" (pas seulement au niveau "état"), c'est équivalent à une situation où le second joueur prend une décision dépendant de l'état courant et de l'action du premier joueur. Cette situation revient à un jeu *séquentiel* où le mouvement du joueur précédent est connu du joueur suivant : les deux joueurs peuvent agir de manière déterministe sans perte d'efficacité.

Le résultat de cette hypothèse est que le pire modèle peut être choisi localement quand Q est mis à jour pour une paire état-action donnée. Comme on peut le voir sur l'algorithme 2, le pire modèle local m_s^a peut changer pendant que les Q-valeurs évoluent. De précédentes mises à jour des coûts à long terme d'états atteignables peuvent changer leur ordre relatif, de sorte que les résultats considérés comme les plus mauvais ne sont pas les mêmes.

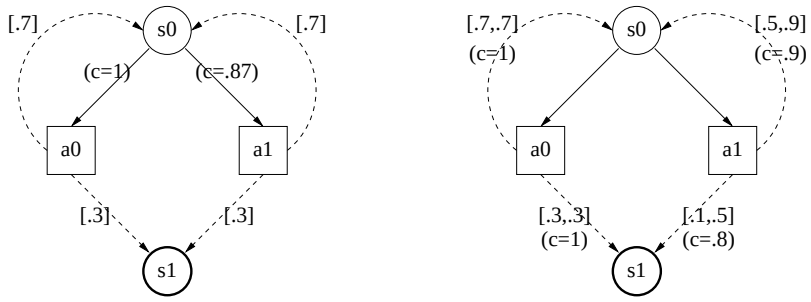
Algorithme 2 Robust Value Iteration (pour un SSP)

Initialiser J to 0.
répéter
  pour tout s : état **faire**
    pour tout a : action **faire**
      $Q_{\max}(s, a) \leftarrow \max_{m_s^a \in \mathcal{M}_s^a} \sum_{s' \in S} T_{m_s^a}(s, a, s') \left[J(s') + c_{m_s^a}(s, a, s') \right]$
    fin pour
    $J(i) \leftarrow \min_{a \in A} Q_{\max}(s, a)$
  fin pour
jusqu'à J converge

L'apport principale de cet article est de montrer que RTDP peut être rendu *robuste*, permettant la planification dans des domaines très grands et incertains, en assurant le comportement dans le pire cas.

3 Robust RTDP

Nous considérons désormais des SSPs incertains **basés sur des intervalles**, où $T(s, a, s')$ se trouve dans un intervalle $[Pr^{\min}(s'|s, a), Pr^{\max}(s'|s, a)]$. La figure 1 montre un exemple d'un tel SSP. Nous discutons l'approche pessimiste, l'optimiste amenant à des résultats similaires.



a) SSP certain b) SSP incertain

FIG. 1 – Deux vues d'un même SSP, selon que l'incertitude sur le modèle est prise en compte (coûts entre parenthèses). Dans le SSP incertain, l'action a_0 sera préférée du fait qu'elle atteint rapidement le but s_1.

Pour une paire état-action donnée (s, a), il existe une liste $R = (s'_1, \cdots, s'_k)$ d'états atteignables (R est choisi pour "reachable"). Pour chaque état atteignable, $T(s, a, s'_i) \in I_i = [p_i^{\min}, p_i^{\max}]$. Ainsi, les modèles possibles sont ceux qui respectent les contraintes représentées par ces intervalles tout en assurant $\sum_i T(s, a, s'_i) = 1$. La figure 2 illustre ceci avec trois états atteignables.

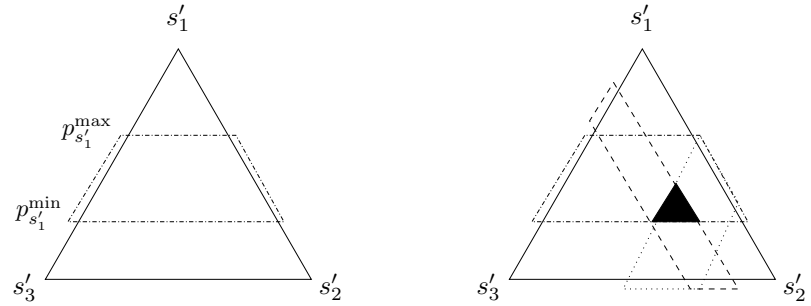


FIG. 2 – Un triangle est un simplexe représentant toutes les distributions de probabilité possibles pour trois résultats différents ($Pr(s'_i) = 1$ au sommet s'_i). Sur le triangle de gauche, le trapèze montre l'intervalle-contrainte pour s'_1. Le triangle de droite montre les modèles possibles à l'intersection des trois intervalles-contraintes.

Pires modèles locaux —

L'étape de maximisation pour calculer $Q(s, a)$ dans l'algorithme 2 est effectuée en donnant la plus grande probabilité au pire résultat. Ceci requiert d'abord d'ordonner les états atteignables de manière décroissante selon les valeurs : $c(s, a, s'_1) + J(s'_1) \geq c(s, a, s'_2) + J(s'_2) \geq \cdots c(s, a, s'_k) + J(s'_k)$. Après, la pire distribution est celle associant la plus grand probabilité au premier état s'_1, puis à s'_2, et ainsi de suite jusqu'à s'_k. Comme indiqué dans (Givan *et al.*, 2000), il est équivalent de trouver l'index $r \in [1..k]$

tel que

$$\sum_{i=1}^{r-1} p_i^{\max} + \sum_{i=r}^{k} p_i^{\min} \leq 1.$$

Les transitions de probabilités résultantes sont alors :

$$Pr(s'_i) = \begin{cases} p_i^{\max} \text{ if } i < r \\ p_i^{\min} \text{ if } i > r \end{cases} \quad (2)$$

$$Pr(s'_r) = 1 - \sum_{i=1, i\neq r}^{k} Pr(s'_i). \quad (3)$$

En utilisant la borne pré-calculée $B_{\min} = \sum_{i=1}^{k} p_i^{\min}$, l'algorithme 3 donne une implémentation complète. L'algorithme de *tri par insertion*[5] est choisi pour profiter de ce que la liste sera souvent déjà ordonnée.

Algorithme 3 Pire modèle pour la paire état-action (s, a)

PIREMODÈLE(s : état, a : action)
$R = (s'_1, \cdots, s'_k) =$ ETATSATTEIGNABLES(s,a)
TRI(R)
$i = 1$, $borne = B_{\min}$
tant que $(borne - p_i^{\min} + p_i^{\max} < 1)$ **faire**
  $borne \leftarrow borne - p_i^{\min} + p_i^{\max}$
  $Pr(s'_i) \leftarrow p_i^{\max}$
  $i \leftarrow i + 1$
fin tant que
$r = i$
$Pr(s'_r) \leftarrow 1 - (borne - p_r^{\min})$
pour tout $i \in \{r + 1, \ldots, k\}$ **faire**
  $Pr(s'_i) \leftarrow p_i^{\min}$
fin pour
return $(R, Pr(\cdot))$

En résumé, Robust VI sur un SSP basé sur des intervalles consiste à appliquer Value Iteration tout en mettant à jour les probabilités de transition à travers l'algorithme 3.

Nous n'avons besoin que d'un seul pire modèle pour calculer les pires Q-valeurs. Toutefois, parce que plusieurs états atteignables s'_i peuvent avoir la même valeur $c(s, a, s'_i) + J(s'_i)$ que s'_r (on note cet ensemble d'états S'_r), il peut y avoir une infinité de pire modèles locaux équivalents. Tout modèle ne différant que par la distribution de la masse de probabilité parmi les états également mauvais de S'_r est aussi un pire modèle local.

Pires modèles globaux —

Contrairement à VI, RTDP ne visite pas nécessairement tout l'espace d'états. C'est pourquoi (Barto *et al.*, 1995) introduit la notion d'état *pertinent* ("relevant state"), que

[5] `http ://en.wikipedia.org/wiki/Insertion_sort`

nous étendons au cas incertain : un état s est dit être *pertinent* pour $\mathcal{M}$ s'il existe un état de départ s_0, un modèle $m \in \mathcal{M}$ et une politique optimale π sous ce modèle tels que s peut être atteint de l'état s_0 quand le contrôleur utilise cette politique sous ce modèle.

Cette notion est importante parce que deux modèles locaux également mauvais sur une paire état-action peuvent interdire l'accès à différents états, de sorte que pour deux modèles m_1 et m_2, un état peut être pertinent (dans le sens de (Barto *et al.*, 1995)) dans m_1 mais pas dans m_2. Mais RTDP ne devrait pas trouver une politique optimale juste pour les états pertinents d'un seul pire modèle global. Et la politique ne doit pas s'appliquer à tous les états possibles. Elle devrait s'appliquer à tous les états *atteignables* sous *tout* modèle (pour des politiques optimales), i.e. à tous les états pertinents. Mais couvrir les états pertinents du pire modèle utilisé pour ré-évaluer les Q-valeurs ne couvre pas nécessairement tous les états pertinents pour $\mathcal{M}$: cela dépend du modèle utilisé pour choisir l'état suivant, c'est-à-dire pour simuler la dynamique du système.

Pour éviter de manquer des états pertinents, chaque modèle local utilisé pour la simulation doit assurer que tout état atteignable (d'après $\mathcal{M}$) peut être visité. Comme on peut le voir sur la figure 2, l'ensemble des modèles locaux possibles pour une paire état-action est un polytope convexe à n dimensions. Tout modèle *à l'intérieur* de ce polytope, excluant la frontière, est ainsi approprié puisque, pour tout s'_i, il garantie que $P(s'_i|s,a) > 0$.

Ainsi il existe un modèle global m_d qui peut être employé pour simuler la dynamique du système sans manquer quelque état potentiellement atteignable qu'il soit.

3.1 Robust (Trial-Based) RTDP

Robust RTDP diffère du RTDP original en ce que :

- A chaque fois que l'algorithme met à jour l'évaluation d'un état, l'adversaire cherche le pire modèle local, utilisé pour calculer les Q-valeurs.
- Pour l'exploration, l'algorithme suit une dynamique possible du système qui tient compte de toutes les transitions possibles (utilisant le modèle m_d).
- Les états "pertinents" sont maintenant les états atteignables en suivant une politique optimale sous n'importe quel modèle possible.

De là, nous pouvons adapter à notre contexte le théorème de convergence 2 de (Barto *et al.*, 1995), ainsi que la preuve correspondante, en discutant principalement les modifications qu'elle requiert.

Théorème 1
Dans des problèmes de chemin le plus court stochastique **incertain** *et avec atténuation,* **robust** *Trial-Based RTDP, avec l'état initial de chaque essai restreint à un ensemble d'états de départ, converge (avec probabilité un) vers J^* sur l'ensemble des états pertinents, et la politique optimale du contrôleur converge vers une politique optimale (éventuellement non-stationnaire) sur l'ensemble des états pertinents, sous les mêmes conditions que le théorème 3 dans (Barto* et al.*, 1995).*

Preuve :

La preuve dans (Barto et al.*, 1995) montre que les états indéfiniment mis à jour par RTDP sont les états pertinents, de sorte qu'une preuve de convergence classique sur les SSP peut être invoquée.*

Une première remarque est qu'introduire $\max_{m \in \mathcal{M}}$ *dans la formule de mise à jour ne change pas le fait que* J_t *est croissante et non-surestimante.*

Dans notre cas, l'utilisation du modèle m_d *assure de manière similaire que les états indéfiniment mis à jour par robust RTDP sont tous les états pertinents (du SSP incertain).*

Nous avons établi que nous sommes dans un jeu séquentiel de type chemin le plus court ("Stochastic Shortest Path Games"=SSPG). (Bertsekas & Tsitsiklis, 1996) montre qu'il s'agit de cas particuliers de SSPG général (simultané). La convergence pour les SSPGs généraux est prouvée dans la proposition 4.6 de (Patek & Bertsekas, 1999), laquelle établie que les coûts à long terme convergent avec une probabilité 1 *sur l'ensemble des états pertinents.* □

Quel que soit le modèle réel, l'algorithme apprend toutes les décisions optimales pour tout état pertinent sous l'hypothèse la plus pessimiste. Un état pertinent s peut d'ailleurs ne pas être atteignable à travers un pire modèle global, mais l'environnement réel peut y mener. Ainsi la politique doit couvrir tous les états pertinents mais fait l'hypothèse que le pire modèle s'applique depuis ces états.

4 Expérimentations

Labelled RTDP (Bonet & Geffner, 2003) est une version modifiée de RTDP qui peut être rendue robuste de manière similaire. Les expériences effectuées illustrent le comportement de *robust LRTDP*. Dans ce but, il est comparé au *Robust Value Iteration* de Bagnell, ainsi qu'à *LRTDP*. Dans tous les cas, le critère de convergence est $\epsilon = 10^{-3}$:

- pour LRTDP, un état s a convergé si ses enfants aussi et si son résidu $|J_{t+1}(s) - J_t(s)|$ est plus petit que ϵ, et
- pour VI, nous nous arrêtons quand le plus grand changement dans le coût à long terme d'un état au court d'une itération est plus petit que ϵ.

4.1 Cœur

Dans cette première expérimentation, nous comparons une politique non-robuste optimale avec une robuste sur le petit exemple de la figure 1-b. Le tableau 1 montre les coûts à long terme espérés théoriques de chaque politique sur le modèle normal (plus probable), ainsi que sur les modèles pessimistes et optimistes. La politique robuste est largement meilleure dans le cas pessimiste. On a ici un exemple caricatural du fait qu'une politique robuste fait usage de transitions moins incertaines qu'une politique qui est optimale pour le modèle le plus probable, d'où une moins grande variabilité de son efficacité quand elle est évaluée sur divers modèles.

TAB. 1 – Evaluation théorique de politiques robustes et non-robustes sur divers modèles, concordant exactement à l'évaluation empirique.

	Normal	Pessimiste	Optimiste
Non-robuste	2.90	8.90	1.70
Robuste	3.33	3.33	3.33

4.2 La voiture sur la montagne

Nous employons ici le problème de la voiture sur la montagne tel que défini dans (Sutton & Barto, 1998) : partant du fond d'une vallée, une voiture doit acquérir assez d'élan pour atteindre le haut d'une montagne (voir figure 3). La dynamique utilisée est la même que décrite dans le logiciel "mountain car".[6] L'objectif est de minimiser le nombre de pas de temps pour atteindre le but.

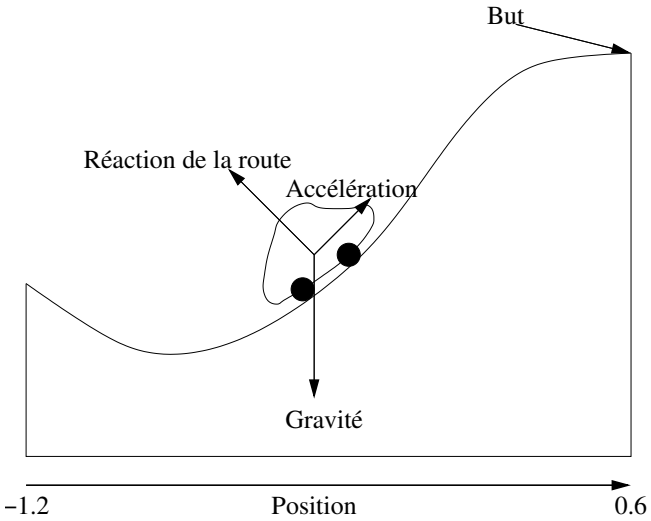


FIG. 3 – Le problème de la voiture sur la montagne.

L'espace d'état continu est discrétisé (grille 32×32) et le modèle de transitions incertaines correspondant est obtenu par échantillonage de 1000 transitions depuis chaque paire état-action (s, a). Pour chaque transition, on calcule les intervalles dans lesquels se trouve le vrai modèle avec une confiance de 95%. Ce calcul décrit en annexe B dans (Buffet & Aberdeen, 2004) utilise la variance empirique et assure que le modèle le plus probable satisfait les contraintes obtenues (on a donc toujours au moins un modèle possible).

[6] `http ://www.cs.ualberta.ca/~sutton/MountainCar/MountainCar.html`

Résultats

Remarque préliminaire : simuler un chemin montre généralement une voiture oscillant plusieurs fois avant de quitter la vallée. Ceci a deux explications principales : 1- la vitesse acquise est juste suffisante pour atteindre le sommet, et 2- le modèle discrétisé n'est pas assez précis : appliquer la politique obtenue sur le vrai modèle mathématique (au lieu du discrétisé) devrait donner de bien meilleurs résultats.

La figure 4 montre la fonction de coût à long terme obtenue en utilisant *value iteration*, LRTDP et leur contreparties robustes sur le problème de la voiture sur la montagne. Les axes x et y donnent la position et la vitesse de la voiture. L'axe z est l'espérance du coût jusqu'au but. Sur la surface est représenté un exemple de chemin depuis l'état de départ jusqu'à l'état but : il suit la politique gloutonne face au modèle le plus probable.

La forme générale de la surface obtenue est toujours la même, avec des parties de l'espace d'états inexplorées par LRTDP et Robust LRTDP (comme attendu). Les échelles verticales sont bien plus grandes dans les cas robustes. Cela reflète le fait qu'atteindre le but consomme bien plus de temps sous un modèle pessimiste. Parce que J peut ici être interprêté comme le temps moyen avant d'atteindre le but, ces graphes montrent comment l'accumulation de petites incertitudes peut amener à des politiques plus longues. Ici les temps sont multipliés par plus de 2.5.

Lors de l'exécution des quatre différents algorithmes, une évaluation de la politique gloutonne courante était effectuée tous les $10 * nStates = 10\,240$ mises à jour d'une Q-valeur. Les résultats apparaissent sur les figures 5 a) et b), l'axe des ordonnées donnant l'espérance de coût à long terme depuis l'état de départ. Sur les deux sous-figures, les algorithmes basés sur LRTDP obtiennent de bonnes politiques rapidement, mais ont de longs temps de convergence de : VI=2.46×10^6 mises à jour, LRTDP=9.00×10^6, rVI=8.09×10^6, rLRTDP=11.5×10^6.

Une dernière mesure intéressante à observer est la "Value-at-Risk" (VaR), même si l'on cherche ici un comportement optimal face au pire modèle et non un comportement minimisant le risque. La VaR donne, pour un seuil de "risque" $r \in [0, 1]$, le coût à long terme J' tel que $Pr(J > J') \leq r$. La figure 6 a) montre les courbes de Value-at-Risk pour trois modèles possibles (moyen, bon et mauvais) et pour une politique optimale normale et une politique optimale robuste (d'où un total de 6 courbes). Dans ce cas précis, les courbes de l'une et l'autre politiques se superposent, leurs comportements étant identiques sur chacun des trois modèles. Il semble donc que la politique optimale normale soit déjà robuste, et qu'aucun effet particulier sur le risque n'est observable.

4.3 Navigation maritime

Le problème de navigation utilisé ici partage des similarités avec la voiture sur la montagne. Sa description complète peut être trouvée dans (Vanderbei, 1996), et une autre utilisation se trouve dans (Peret & Garcia, 2004). Ici, l'espace est discrétisé en une grille de 10×10, $\times 8$ angles de vent et $\times 8$ directions possibles. L'incertitude de ce système est due à l'évolution stochastique de la direction du vent. Le modèle incertain est aussi appris en tirant 1000 échantillons au hasard pour chaque paire état-action, en utilisant la même confiance de 95%.

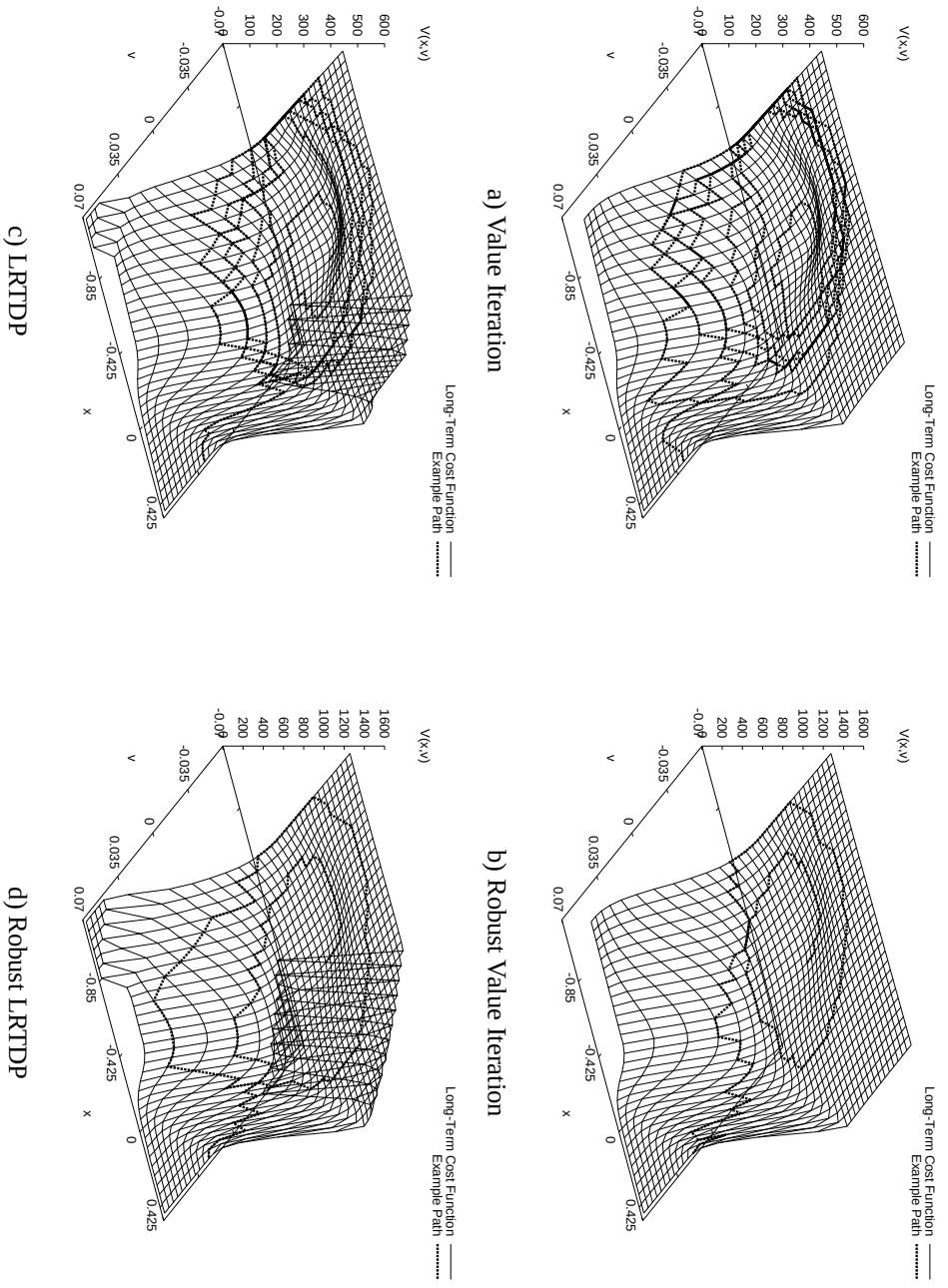


FIG. 4 – Fonction de coût à long terme pour le problème de la voiture sur la montagne Dans tous les cas, le modèle le plus probable est utilisé pour générer un exemple de chemin.

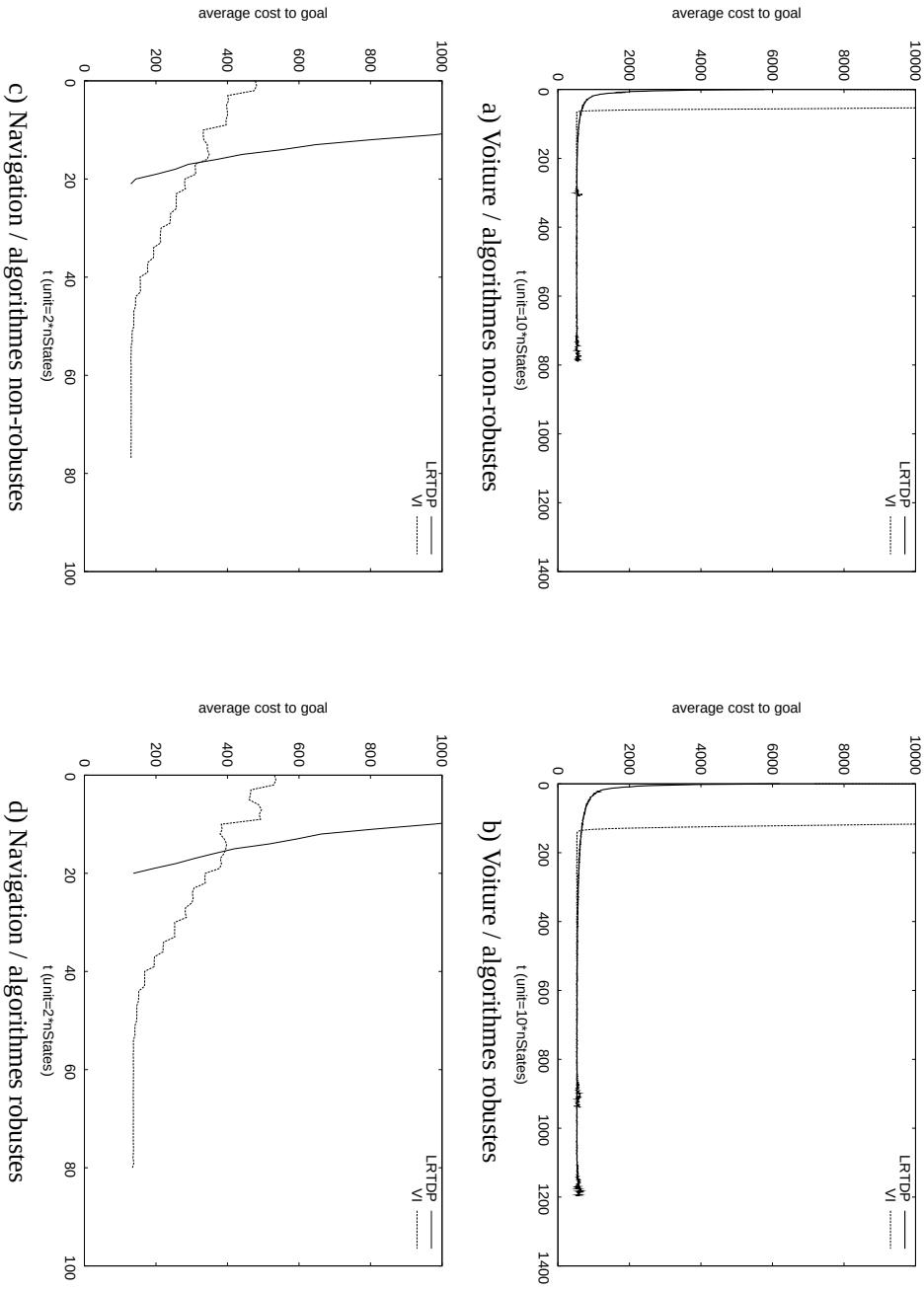


FIG. 5 – Coût moyen pour atteindre le but pour le problème de la voiture sur la montagne (et le problème de navigation), coût mesuré toutes les $10 * nStates$ (resp. $2 * nStates$) mises à jour de Q-valeurs.

Résultats

Les mêmes tests ont été effectués que sur le problème de la voiture sur la montagne. Les fonctions de coût à long terme obtenues montrent des phénomènes similaires tels que l'augmentation du temps pour atteindre le but. Seules les figures 5 c) et d) sont d'un intérêt particulier, puisqu'elles montrent combien les algorithmes de la famille LRTDP convergent vite. Dans ce problème au plus grand nombre de dimensions, trouver des solutions prend plus de temps au début, mais LRTDP s'avère très efficace pour éliminer les chemins inefficaces. En fait, la plupart des états pertinents se trouvent le long de la diagonale principale du lac (la plupart des états latéraux peuvent être évités par les politiques optimales). Pour les différents algorithmes, le temps de convergence est : VI=3.67×10^6, LRTDP=0.49×10^6, rVI=5.22×10^6, rLRTDP=0.60×10^6.

Pour finir, la figure 6 b) donne les courbes de Value-at-Risk pour trois modèles possibles (moyen, bon et mauvais) et pour une politique optimale normale et une politique optimale robuste. Contrairement au problème de la voiture sur la montagne, on observe un comportement différent selon que la politique est robuste ou non. En fait, dans le cas de la politique robuste, les 3 courbes correspondants aux 3 différents modèles employés sont presque confondues (avec la courbe du milieu : "politique optimale normale / modèle moyen"). On en déduit que la recherche d'une politique robuste amène à des prises de décisions différentes "uniformisant" la prise de risque : la probabilité de dépasser un certain coût à long terme est la même quel que soit le modèle réel du système.

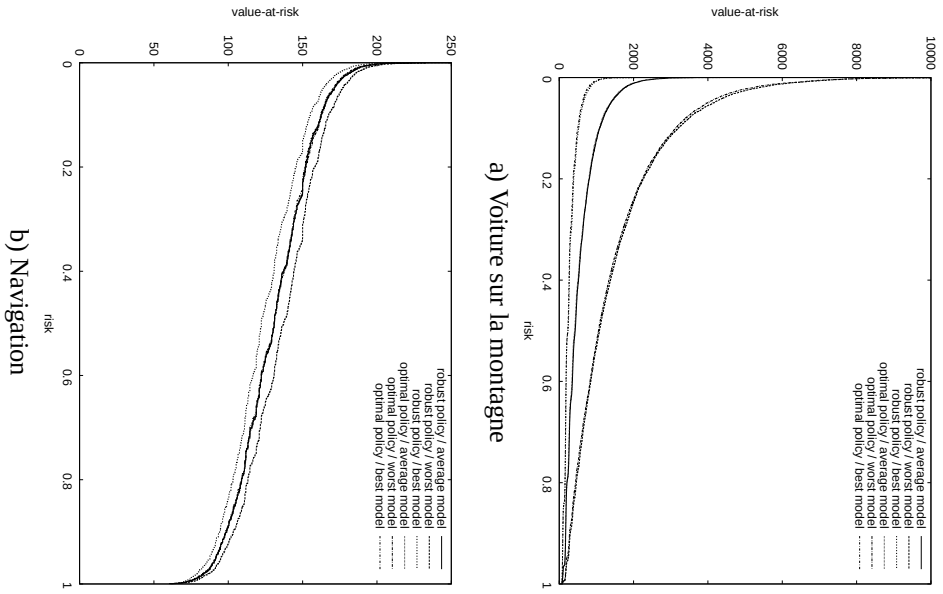


FIG. 6 – Courbes de Value-at-Risk pour les deux problèmes considérés et 3 modèles de référence.

Une autre expérimentation (Buffet & Aberdeen, 2004) confirme ces résultats sur un

exemple illustrant cette approche sur un problème de planification temporelle. Dans ce cas, l'incertitude vient de ce que les probabilités d'échec des différentes tâches n'est connue que par consultation d'experts du domaine (Aberdeen *et al.*, 2004).

5 Discussion et conclusion

Une extension directe de ce travail, suggérée par (Hosaka *et al.*, 2001), est de trouver les meilleures décisions face aux pires modèles (comme nous le faisons dans cet article), puis de choisir parmi celles-ci les décisions optimales pour un *modèle optimiste*. Cette idée a été développée dans l'annexe C de (Buffet & Aberdeen, 2004). Si les calculs supplémentaires requis sont raisonnables, ils ne sont utiles que si diverses politiques robustes équivalentes existent, avec une variabilité de leurs résultats sur des modèles optimistes.

L'approche de la robustesse adoptée dans ce papier considère que l'on connait un ensemble de modèles possibles. Une question ouverte est de savoir s'il est possible d'utiliser plus d'information issue du modèle incertain en prenant en compte la distribution de probabilité sur les modèles possibles.

De manière similaire, l'incertitude sur le modèle a été considérée pour apprendre un modèle pendant la planification (Strehl & Littman, 2004). L'algorithme proposé est optimiste, mais ne semble pas s'adapter à notre cadre dans la mesure où l'évolution du modèle brise l'hypothèse de "non-surestimation" : $\forall s \in S,\ t \geq 0,\ J_t(s) \leq J^*(s)$. Il reste toutefois important de noter que *robust* RTDP ne souffrirait pas d'être utilisé en ligne, puisque la dynamique réelle peut être employée pour choisir l'état suivant (le pire modèle n'apparaît que dans la formule de mise à jour du coût à long terme).

Enfin, une hypothèse cruciale de RTDP est qu'un état but doit être atteignable depuis tout état. Nous présentons un algorithme répondant à ce problème dans (Buffet, 2004).

Conclusion

Des travaux récents montrent que l'incertitude du modèle est un problème important pour la planification dans le cadre de la théorie de la décision. Il peut être intéressant aussi bien d'analyser le modèle pour savoir où il pourrait être raffiné, que prendre des décisions en tenant compte de l'incertitude connue. Nous avons proposé une modification de l'algorithme RTDP lui permettant de calculer des politiques robustes efficacement dans des domaines de grande taille et incertains. L'incertitude sur le modèle est représentée à travers des intervals de confiance sur les probabilités de transition. La preuve de convergence de l'algorithme résultant est esquissée (détails dans (Buffet & Aberdeen, 2004)). Nous faisons la démonstration de robust LRTDP sur un domaine où les intervalles sont estimés de manière statistique.

Remerciements

Grand merci à Sylvie Thiébaux pour son aide et ses encouragements.

Le National ICT Australia est financé par le gouvernement australien. Ce travail a aussi bénéficié du soutien du DSTO (Australian Defence Science and Technology Organisation).

Références

ABERDEEN D., THIÉBAUX S. & ZHANG L. (2004). Decision-theoretic military operations planning. In *Proceedings of the Fourteenth International Conference on Automated Planning and Scheduling (ICAPS'04)*.

BAGNELL J., NG A. Y. & SCHNEIDER J. (2001). *Solving Uncertain Markov Decision Problems*. Rapport interne CMU-RI-TR-01-25, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

BARTO A., BRADTKE S. & SINGH S. (1995). Learning to act using real-time dynamic programming. *Artificial Intelligence*, **72**.

BERTSEKAS D. & TSITSIKLIS J. (1996). *Neurodynamic Programming*. Athena Scientific.

BONET B. & GEFFNER H. (2003). Labeled rtdp : Improving the convergence of real time dynamic programming. In *Proceedings of the Thirteenth International Conference on Automated Planning and Scheduling (ICAPS'03)*.

BUFFET O. (2004). *Robust (L)RTDP : Reachability Analysis*. Rapport interne, National ICT Australia.

BUFFET O. & ABERDEEN D. (2004). *Planning with Robust (L)RTDP*. Rapport interne, National ICT Australia.

GIVAN R., LEACH S. & DEAN T. (2000). Bounded parameter markov decision processes. *Artificial Intelligence*, **122**(1-2), 71–109.

HOSAKA M., HORIGUCHI M. & KURANO M. (2001). Controlled markov set-chains under average criteria. *Applied Mathematics and Computation*, **120**(1-3), 195–209.

MUNOS R. (2001). Efficient resources allocation for markov decision processes. In *Advances in Neural Information Processing Systems 13 (NIPS'01)*.

NILIM A. & GHAOUI L. E. (2004). Robustness in markov decision problems with uncertain transition matrices. In *Advances in Neural Information Processing Systems 16 (NIPS'03)*.

PATEK S. D. & BERTSEKAS D. P. (1999). Stochastic shortest path games. *SIAM J. on Control and Optimization*, **36**, 804–824.

PERET L. & GARCIA F. (2004). On-line search for solving markov decision processes via heuristic sampling. In *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI'2004)*.

STREHL A. L. & LITTMAN M. L. (2004). An empirical evaluation of interval estimation for markov decision processes. In *Proceedings of the Sixteenth International Conference on Tools with Artificial Intelligence (ICTAI'04)*.

SUTTON R. & BARTO G. (1998). *Reinforcement Learning : an introduction*. Bradford Book, MIT Press, Cambridge, MA.

VANDERBEI R. J. (1996). Optimal sailing strategies, statistics and operations research program. University of Princeton, http ://www.sor.princeton.edu/˜rvdb/sail/sail.html.

Réseau Bayésien Aplati pour l'Inférence dans les HMM hiérarchiques factorisés et Apprentissage avec peu de données[1]

Sylvain Gelly*, Nicolas Bredeche*, Michèle Sebag*

*Équipe Inférence&Apprentissage - Projet TAO (INRIA futurs)
LRI - Université Paris-Sud, 91405 Orsay Cedex
(gelly,bredeche,sebag)@lri.fr - http ://tao.lri.fr

Une limite essentielle des HMM, et plus généralement des modèles de Markov, concerne le passage à l'échelle, l'impossibilité de la prise en compte efficace de l'influence de phénomènes indépendants et la difficulté de généralisation.

Pour répondre à ces problèmes, plusieurs extensions existent. En particulier, nous nous intéresserons dans ce qui suit à la *hiérarchisation* (Shai Fine, 1998; Theocharous *et al.*, 2004) et à la *factorisation* (Ghahramani & Jordan, 1997).

La hiérarchisation permet de réduire le nombre de liens entre états nécessaires dans un HMM et par là même de réduire la complexité algorithmique de l'apprentissage ainsi que l'imprécision. Quant à la factorisation, le principe est d'expliquer les observations par plusieurs causes plutôt qu'une seule. C'est-à-dire qu'on remplace le $P(Y|X)$ des HMM par $P(Y|X^1, X^2, ..., X^n)$. Les X^i sont des variables cachées pouvant être gérées indépendamment. Les $P(X^i_{t+1}|X^i_t)$ sont alors différents pour chaque i.

En pratique, on ne peut pas adapter directement les algorithmes d'inférence existants dans le cas des HMM factorisés, ou hiérarchiques. De plus, un aspect important du problème est que notre système apprend à partir de données éparses car nous faisons l'hypothèse que nous ne disposons que d'un petit nombre d'exemples pour apprendre. Ceci se justifie par notre domaine d'application (la robotique située), où le processus d'échantillonnage des données est contrôlé par un comportement dépendant entre autres de l'environnement et des capacités du robot qui ne permet pas d'obtenir beaucoup d'exemples. Par conséquent, nous souhaitons exprimer un compromis entre précision et vitesse de l'apprentissage.

L'approche que nous proposons consiste à changer de formalisme de représentation en transformant un graphe orienté et avec circuits (i.e. HMM hiérarchique factorisé dans lequel on fait abstraction du typage des dépendances) en un réseau Bayésien. Le formalisme des réseaux Bayésiens s'inscrit en effet dans un cadre théorique développé et bien connu qui laisse espérer une résolution plus facile. Toutefois, nous identifions les deux problèmes suivants :

- L'existence de dépendances multiples dans les FHHMM entraîne à priori une explosion combinatoire du nombre de paramètres à apprendre, ce qui est d'autant plus problématique lorsque peu d'exemples sont à notre disposition ;
- La présence de circuits dans les dépendances conditionnelles entre les variables d'un FHHMM empêchent la modélisation directe par un réseau Bayésien. Il est à noter que ces dépendances ne concernent les variables qu'à un même pas de temps (synchrones).

[1]Acknowledgements : This work was supported in part by the PASCAL Network of Excellence.

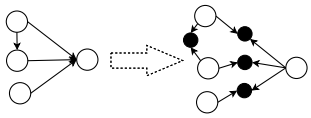


FIG. 1 – Exemple de changement de représentation (RB $=>$ RBA). Le graphe de gauche donne la structure des dépendances conditionnelles entre les variables, et le graphe de droite montre l'expression dans le formalisme des réseaux Bayésien après transformation et ajout de variables additionnelles booléennes. Ces variables additionnelles permettent de forcer la dépendance entre deux variables du réseau Bayésien (elles sont systématiquement observées à $vrai$). Les probabilités conditionnelles de ces variables sont calculées de telle sorte que les probabilités jointes entre deux variables d'origine soient cohérentes avec les données.

En conséquence, nous avons mis au point un algorithme de changement de représentation permettant de construire ce que nous appelons des réseaux Bayésiens aplatis (RBA). Cette modélisation est basée sur un compromis entre précision et vitesse d'apprentissage et repose sur la prise en compte des dépendances multiples en les exprimant deux à deux seulement (cf. Fig. 1).

Les résultats obtenus sont prometteurs puisqu'ils montrent que les réseaux Bayésiens aplatis ont les propriétés suivantes :

- prise en compte et modélisation des circuits, ceux-ci étant fréquents dans les HMM considérés ;
- apprentissage plus rapide avec peu d'exemples, au prix, il est vrai, d'une perte de précision à long terme. En robotique, ce compromis est avantageux puisque l'on dispose souvent d'exemples peu nombreux.

Cependant, le formalisme final étant toujours celui des RB (avec des variables additionnelles et un calcul des paramètres satisfaisant des axiomes), nous pouvons très bien envisager des représentations "hybrides" dans lesquelles certaines dépendances (pour lesquelles peu de données sont disponibles) sont exprimées dans le cadre des RBA, tandis que d'autres sont exprimées de façon classique. Ainsi, cette méthode permettrait de tirer parti du compromis expressivité/apprentissage avec peu d'exemples des RBA tout en gardant une expressivité pouvant atteindre celle des RB lorsque le nombre d'exemples disponibles croît suffisamment.

Références

GHAHRAMANI Z. & JORDAN M. I. (1997). Factorial hidden markov models. *Machine Learning, vol. 29. 1996, pages 245-273.*

SHAI FINE, YORAM SINGER N. T. (1998). The hierarchical hidden markov model : Analysis and applications. *Machine Learning.*

THEOCHAROUS G., MURPHY K. & KAELBLING L. (2004). Representing hierarchical pomdps as dbns for multi-scale robot localization. *Proc. of the IEEE international Conference on Robotics and Automation (ICRA'04).*

Systèmes inductifs-déductifs : une approche statistique

Nicolas Baskiotis, Michèle Sebag, Olivier Teytaud

Equipe TAO, CNRS - Université Paris-Sud (LRI) - INRIA (Futurs),
bât 490 Universite Paris-Sud 91405 Orsay Cedex France
{nbaskiot, sebag, teytaud}@lri.fr

Résumé : Les théorèmes d'essentielle indécidabilité de l'arithmétique ont souvent été cités comme limites à la démonstration automatique ou aux systèmes experts. Toutefois, ces résultats considèrent l'impossibilité d'établir la démonstration d'énoncés *au pire cas* sur le choix de l'énoncé, ce qui est trop pessimiste pour la vie réelle. C'est pourquoi nous proposons un cadre probabiliste. Précisément, nous examinons le taux d'énoncés non-décidables à mesure que le système d'axiomes s'enrichit, en particulier selon que l'on empile simplement les exemples comme autant d'axiomes ou selon que l'on effectue une réelle induction de système d'axiomes[1].

Inspired by the "Learning to Reason" framework and the debate about Quine's underdetermination thesis, this paper investigates the conditions for a hybrid inductive-deductive system (IDS). This system is provided with a set of axioms or statements (e.g. examples), and its goal is to determine the truth value of any further statement e.

From a mathematical logic perspective, the question is whether i) the available set of statements is complete, and ii) the logical setting is complete (decidable logic). The truth value of e is determined using mathematical deduction ; the algorithmic challenge is to provide an efficient search engine for constructing a proof of e or $\neg e$.

When the set of statements is not complete, inductive reasoning is needed to find additional axioms, consistent with the available ones and sufficient for determining the truth value of e. The challenge here is to determine the statistically relevant level of generalization.

From a hybrid inductive-deductive perspective, the logical setting considered must thus be examined with respect to both its completeness (deduction-oriented performances), and its VC-dimension or PAC learnability (induction-oriented performances).

Typically, statements $C(1)$, $C(3)$, $C(5)$, $\neg C(4)$, $\neg C(6)$, $\neg C(2)$, do not allow for any further deduction. In the meanwhile, inductive logic programming might hypothesize $\forall n; C(2n+1) \wedge \neg C(2n)$, which could in turn allow for many other deductions[2].

[1] Voir `http://www.lri.fr/~teytaud/decid.pdf` pour la bibliographie et les démonstrations complètes.

[2] It must be emphasized that this combination of induction and deduction corresponds to the standard reasoning in mathematics : modifying, adding and removing axioms in order to avoid inconsistencies and provide

This paper examines the convergence properties of an inductive-deductive system (e.g. the probability that the $n+1$-th example can be proved from the axioms learned from the previous n examples). The originality of the work is to propose a probabilistic analysis of logic decidability and completeness, contrasting with the worst-case analysis and undecidability results used in the literature. Indeed, a worst-case perspective does not account for the fact that many statements can yet be proved in an essentially undecidable setting.

Overview

The alternative to a worst-case analysis framework proposed in this paper is based on a logically consistent probability distribution over the set of statements. In each step n, the system outputs a theory A_n from the first n statements, and one examines whether this theory allows for proving further statements.

If these further statements are selected in a worst-case manner, A_n does not allow for deciding their truth value even with unbounded computational resources. However, a worst-case perspective often leads to overly pessimistic conclusions. In everyday life, many statements are (provably) true or false despite the incompleteness and undecidability of the underlying axiom set.

The probabilistic setting proposed is inspired from the standard Probably Approximately Correct (PAC) framework, and the study borrows the standard statistical learning tools (VC-dimension) in order to bound the completeness expectation $L_n = M(\{e\ s.t.\ A_n \nvdash e \wedge A_n \nvdash \neg e\})$ where M is the measure of probability of the examples.

The complete paper presents results about the induction of a target theory with bounded description length, comparing the *naive* learning (naive adding of examples as axioms), *pruned* learning setting (reduction of axioms) and *fine* learning setting (induction of axioms by minimum description length). It is shown that (corollaries 1-4) : i) in all cases, non-asymptotic performance depends on the underlying distribution M and it might be arbitrarily bad (as in the worst-case setting) ; ii) *fine* learning, and more generally restrictions on the description length entails faster convergence rates than *naive* learning ; iii) for any algorithm with a faster completeness convergence rate than *naive* learning, there exists a distribution such that the error or falsity is not almost surely zero ($\exists M, e\ s.t.\ \forall n, P(A_n \vdash \neg e) > 0$ and $M(e) > 0$) ; iv) *pruned* learning can behave arbitrarily badly in the sense of an infinite asymptotic description length.

Further works consider considers the case of a target theory with infinite description length, and presents negative results (corollaries 5-8) : i) arbitrarily slow convergence rates can occur ; ii) a fast increase of the axiom set can occur. However, the completeness rate goes to 1 as the number of examples goes to infinity.

While results outlined above are based on an oracle (axiomatic optimization or theorem proving with unbounded computational power), we considers as well the case of Turing-computable approximations of such an oracle. Results similar to those of the oracle case are presented (with, unfortunately, a huge computational complexity).

---

a better model for reality, with no guarantee that the current axiom set is correct (see e.g. contradictions in early axiomatisations of the set theory).

Statistical asymptotic and non-asymptotic consistency of Bayesian networks : convergence to the right structure and consistent probability estimates

Sylvain Gelly, Olivier Teytaud

Equipe TAO - INRIA Futurs - Laboratoire de Recherche en Informatique,
Bâtiment 490, Université Paris-Sud, 91405 - Orsay Cedex - France
{sylvain.gelly,olivier.teytaud}@lri.fr

Résumé : Le problème du calibrage de relations à partir d'exemples est un problème classique d'apprentissage. Cette question a en particulier été traitée très différemment par la théorie du processus empirique (fournissant des résultats asymptotiques), par la théorie de l'apprentissage ((A.-N. Kolmogorov, 1961),(V. Vapnik, 1971)), et diverses méthodes plus ou moins empiriques. L'application de la théorie de l'apprentissage aux réseaux bayésiens est encore incomplète et nous proposons une contribution, notamment par l'utilisation de nombres de couverture et son application à des minimisations de risque structurel. Nous considérons en particulier les problèmes suivants :

– Consistance de l'apprentissage dans les réseaux Bayésiens : quels paradigmes conduisent à la consistance/consistance universelle ? En particulier, nous proposons un algorithme pour lequel la consistance est démontrée. Les méthodes usuelles de calibrations locales sont en fait non consistentes.
– Le choix de la structure d'un réseau Bayésien : Comment garantir que la structure ne va pas converger asymptotiquement vers une structure trop complexe ? En particulier, comment générer des états non observables qui simplifient le réseau résultant ? Nous montrons de plus l'influence d'une entropie structurelle sur les nombres de couvertures, qui n'est pas prise en compte par les scores usuels.
– La complexité de l'échantillon dans les réseaux Bayésiens : combien d'exemples faut-il pour atteindre une précision donnée dans l'estimation de la densité ?
– La convergence vers la structure réelle : comment éviter que l'arc $A \rightarrow B$ soit choisi comme paramètre du réseau alors qu'il est inutile ?

1 Introduction

Bayesian networks are a well known and powerful tool for representing and reasoning on uncertainty. One can refer to (Pearl, 2000),(P. Naim, 2004) for a general introduction

to Bayesian networks. Learning the structure and the parameters of Bayesian networks can be done through either expert information or data. Here, we only address the problem of learning from data, i.e. learning a law of probability given a set of examples following this law. Although a lot of algorithms allowing learning Bayesian networks from data exist, several problems remain. Furthermore, the use of learning theory for Bayesian network is still far from complete.

The purpose of this paper is to provide some theoretical insights into the problems of learning Bayesian networks, especially on the problem of structure learning. Statistical Learning Theory is a mature area of Machine Learning that provides efficient theoretical tools to analyse aspects of learning accuracy and algorithm complexity. The use of this tool gives us first of all bounds on the risk given a maximal error and a number of examples, or the number of examples needed to approximate the distribution for a given risk. We also provide, among other things, an algorithm which is guaranteed to converge to an optimal structure.

Furthermore, we make comparisons between the form of our bound to the form of the different scores classically used on Bayesian network structure learning.

The paper is organised as follows : in section 2 we present an overview of our most concrete results. In section 3 we briefly survey some classical ways to learn Bayesian networks from data and discuss the contribution of this paper in regard of existing results. In section 4 we introduce formally the problem and the notations. Section 5 first recalls some classical results of learning theory and presents our result about evaluation of VC-dimensions and covering numbers. We then generalise our results to more general Bayesian networks, with hidden variables, in section 6. Finally, section 7 shows usefull corollaries applied to structure learning, parameters learning, universal consistency, and others.

Due to length constraint, detailed proofs, additional references and further developments can be found in (S. Gelly, 2005).

2 Overview of results

First of all, the usual learning methods for parameters (section 3.1) lead asymptotically to the best parameters if the structure of the Bayesian network is exact. However, we show that the classical method is under optimal if the structure does not match the decomposition of the joint law. On the other hand, we prove universal consistency of global fitting during the minimisation of the empirical error (section 7.2).

We obtain risk bounds. Therefore, given a number of example, and after learning, we can say that the probability to have an error larger than ϵ is bounded by δ. Equivalently, we can deduce the number of examples needed to have a risk $\leq \delta$ to have a error larger than ϵ.

We first notice that we address also the case with hidden variables (section 6). We apply these bounds either in the case of a finite number of variables (section 7.3) and infinite number of variables (section 7.4).

Section 7.5 and theorem 8 gives an algorithm that guarantees universal consistency and overall convergence toward the "good" structure asymptotically. The "good" structure is given in the sense of the user-defined complexity of the structure. Hence, we

prove that the algorithm gives us a not too complex structure.

Let's now compare the form of our bound to the form of existing scores. This comparison gives interesting insights on what is important to measure the complexity of a structure.

The first lemmas helps calculating the covering number of the set of Bayesian networks for a given structure. These covering numbers are directly related to the complexity of the structure. Theorem 7 states that the dominating term of the bound is $RH(r)$ where R is the number of parameters of the structure and where $H(r) = -\sum_{k=1}^{a}(r(k)/R)\ln(r(k)/R)$ with $r(k)$ the number of parameters for the node k. Hence, $H(r)$ is the entropy of the number of parameters calculated over the nodes.

We show then that the number of parameters of the Bayesian network is not the only (and even not the most important) measure of the complexity. Hence, the AIC, BIC or MDL measure are quite different because they don't take into account this $H(r)$.

We also show (difference between theorem 6 and theorem 7) that we have a tighter bound if we consider the number of parameters node by node, without trying to gather the nodes in a more smart way. This means, that more complex patterns on the structure of the Bayesian network do not play a role, for our bound. Only the distribution of the number of parameters between the different nodes is important.

3 Bayesian network learning

The problem of learning a Bayesian network can be divided in two parts :

– Learning the structure of the network, which is related to a graph, and not to the values of the probabilities.
– Given a structure, learning the parameters of the Bayesian network, that is to say the conditional probabilities among variables.

Learning the structure, is a much more challenging problem than estimating the parameters. Hence, the larger part of the works have addressed this issue.

3.1 Learning parameters

The classical approach of this problem is to calculate the maximum of likehood. This leads, with the classical decomposition of the joint probability in a product, to estimate separately each term of the product with the data. This method asymptotically converge toward the true probability, if the proposed structure is exact.

The Bayesian method rather try to calculate the most probable parameters given the data, and this is equivalent, with the Bayes theorem, to weight the parameters with an *a priori* law. The most used *a priori* is the Dirichlet distribution (see for example (Robert, 1994)).

3.2 Structure learning

Structure learning can be divided in two different methods :

– Determine causal relations (and independencies and conditional dependencies) between the random variables, and deduce the structure of the graph.

– Map every structure of Bayesian network to a score and search into the space of all structures for a "good" Bayesian network, that is to say, a structure with a good score.

The space of all structures is super exponential, so heuristics must be provided to search using the second method (limiting to the tree structures, sorting the nodes, greedy search...).

The search could also be done on the space of Markov equivalent structures (the structures which code the same probability law), which has better properties.

Our work, among other results, provide a score to the structures of Bayesian networks, and so is closer to the second category.

3.2.1 Learning causality

The principle of this method is to calculate the independencies (conditionally or not) between the variables. We can cite the algorithms IC, IC*, PC, and more recently BN-PC.

The classical statistical tests used to test the independencies between variables is the χ^2 test. For the hidden variables, the method is more complex, and we must distinguish several types of causality. We will not go further on this point here.

3.2.2 Algorithms based on a score

The notion of score of a structure is generally based on the Occam's razor principle. The score measures the "complexity" of the structure (the meaning depends upon the algorithm). Therefore, the algorithm choose a compromise between the empirical error made by the structure and the score of this structure. A used notation is $Dim(bn)$, "dimension" of the Bayesian network, which counts the number of parameters.

Here follows some well known scores for Bayesian networks.

– AIC criteria (Akaike, 1970) or BIC (Schwartz, 1978) use essentially the $dim(bn)$ to penalise the complexity of the Bayesian network.
– The Minimum Description Length (MDL) principle (Rissanen, 1978) uses the number of arcs and the number of bits used to code the parameters.
– The Bayesian approach puts an *a priori* probability on the structure. For example, The Bayesian Dirichlet score (G. Cooper, 1992) chooses to put a Dirichlet *a priori* on the parameters. Some variants exist, like BDe (D. Heckerman, 1994) , or BD-gamma (C. Borglet, 2002) which uses an hyperparameter, or methods using *a priori* probabilities on each relations child/parent (given for example by an expert).

4 Problem definition and notations

One can refer to (Pearl, 2000),(P. Naim, 2004) for a general introduction to Bayesian networks. Let A_1, ... A_a be a binary random variables. We note $\mathcal{A} = \{A_1, \ldots, A_a\}$. The choose of binary variables is to make the results clearer, but all the results presented below can be measly extended to the general case of discrete random variables.

4.1 Notations

We note, A_b, where b is a subset of $[1, a]$, the random variable product of A_i where $i \in b$. If $b = \emptyset$, then A_b is the event always true. A **Bayesian network** is considered as a set K_1,...,K_a of subsets of $[1, a]$ where $i \notin K_i$ (We can suppose that $i < K_i$, that is to say that i is smaller than every element in K_i, without loss of generality). A **instanced Bayesian network** ibn, associated with a Bayesian network bn, is a law on $(A_1, \ldots, A_i)$ such that $ibn(A_1, \ldots, A_a) = \prod_j P(A_j|A_{K_j})$. With bn a Bayesian network, and ibn an instance of bn, we will say by abuse that $ibn \in bn$. We will map ibn with a vector of size 2^a corresponding to all the probabilities of all events $(A_1 = v_1, \ldots, A_a = v_a)$. A **Bayesian network** bn is said **well defined** if there exists an instance ibn and if there does not two instances with different probabilities $P(A_j|A_{K_j})$. We call **parameter** of a Bayesian network (BN), one of the real number $P(A_j|A_{K_j})$. We call **number of parameters** of a BN, and we note $p(bn)$ the sum of $2^{\#K_j}$, where $\#b$ is the cardinal of b.

We consider $\hat{P}$ an empirical law (i.e. a sample of Dirac masses located at examples). Let P be a target law of probability. The sample leading to $\hat{P}$ is supposed independent and identically distributed (i.i.d.). We note E and $\hat{E}$ the expected value operators associated to P and $\hat{P}$ respectively. We note

$$\chi = (0, 0, 0, 0, \ldots, 0, 1, 0, \ldots, 0, 0, 0) \in \{0, 1\}^{2^a}$$

(all zeros except one 1 on the ith position with probability the probability of the i^{th} set of possible values of $A_1 \ldots A_a$).

For Q a vector of size 2^a, of sum 1, identified as a probability distribution on the random vector $(A_1, \ldots, A_a)$ (more precisely $Q(i)$ is the probability of $(A_1 = a_1, ..., A_a = a_a)$, with $(a_1, ...a_a)$ the ith tuple of size a, among the 2^a tuples possible), we define

$$L(Q) = E(\sum_{i \in [1, 2^a]} |Q(i) - \chi(i)|^2)$$

where $\sum$ is the sum operator on vector, and $\hat{L}(Q) = \hat{E}(\sum_{i \in [1,2^a]} |Q(i) - \chi(i)|^2)$. If bn is a well defined BN, we note $L(bn) = \inf_{ibn \in bn} L(ibn)$ where inf is on the ibn instanced Bayesian networks associated to bn

4.2 Preliminary lemmas and propositions

To spot the interest of $L(.)$ and $\hat{L}(.)$, we can remark that :

Lemma 0 :

With $N(Q) = \sum_{i \in [1,2^a]} (P(i) - Q(i))^2$. and $\hat{N}(Q) = \sum_{i \in [1,2^a]} (\hat{P}(i) - Q(i))^2$, we claim :

$$L(Q) = N(Q) + 1 - \sum_{i \in [1, 2^a]} P_i^2$$

$$\hat{L}(Q) = \hat{N}(Q) + 1 - \sum_{i \in [1, 2^a]} \hat{P}_i^2$$

Moreover, we can remark that

Proposition A (see (S. Gelly, 2005) for the proof) :
With probability $1-\delta$, with $x^* \in argminL(.) = argminN(.)$, for all $\hat{x} \in argmin\hat{L} = argmin\hat{N}$, with $\sup_\delta X$ the $1-\delta$ quantile of X :

$$L(\hat{x}) \leq L(x^*) + 2\sup_\delta |L - \hat{L}|$$

And finally :
Proposition B :
With probability $1-\delta$, with $x^* \in argminL(.) = argminN(.)$, For all $\hat{x} \in argmin\hat{L} = argmin\hat{N}$, with $\sup_\delta X$ the $1-\delta$ quantile of X :

$$N(\hat{x}) \leq N(x^*) + 2\sup_\delta |L - \hat{L}|$$

Proof : Consequence of lemma 0 and proposition A.
All these elements confirm the interest of $\hat{L}$, which has both the interest of being an empirical average and the advantage of being closely related to natural cost functions.

5 Learning theory results

The VC dimension ((V. Vapnik, 1971)) is the more classical tool of learning theory. It allows to bound, depending on the "complexity" of the function family, the difference between the empirical mean and the expected value of the loss function. In particular, it allows to quantify the inaccuracy of calibration of a function. This type of calculus has already been done in (P. Wocjan, 2002). We show similarly results in section 5.1. The use of covering numbers, already known on the time of (A.-N. Kolmogorov, 1961), allows more precise bounds, as shown in section 5.2.
We will note $F(H,\delta)$ the smallest real Δ such that $P(\sup_{h\in H} |\hat{L}(h) - L(h)| \geq \Delta/\sqrt{n}) \leq \delta$, with n the number of examples. $F(H,\delta)$ depends upon n, but in many cases the dependency upon n can be removed (i.e. the supremum on n is not a bad approximation) and so we often refer to $F(H,\delta)$.

5.1 Bounds based on VC dimension

Let bn a Bayesian network defined by $K_0 = \emptyset$ and $K_j = [1, j-1]$ for $j > 1$ (totally connected Bayesian network). Then with a probability more than $1-\delta$:

$$\sup_{ibn\in bn} |\hat{L}(ibn) - L(ibn)| \leq F([0,1]^{2^a}, \delta)/\sqrt{n}$$

For a Bayesian network bn, with a probability more than $1-\delta$:

$$\sup_{ibn\in bn} |\hat{L}(ibn) - L(ibn)| \leq F(\{ibn\}, \delta)/\sqrt{n}$$

The application which maps an instance of a Bayesian network and a value of $A_{[1,a]} = A_1 \times A_2 \times A_3 \times \cdots \times A_a$ to the *log* (extended by $\log(0) = -\infty$) of the probability of this value is linear in the log of the parameters of the Bayesian network. The

VC dimension is so upper bounded by the number of parameters. Taking exponential is preserving the VC dimension, and so the VC dimension of $\{ibn\}$, seen as application mapping $A_{[1,a]}$ to a probability is upper bounded by the number of parameters.

We then deduce the result :

Theorem C :

The VC dimension of the set of Bayesian network $ibn \in bn$ is upper bounded by the number of parameters V of bn. So thanks to classical results of learning theory

$$P(\exists ibn \in bn | \hat{L}(ibn) - L(ibn)| \geq \epsilon) < 8(32e/\epsilon)\log(128e/\epsilon))^V \exp(-n\epsilon^2/32)$$

if $n \geq V$, and the covering number of ibn for the metric $d(ibn_1, ibn_2) = E(|ibn_1(A_{[1,a]}) - ibn_2(A_{[1,a]})|)$ is upper bounded by $e(R+1)(4e/\epsilon)^R$.

Proof : These results are classical in learning theory. See e.g. (M. Antony, 1999, Th18.4 and 17.4) for the upper bound on the probability and (M. Antony, 1999, Th18.4, p251) for the covering number. We note that our results, even if they are for a sharper norm $N1(.)$, defined on the following, are better.

5.2 Bound based on the covering number

The covering numbers are a classical tool of learning theory. Inequalities of large deviations obtained with this tool are usually tighter than those obtained using VC-dimension.

5.2.1 Introduction

If one can cover $\mathcal{F}$ with $N1(\mathcal{F}, \epsilon)$ ϵ balls for the distance $d(x, y) = \sum |x_i - y_i|$, if $\hat{L}$ and L are between 0 and 2, then :

1. the risk, for a given function, to have a deviation $|\hat{L} - L|$ more than 2ϵ, is bounded by $2\exp(-2n\epsilon^2)$;
2. The risk to have at least one of the centers of the balls having a deviation more than 2ϵ is upper bounded by $2N1(\mathcal{F}, \epsilon)\exp(-2n\epsilon^2)$;
3. If $d(f, g) \leq \epsilon \Rightarrow |L(f) - L(g)| \leq \epsilon$ and $d(f, g) \leq \epsilon \Rightarrow |\hat{L}(f) - \hat{L}(g)| \leq \epsilon$, (which is the case here, see lemma 2), then the risk to have at least a function in $\mathcal{F}$ having a deviation more than 4ϵ is upper bounded by $2N1(\mathcal{F}, \epsilon)\exp(-2n\epsilon^2)$. Indeed, if for all g of ϵ-skeleton C, we have $|\hat{L}(g) - L(g)| \leq 2\epsilon$, so we can map every f to one $g \in C$ such that $d(f, g) < \epsilon$ and so

$$|\hat{L}(f) - L(f)| \leq |\hat{L}(f) - \hat{L}(g)| + |\hat{L}(g) - L(g)| + |L(g) - L(f)| \leq \epsilon + 2\epsilon + \epsilon \leq 4\epsilon$$

The risk to have, among $\mathcal{F}$, a deviation more than ϵ is then upper bounded by $\delta = 2N1(\mathcal{F}, \epsilon/4)\exp(-2n(\epsilon/4)^2)$.

Then we can write :

Proposition (maximal deviation for a covering number given) :

$$\sqrt{n}F(\mathcal{F}, \delta) \leq \inf\{\epsilon | \log(2N1(\mathcal{F}, \epsilon/4)) - n\epsilon^2/8 \leq \log\delta\}$$

A lot of variations of this type of result exists in the literature. One can for example see (Vidyasagar, 1997) and (M. Antony, 1999).

The covering number $N_\infty(\mathcal{F}, \epsilon)$ of $\mathcal{F} = [0,1]^{2^a}$ is upper bounded by $\lceil 1/2\epsilon \rceil^{2^a}$ for the distance $d(x,y) = \sup_i |x_i - y_i|$.

The covering number $N1(\mathcal{F}, \epsilon)$ of $\mathcal{F} = \{ibn \in bn\}$ is upper bounded as explained in the following subsection for the distance $d(x,y) = \sum |x_i - y_i|$.

5.2.2 Cover number of $\mathcal{F}$

We assume, without loss of generality that the nodes of the Bayesian network are topologically sorted ($i < K_i$ for i node of the BN)

Let E_k a partition of the node set such as :

– If $k \leq k'$ then $\forall (i,j) \in E_k \times E_{k'}, i \leq j$

– There is no edge between two nodes of a same E_k.

We call depth the number k corresponding to the partition E_k and l_k the number of the last element (node) of E_k. By convention, $E_0 = \emptyset$ and $l_0 = 0$.

Lemma 1 :

$$N1(F_k, 2^{nbe(k)}\epsilon' + \epsilon) \leq N(F_{k-1}, \epsilon) N_{\inf}(T_k, \epsilon')$$

where

– F_k indicates the set of the functions calculated by the Bayesian network until the level k (that is to say using only the nodes of $\bigcup_{i=1}^{k} E_i$).

– $N_{\inf}$ indicates the covering number for the sup norm.

– T_k indicates the set of the vectors of the probabilities involved in the transition from the level $k-1$ to the level k (it is $[0,1]^{2^{l_k}}$).

– $nbe(k)$ indicates the number of the nodes of the Bayesian network in the level k, so $\#E_k$;

– $l_k = \sum_{i=1}^{k} nbe(i)$;

– $N1(.,.)$ indicates the covering numbers for the norm $x \mapsto \sum |x|$.

Lemma 2 (proof in (S. Gelly, 2005)) :

$$|L(Q) - L(Q')| \leq \sum_i |Q_i - Q'_i|$$

Lemma 3 :

$$N_\infty([0,1]^h, \epsilon) \leq \lceil \frac{1}{2\epsilon} \rceil^h$$

Lemma 4 : (proof in (S. Gelly, 2005))

$$N_\infty(T_k, \epsilon) \leq \lceil \frac{nbe(k)}{2\epsilon} \rceil^{r(k)}$$

where T_k indicates the set of the vectors of the conditional probabilities involved in the transition from the level $k-1$ to the level k and where $r(k)$ indicates the number of parameters of the network involved in the transition between level $k-1$ and k.

Precisely, for a fixed k, T_k is the set of $P(E_k|\bigcup_{i=1}^{k-1} E_i)$, the $E_i, i = 1, ..., k$ taking the 2^{l_k} possible values. $r(k)$ indicates the number of the $P(A_i|K_i)$ with $A_i \in E_k$, that is to say the number of parameters for this level.

Lemma 5 (proof in (S. Gelly, 2005)) :

Let K be the number of levels ; then

$$lN1(K) \leq \sum_{i=1}^{K} r(i) \ln(\lceil \frac{nbe(i)2^{nbe(i)-1}}{\Delta_i} \rceil)$$

where $\epsilon_i > 0, i = 1...K$, $\epsilon_i < \epsilon_K, i = 1...K-1$, $\epsilon_0 = 0$, $\Delta(i) = \epsilon_i - \epsilon_{i-1}$ and $lN1(i) = log(N1(F_i, \epsilon_i))$ and with the notation $lN1(0) = 0$.

Theorem 6 (proof in (S. Gelly, 2005)) :

$$lN1(\epsilon) \leq \sum_{k=1}^{K} r(k) \ln(nbe(k)2^{nbe(k)-1} + \epsilon) - \sum_{k=1}^{K} r(k) \ln(\epsilon r(k)/R)$$

with $R = \sum_{i=1}^{K} r(i)$, $\epsilon = \epsilon_K$ et $lN1(\epsilon) = lN1(F_K, \epsilon)$, in particular for K the number of the last level.

Theorem 7 (the proof can be found in (S. Gelly, 2005)) :

The best partition $\{E_k\}$ (for this bound) is the one where all the E_k contain only one node. We have then :

$$lN1(\epsilon) \leq \sum_{k=1}^{a} r(k) \ln(1+\epsilon) - \sum_{k=1}^{a} r(k) \ln(\epsilon r(k)/R)$$

$$\leq R \ln((1+\epsilon)/\epsilon) + RH(r)$$

where $H(r) = -\sum_{k=1}^{a} (r(k)/R) \ln(r(k)/R)$.

Remark 1 : We get a better bound on the covering number than the one we get from the VC-dimension which is $R(2^a/\epsilon)^R$ (because $R \leq 2^a$).

Remark 2 : For a fixed R (total number of parameters), our inequality has a term in $log((1/\epsilon)^R)$ and a term which is the entropy of the vector $(r(1), \ldots, r(a))$, which shows that the less the parameters are equally distributed, the more the covering number is well controlled.

Proof of lemma 1 :

Let $k \geq 1$ fixed. Let $Pa(E_k)$ be the set of parent nodes of E_k. Let X be the set of the vectors of size $2^{\sum_{i=1}^{k-1} \#E_i} = 2^{l_{k-1}}$ representing the probabilities (hence of sum 1) of all Bayesian networks of a given structure (all $ibn \in bn$) until the level $k-1$. More precisely $X = \{x = P(A_1, ..., A_{l_{k-1}})\}$, the l_{k-1}-tuple of A_i taking all the $2^{l_{k-1}}$ possible values. Let Y be the set of vectors of size $2^{\sum_{i=1}^{k} \#E_i} = 2^{l_k}$ representing the probabilities of $ibn \in bn$ until the level k. More precisely, $Y = \{y = P(A_1, ..., A_{l_k})\}$, the l_k-tuples of A_i taking all the 2^{l_k} possible values.

Let's cluster the vectors of the set X by classes $\tilde{X}_i$ such as for all $x \in \tilde{X}_i$ the values of the parents $Pa(E_k)$ are identical. Let N be the number of such classes. Let $t_i^j, i \in [1, N], j \in [1, 2^{neb(k)}]$ the probability of the jth value of the new variables

(of level k) knowing a value of the class $\tilde{X}_i$ (each value of the variables in $\tilde{X}_i$ is appropriate because, by definition of $Pa(.)$, the new variables depend only on $Pa(E_k)$ among $E_1, ..., E_k$).

Let $y, y' \in Y$. We can then claim $y = (y_1, y_2, ..., y_N)$ with $y_i = (t_i^1 \tilde{X}_i, t_i^2 \tilde{X}_i, ..., t_i^{2^{neb(k)}} \tilde{X}_i)$ and $y' = ({y'}_1, {y'}_2, ..., {y'}_N)$ with ${y'}_i = ({t'}_i^1 \tilde{X}'_i, {t'}_i^2 \tilde{X}'_i, ..., {t'}_i^{2^{neb(k)}} \tilde{X}'_i)$.

Let $\epsilon' = sup_{i,j}|t_i^j - {t'}_i^j|$ and $\epsilon = sup_i \|\tilde{X}_i - \tilde{X}'_i\|_1$. Then :

$$\|y - y'\| = \sum_{i=1}^{N} \sum_{j=1}^{2^{neb(k)}} \|((t_i^j - {t'}_i^j)\tilde{X}_i + {t'}_i^j(\tilde{X}_i - \tilde{X}'_i)\|_1$$

$$\|y - y'\| \leq \sum_{i=1}^{N} \sum_{j=1}^{2^{nbe(k)}} \epsilon' \|\tilde{X}_i\|_1 + {t'}_i^j \|\tilde{X}_i - \tilde{X}'_i\|_1$$

$$= \sum_{i=1}^{N} 2^{neb(k)} \epsilon' \|\tilde{X}_i\|_1 + \|\tilde{X}_i - \tilde{X}'_i\|_1 \leq 2^{nbe(k)} \epsilon' + \epsilon$$

Therefore,

$$N1(F_k, 2^{nbe(k)}\epsilon' + \epsilon) \leq N(F_{k-1}, \epsilon) N_{\inf}(T_k, \epsilon')$$

■

5.2.3 Summary of the results

We have calculated an upper bound on the covering number of the family of instanced Bayesian networks $ibn \in bn$ for a given structure bn. This structure determines the number of parameters $r(k)$ for $k \in [1, K]$ (and $R = \sum_{k=1}^{a} r(k)$).

Then, theorem 7 states that for all $\epsilon > 0$:

$$lN1(\epsilon) \leq \sum_{k=1}^{a} r(k) \ln(1 + \epsilon) - \sum_{k=1}^{a} r(k) \ln(\epsilon r(k)/R)$$

The lemma 2 states that the conditions $d(f, g) \leq \epsilon \Rightarrow |L(f) - L(g)| \leq \epsilon$ and $d(f, g) \leq \epsilon \Rightarrow |\hat{L}(f) - \hat{L}(g)| \leq \epsilon$ are true.

So we can here apply the results stated in the subsection 5.2.1, and then the risk to have, among $\mathcal{F}$, a deviation more than ϵ is then upper bounded by $\delta = 2N1(\mathcal{F}, \epsilon/4) \exp(-2n(\epsilon/4)^2)$.

Therefore, $F(\mathcal{F}, \delta) \leq \sqrt{n} \inf\{\epsilon | \log(2N1(\mathcal{F}, \epsilon/4)) - n\epsilon^2/8 \leq \log \delta\}$.

6 Results with hidden variables

We here consider the case where some variables are hidden, so only a part of all the variables are involved in the calculus of $\hat{L}$ or L. It is important to remark that it is not equivalent to reduce the Bayesian network to a smaller Bayesian network. For

example, a network with a hidden variable B and observed variables A_i for $i \in [1, d]$, with dependencies $P(A_i|B)$, has only $2d + 1$ parameters and is difficult to modelise (i.e. would need much more parameters) with a Bayesian network which has only the A_i as variables.

By mapping a Bayesian network to a vector (of sum 1) of the probabilities it calculates, a Bayesian network in which some variables are hidden can be mapped to a reduced vector (the vector of marginalized probabilities). If all the variables are binary (which is the case in this paper), the number of probabilities to code is divided by 2 by each variable become hidden. A instance of a Bayesian network ($\tilde{ibn}$) which has v variables, among them l hidden variables, can be mapped to an element of $[0, 1]^{2^{v-l}}$ summing to 1, whereas the Bayesian network ibn corresponding which does not have hidden variables, gives 2^v probabilities (hence a vector in $[0, 1]^{2^v}$, summing to 1). $\tilde{ibn}$ then equals $summing(ibn)$, where $summing(.)$ is an application summing some quantities.

As $summing(.)$ is 1-lipschitz for the distance $d(x, y) = \sum |x_i - y_i|$ (i.e. $d(\tilde{x}, \tilde{y}) \leq d(x, y)$), we deduce :

Proposition maximal deviation in a Bayesian network with hidden variables :

The risk to have a deviation at least ϵ for a $\tilde{ibn} \in \tilde{bn}$ is upper bounded by $2N1(bn, \epsilon/4) \exp(-n\epsilon^2/8)$, with $\tilde{bn} = \{\tilde{ibn}/ibn \in bn\}$, and $F(\tilde{bn}, \delta) \leq F(bn, \delta)$.

Remarks : We can notice that we don't improve the bound in spite of simpler network. We can of course bound $F(\tilde{bn}, \delta)$ by $F([0, 1]^{v-l}, \delta)$ if the number of hidden variables is so large that this rough bound becomes the best.

7 Algorithms

Many applications of the calculus above can be defined, in the same spirit of use of covering numbers, to give :

– confidence intervals non-parametric non-asymptotic ;
– algorithms universally consistents.

Furthermore, results of type boostrap on Donsker classes, show how to build asymptotic confidence intervals of inaccuracy of Bayesian networks. One can refer to (Van Der Vaart A., 1996).

We state in the sections below some of the numerous corollaries one can deduce from the calculus of covering numbers above. Theses corollaries are also true with hidden variables.

7.1 Choose between several structures of Bayesian network

Let's assume that someone have to choose between several structures $bn_1, \ldots, bn_h$. Consider the algorithm that chooses bn_{i_0} such as $\inf_{ibn \in bn_{i_0}} \hat{L}(ibn) + F(bn_{i_0}, \delta)/\sqrt{n}$ is minimal and chooses $\hat{ibn} \in bn_{i_0}$ such as $\hat{ibn} = argmin_{ibn \in bn_{i_0}} \hat{L}(ibn)$. So, the algorithm chooses the *structure* minimising the empirical error penalised by a term depending upon the complexity of the structure. Then, it chooses the *Bayesian network* of this structure minimising the empirical error.

Corollary C1 (proof in (S. Gelly, 2005)) : Then, $L(\hat{ibn}) \leq L(ibn') + \epsilon$ for all $ibn' \in \cup bn_i$, with $\epsilon = 3 \sup F(bn_i, \delta)/\sqrt{n}$, with a risk upper bounded by $h\delta$. (the constant 3 in ϵ is not optimal)

This gives a natural criteria to choose between several structures, in the spirit of the method of *"minimisation of structural risk"*, which is classical in learning theory.

7.2 Comparison between local and global fitting : consistency of the minimisation of $\hat{L}$

Corollary C2 (proof in (S. Gelly, 2005)) : Consider bn a Bayesian network. Then for any distribution P,

$$L(argmin_{ibn \in bn} \hat{L}) \rightarrow \inf_{bn} L$$

whereas for some distributions P,

$$L(ibn \,/ \forall P(A_i|A_{K_i}) \in bn,\, ibn(A_i, A_{K_i})/ibn(A_{K_i}) = \hat{P}(A_i\, A_{K_i})/\hat{P}(A_{K_i})) \not\rightarrow inf_{bn} L$$

(i.e., calibrating each coefficient of bn on $\hat{P}$ leads asymptotically to a non-optimal ibn), with $ibn(B)$ for B a set of variable, is the probability given by the Bayesian network ibn for the variables B.

7.3 Universal consistency and bound with a finite number of variables

We assume that a heuristic system is given in order to rank dependencies between variables, for the building of the structure. This method, whenever asked, provides a dependency $A_j \rightarrow A_i$ to be added to increase a dependency $P(A_i|A_{K_i})$ to a dependency $P(A_i|A_{K_i \cup \{j\}})$. This method is designed to increase step by step the complexity of the structure.

Consider the following algorithm, for $\epsilon(n)$ a sequence converging to 0 as $n \rightarrow \infty$:

- Consider n the number of examples and δ the risk threshold chosen by the user.
- Heuristically sort the list of dependencies (possibly using a separate database).
- As long as the next dependency added to bn does not lead to $F(bn, \delta)/\sqrt{n} > \epsilon(n)$, add the dependency the most suitable according to the heuristic ;
- Choose $\hat{ibn} \in bn$ minimising $\hat{L}$.
- Claim $L(ibn) \leq \hat{L}(ibn) + F(bn, \delta)/\sqrt{n}$.

Corollary C3 :

- with confidence at least $1 - \delta$, the bound provided on $L(ibn)$ is true ;
- in the limit of a large number of examples, $L(\hat{ibn})$ converges to $inf_{ibn} L(ibn)$ (inf among any ibn, independently of the structure, and not only $inf_{ibn \in bn} L(ibn)$), at least if the heuristic, within a finite number of increases of the structure, leads to bn such that $inf_{ibn \in bn} L(ibn) = inf_{ibn} L(ibn)$ (this is a small and natural hypothesis as the heuristic can simply lead to the complete graph between observable variables if the number of dependencies is sufficiently large).

The proof is a consequence of the convergence of $F(bn, \delta)/\sqrt{(}n)$ to 0 (as it is upper bounded by $\epsilon(n)$) as $n \rightarrow \infty$.

7.4 Universal consistency and confidence intervals with infinitely many variables

We consider here an infinite number of states, but a finite number of examples. Variable j of example i is noted $a_{i,j}$. The sequence of vectors [1] $(a_{i,1}, \ldots, a_{i,743}, \ldots)$ for $i \in \mathbb{N}$ is assumed independently identically distributed. The algorithm is as follows :

1. the user provides n, ϵ and δ ; an oracle provides the $a_{i,j}$ when they are required by the program.
2. evaluate bn maximal for the inclusion[2] (chosen by any heuristic among multiple possible solutions, provided that bn increase as n increases), such that $F(bn, \delta)$ is upper-bounded by ϵ ; the variables modelled by b_n are the observable ones among the union of the A_j and A_{K_j} such that bn is defined by the $P(A_j|A_{K_j})$;
3. choose $ibn \in bn$ minimising $\hat{L}$;
4. provide to the user a bound $L(ibn) \leq \hat{L}(ibn) + F(bn, \delta)/\sqrt{n}$;

Corollary C4 :
Let's note $mod(bn)$ the set of events which are deterministic functions of observable variables modelled by bn.

– for any E event depending upon a finite number of A_j, $ibn(E)$ is evaluated if n is large enough and its value converges to $P(E)$ as $n \to \infty$, if at least the heuristic method guarantees that for a given increasing sequence of integers k_i, the number of dependencies is bounded by k_i as long as the i^{th} observable variable is not added to the network (this is a natural requirement).
– the bound provided on $L(ibn)$ holds with probability at least $1 - \delta$.
– thanks to the Borell-Cantelli lemma (see e.g. (Vidyasagar, 1997, p26)), one can write that if $\sum_n \delta_n$ is finite (for example $\delta_n = 1/n^2$) and if $F(bn_n, \delta_n)/\sqrt{n} \to 0$ as $n \to \infty$, with bn_n the structure chosen for a number n of examples, then there is almost sure convergence of $\sup |P(E) - ibn(E)|$ for $E \in mod(b_n)$ to 0 ; we must ensure $\delta_n \leq \delta$ to assert, moreover, that the bound $\hat{L}(ibn) + F(bn, \delta)/\sqrt{n}$ holds.

7.5 Universal consistency and convergence to the right network of dependencies

We propose in this section an algorithm in order to build Bayesian networks having two important properties :

– it is universally consistant ;
– the size of the structure converges to the optimal one.

The second point is not trivial, as it is very difficult to guarantee convergence to a non-redundant structure.

Precisely, we claim the

Theorem 8 : universal consistency and convergence to the right structure

[1] There are infinitely many vectors but these vectors are countable.

[2] We say that a Bayesian network bn_1 is included in a Bayesian network bn_2 if any dependency in bn_1 is a dependency in bn_2 within a renumbering of latent variables.

Define

$$ibn \in argmin_{S(ibn)\leq n} \hat{L}(ibn) + R(ibn, n)$$

where S is an application which associates a real number to any instantiated Bayesian network, such that $\forall (ibn_1, ibn_2) \in bn\ S(ibn_1) = S(ibn_2)$ (i.e., two Bayesian networks having the same structure have the same image through S) , and where $R(ibn, n) = R'(ibn)R(n)$ associates a real number to an instantiated Bayesian network ibn and to a sample size n.

We note in the sequel (by abuse of notation) $S^{-1}(n) = \{ibn / S(ibn) \leq n\}$.

Then :

1. **universal consistency :** if H0, H1 and H2 hold, then $L(ibn)$ almost surely goes to L^* ;
2. **convergence of the size of the structure :** if H0, H1, H2 and H3 hold, then $R'(ibn) \to R'(ibn^*)$ where ibn^* is such as $L^* = L(ibn^*)$.

H0 : for n sufficiently large, $ibn^* \in S^{-1}(n)$;
H1 : $sup_{ibn \in S^{-1}(n)}\ R'(ibn)R(n) \to 0$ as $n \to \infty$;
H2 : $F(S^{-1}(n), 1/n^2)/\sqrt{n} \to 0$ as $n \to \infty$;
H3 : $F(S^{-1}(n), 1/n^2)/(R(n)\sqrt{n}) \to 0$ as $n \to \infty$;

Proof :

Define $bn = S^{-1}(n)$ and $\epsilon(bn, n) = sup_{ibn \in S^{-1}(n)} |\hat{L}(ibn) - L(ibn)|$.

Let's proof the universal consistency under hypothesis H0, H1, H2.

$$\begin{aligned} L(ibn) &\leq \hat{L}(ibn) + \epsilon(bn, n) \\ &\leq \inf_{ibn' \in bn} \hat{L}(ibn') + R(ibn', n) - R(ibn, n) + \epsilon(bn, n) \\ &\leq \inf_{ibn' \in bn} L(ibn') + \epsilon(bn, n) + R(ibn', n) - R(ibn, n) + \epsilon(bn, n) \\ &\leq \inf_{ibn' \in bn} L(ibn') + R(ibn', n) + 2\epsilon(bn, n) \end{aligned}$$

Thanks to H1, we only have to prove that $\epsilon(bn, n) \to 0$ almost surely.

By definition of $F(.,.)$, $P(\epsilon(bn, n) \geq F(bn, 1/n^2)/\sqrt{n}) \leq 1/n^2$.

In particular, for any ϵ, H2 implies that for n sufficiently large, $F(bn, 1/n^2)/\sqrt{n} < \epsilon$, and so $P(\epsilon(bn, n) > \epsilon) \leq 1/n^2$. Thanks to the Borell-Cantelli lemma, the sum of the $P(\epsilon(bn, n) > \epsilon)$ being finite for any $\epsilon > 0$, $\epsilon(bn, n)$ almost surely converges to 0.

We have achieved the proof of consistency. We now start the proof of the convergence of the size of the structure.

Thanks to H0, if n is sufficiently large, $ibn^* \in bn$. We restrict our attention to such n.

$$\begin{aligned} \hat{L}(ibn) + R(ibn, n) &\leq \hat{L}(ibn^*) + R(ibn^*, n) \\ R'(ibn)R(n) &\leq R'(ibn^*)R(n) + \hat{L}(ibn^*) - \hat{L}(ibn) \\ R'(ibn)R(n) &\leq R'(ibn^*)R(n) + L^* + 2\epsilon(bn, n) - L(ibn) \\ R'(ibn) &\leq R'(ibn^*) + 2\epsilon(bn, n)/R(n) \end{aligned}$$

It is then sufficient, using H3, to show that $\epsilon(bn,n)/R(n) \rightarrow 0$ almost surely. Let's show this by Borell-Cantelli as well. By definition of $F(.,.)$, $P(\epsilon(bn,n) \geq F(bn,1/n^2)/\sqrt{n}) \leq 1/n^2$.

In particular, for any ϵ, H3 implies that for n sufficiently large, $F(bn,1/n^2)/(R(n)\sqrt{n}) < \epsilon$, and so $P(\epsilon(bn,n)/R(n) > \epsilon) \leq 1/n^2$. Thanks to the Borell-Cantelli lemma, the sum of the $P(\epsilon(bn,n)/R(n) > \epsilon)$ being finite for any $\epsilon > 0$, $\epsilon(bn,n)/R(n)$ almost surely converges to 0. ■

8 Conclusions

We have evaluated the covering numbers of Bayesian networks. We have applied these results to algorithms and scores for choosing between structures. We then establish results on consistency and discovering of the real structure of data. Our results concern networks with non-observable states as well. In particular, we have :

1. proposed some criterions of quality of an instantiated Bayesian network, showing the links between these criterions and other criterions (lemma 0 and proposition B), thus generalising our results to other criterions as well ;
2. evaluated VC-dimensions and covering numbers of Bayesian networks (including networks with non-observable states) (theorem C and theorem 6) ;
3. proposed an algorithm with guaranteed universal consistency and almost sure convergence towards a structure with optimal size (including networks with non-observable states) (theorem 8) ;
4. derived some corollaries, among which :
   - scores for choosing between structures (corollary C1), showing the influence of an entropy (theorem 7) ;
   - bounds on the precision of probability estimations (corollary C1 and C3) ;
   - a comparison between global optimisation of a Bayesian network and local parametrisation of a Bayesian network (corollary C2) ;
   - established universal consistencies for data mining in large dimension (corollary C4).

Acknowledgements

This work was supported in part by the PASCAL Network of Excellence.

Références

A.-N. KOLMOGOROV V.-M. T. (1961). ϵ-entropy and ϵ-capacity of sets in functional spaces. In *Amer. Math. Soc. Transl. 17, pp 277-364*.

AKAIKE H. (1970). Statistical predictor identification. In *Ann. Inst. Statist. Math., 22 :203-217*.

C. BORGLET K. K. (2002). Graphical models - methods for data analysis and mining. In *John Wiley and Sons, Chichester, UK*.

D. HECKERMAN, D. GEIGER M. C. (1994). Learning bayesian networks : The combination of knowledge and statistical data. In *Ramon Lopez de Mantaras et David Poole, editors, Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*.

G. COOPER E. H. (1992). A bayesian method for the induction of probabilistic networks from data. In *Machine Learning, 9 :309-347*.

M. ANTONY P. B. (1999). Neural network learning : Theoretical foundations. In *Cambridge University Press*.

P. NAIM, P.-H. WUILLEMIN P. L. E. O. P. E. A. B. (2004). Réseaux bayésiens. In *Eyrolles*.

P. WOCJAN, D. JANZING T. B. (2002). Required sample size for learning sparse bayesian networks with many variables. In *LANL e-print cs.LG/0204052*.

PEARL J. (2000). Causality : models, reasonings and inference. In *Cambridge University Press, Cambridge, England*.

RISSANEN J. (1978). Modeling by shortest data description. In *Modeling by shortest data description*.

ROBERT C. (1994). The bayesian choice : a decision theoric motivation. In *Springer, New York*.

S. GELLY O. T. (2005). Statistical asymptotic and non-asymptotic consistency of bayesian networks : convergence to the right structure and consistent probability estimates. In `http://www.lri.fr/~teytaud/Publications/SoumissionsEtDrafts/coltBNLong.%pdf`.

SCHWARTZ G. (1978). Estimating the dimension of a model. In *The annals of Statistics, 6(2) :461-464*.

V. VAPNIK A. C. (1971). On the uniform convergence of relative frequencies of events to their probabilities. In *Theory of probability and its applications, 16 :264-280*.

VAN DER VAART A. W. J. (1996). Weak convergence and empirical processes.

VIDYASAGAR M. (1997). A theory of learning and generalization. In *Springer*.

Apprentissage statistique et programmation génétique: la croissance du code est-elle inévitable?

Sylvain Gelly[1], Olivier Teytaud[1], Nicolas Bredeche[1], Marc Schoenauer[1]

Equipe TAO - INRIA Futurs - Laboratoire de Recherche en Informatique,
Bâtiment 490, Université Paris-Sud, 91405 - Orsay Cedex - France
{nom}@lri.fr

Résumé : Le "Code bloat", l'augmentation inconsidérée de la taille du code, est un problème fondamental en programmation génétique (GP). Ce papier propose une analyse théorique du bloat dans le cadre de la régression symbolique en GP, du point de vue de la théorie statistique de l'apprentissage. Deux sortes de bloat sont distinguées, selon que le concept soit dans l'espace de recherche ou non. Des résultats importants sont prouvés à partir de résultats classiques de théorie de l'apprentissage. Précisément, la dimension VC des programmes est calculée, et des résultats classiques de théorie de l'apprentissage permettent alors de déduire des propriétés de consistance universelle. Nous montrons alors que choisir a priori une taille de programme selon le nombre d'exemples, bien que conduisant à la consistance universelle, conduit à un phénomène de bloat, alors qu'une fitness adéquatement modifiée permet de l'éviter tout en préservant la consistance universelle.

1 Introduction

Code bloat (or code growth) denotes the growth of program size during the course of Genetic Programming (GP) runs. It has been identified as a key problem in GP from the very beginning (Koza, 1992), and to any variable length representations based learning algorithm (Langdon, 1998). It is today a well studied phenomenon, and empirical solutions have been proposed to effectively address the issue of code bloat (see section 2). However, very few theoretical studies have addressed the issue of bloat.

The purpose of this paper is to provide some theoretical insights into the bloat phenomenon, in the context of symbolic regression by GP, from the Statistical Learning Theory viewpoint (Vapnik, 1995). Indeed, Statistical Learning Theory is a recent, yet mature, area of Machine Learning that provides efficient theoretical tools to analyze aspects of learning accuracy and algorithm complexity. Our goal is both to perform an in-depth analysis of bloat and to provide, if possible, appropriate theoretical solutions to avoid it.

The paper is organized as follows : in section 2 we briefly survey some explanations for code bloat that have been proposed in the literature. Section 3 sets the scenery, and provides an informal description of our results from a GP perspective before discussing

their interest for the GP practitioner. Section 4 gives a brief overview of the basic results of Learning Theory that will be used in Section 5 to formally prove all the advertised results. Finally, section 6 discusses the consequences of those theoretical results for GP practitioners and gives some perspectives about this work.

2 Code Bloat in GP

There exists several theories that intend to explain code bloat :

– the *introns* theory states that code bloat acts as a protective mechanism in order to avoid the destructive effects of operators once relevant solutions have been is found (Nordin & Banzhaf, 1995; McPhee & Miller, 1995; Blickle & Thiele, 1994). Introns are pieces of code that have no influence on the fitness : either sub-programs that are never executed, or sub-programs which have no effect ;
– the *fitness causes bloat* theory relies on the assumption that there is a greater probability to find a bigger program with the same behavior (i.e. semantically equivalent) than to find a shorter one. Thus, once a good solution is found, programs naturally tends to grow because of fitness pressure (Langdon & Poli, 1997). This theory states that code bloat is operator-independent and may happen for any variable length representation-based algorithm. As a consequence, code bloat is not to be limited to population-based stochastic algorithm (such as GP), but may be extended to many algorithms using variable length representation (Langdon, 1998) ;
– the *removal bias* theory states that removing longer sub-programs is more tacky than removing shorter ones (because of possible destructive consequence), so there is a natural bias that benefits to the preservation of longer programs (Soule, 2002).

While it is now considered that each of these theories somewhat captures part of the problem (Banzhaf & Langdon, 2002), there has not been any definitive global explanation of the bloat phenomenon. At the same time, no definitive practical solution has been proposed that would avoid the drawbacks of bloat (increasing evaluation time of large trees) while maintaining the good performances of GP on difficult problems. Some common solutions rely either on specific operators (e.g. size-fair crossover (Langdon, 2000), or different Fair Mutation (Langdon *et al.*, 1999)), on some parsimony-based penalization of the fitness (Soule & Foster, 1998) or on abrupt limitation of the program size such as the one originally used by Koza (Koza, 1992). Some other more particular solutions have been proposed but are not widely used yet (Ratle & Sebag., 2001; Silva & Almeida, 2003; Luke & Panait, 2002).

3 Context and main results

In this paper, we intend to use Statistical Learning Theory to study code bloat, and to try to help designing algorithm that do not suffer from excessive code bloat, if at all possible.

However, the main goal of Statistical Learning Theory is to study the convergence of learning algorithms for Machine Learning problems with respect to the number of available examples and the complexity of the hypothesis space. In the framework of this

work – symbolic regression using GP – such results amount to study the algorithms with respect to the number of fitness cases and the allowed size of the GP trees.

3.1 Universal Consistency

In this paper, we intend to prove, under some sufficient conditions, that the solution given by GP actually converges, when the number of examples goes to infinity, toward the actual function used to generate the examples. This property is known in Statistical Learning as **Universal Consistency**. Note that this notion is a slightly different from that of Universal Approximation, that people usually refer to when doing symbolic regression in GP : because polynomial for instance are known to be able to approximate any continuous function, GP search using operators $\{+, *\}$ is also assumed to be able to approximate any continuous function. However, Universal Consistency is concerned with the behavior of the algorithm when the number of examples goes to infinity : being able to find a polynomial that approximates a given function at any arbitrary precision does not imply that any interpolation polynomial built from an arbitrary set of sample points will converge to that given function when the number of points goes to infinity.

But going back to bloat, and sticking to the polynomial example, it is also clear that the degree of the interpolation polynomial of a set of examples increases linearly with the number of examples. This leads us to start our bloat analysis by defining two kinds of bloat.

3.2 Structural vs. functional bloat

On the one hand, we define the **structural bloat** as the code bloat that unavoidably takes place when at least one optimal solution (a function that exactly matches all possible examples) does not lie in the search space. In such a situation, optimal solutions of increasing accuracy will also exhibit an increasing complexity, as larger and larger code will be generated in order to better approximate the target function. The extreme case of structural bloat has also been demonstrated in (Gustafson *et al.*, 2004). The authors use some polynomial functions of increasing difficulty, and demonstrate that a precise fit can only be obtained through an increased bloat (see also (Daida, 2001) for related issues about problem complexity in GP).

On the other hand, we define the **functional bloat** as the bloat that takes place when programs length keeps on growing even though an optimal solution (of known complexity) does lie in the search space. In order to clarify this point, let us use a simple symbolic regression problem defined as follow : given a set $\mathcal{S}$ of *examples*, the goal is to find a function f (here, a GP-tree) that minimized the Least Square Error (or LSE). If we intend to approximate a polynomial (ex. : $14 * x^2$), we may observe code bloat since it is possible to find arbitrarily long polynoms that gives the exact solution (ex. : $14 * x^2 + 0 * x^3 + ...$). Most of the works cited in section 2 are in fact concerned with functional bloat which is the most simple, yet already problematic, kind of bloat.

3.3 Overview of results

In section 5, we shall investigate the Universal Consistency of Genetic Programming algorithm, and study in detail structural and functional bloat that might take place when searching program spaces using GP.
A formal and detailed definition of the program space that will be assumed for GP is given in Lemma 1, section 5, and two types of results will then be derived :

- *Universal Consistency* results, i.e. does the probability of misclassification of the solution given by GP converges to the optimal probability of misclassification when the number of examples goes to infinity ?
- Bloat-related results, first regarding structural bloat, that will be proved to be incompatible with accuracy, and second with respect to functional bloat, for which the consequences of introducing various types of fitness penalization and/or bound on the complexity of the programs on the behavior of the complexity of the solution will be thoroughly studied.

Let us now state precisely, yet informally, our main results :

- First, as already mentioned, we will precisely define the set of programs under examination, and prove that such a search space fulfills the conditions of the standard theorems of Statistical Learning Theory listed in Section 4.
- Applying those theorems will immediately lead to a first Universal Consistency result for GP, provided that some penalization for complexity is added to the fitness (Theorem 3)
- The first bloat-related result, Proposition 4, unsurprisingly proves that if the optimal function does not belong to the search space, then converging to the optimal error implies that the complexity of the empirical optimal solution goes to infinity (unavoidable structural bloat).
- Theorem 5 is also a negative result about bloat, as it proves that even if the optimal function belongs to the search space, minimizing the LSE alone might lead to (structural) bloat (i.e. the complexity of the empirical solutions goes to infinity with the sample size).
- But the last two theorems (5' and 6) are the best positive results one could expect considering the previous findings : it is possible to carefully adjust the parsimony pressure so as to obtain both Universal Consistency and bounds on the complexity of the empirical solution (i.e. no bloat).

Note that, though all proofs in Section 5 will be stated and proved in the context of classification (i.e. find a function from $\mathbb{R}^d$ into $\{0, 1\}$), their generalization to regression (i.e. find a function from $\mathbb{R}^d$ into $\mathbb{R}$) is straightforward.

3.4 Discussion

First of all, it is important to note that all those results in fact study the solution given by perfectly successful GP runs on the search space at hand : given a set of examples and a fitness function based on the the Least Square Error (and possibly including some parsimony penalization), it will be assumed that GP does find one program in that search space that globally minimizes this fitness — and it is the behavior of this ideal solution when the number of examples goes to infinity that is theoretically studied.

Or course, we all know that GP is not such an ideal search procedure, and hence such results might look rather far away from GP practice, where the user desperately tries to find a program that gives a reasonably low empirical approximation error. Nevertheless, Universal Consistency is vital for the practitioner too : indeed, it would be totally pointless to fight to approximate an empirically optimal function without any guarantee that this empirical optimum is anywhere close to the ideal optimal solution we are in fact looking for.

Furthermore, the bloat-related results give some useful hints about the type of parsimony that has a chance to efficiently fight the unwanted bloat, while maintaining the Universal Consistency property – though some actual experiments will have to be run to confirm the usefulness of those theoretical hints.

4 Elements of Learning theory

In the frameworks of regression and classification, Statistical Learning Theory (Vapnik, 1995) is concerned with giving some bounds on the generalization error (i.e. the error on yet unseen data points) in terms of the actual empirical error (the LSE error above) and some fixed quantity depending only on the search space. More precisely, we will use here the notion of *Vapnik-Chervonenkis dimension* (in short, VCdim) of a function space, that somehow gives bounds on the variance of possible better solutions of the regression problem than the one obtained from the limited set of examples.

Consider a set of s examples $(x_i, y_i)_{i\in\{1,\dots,s\}}$. These examples are drawn from a distribution P on the couple (X, Y). They are independent identically distributed, $Y = \{0, 1\}$ (classification problem), and typically $X = \mathbb{R}^d$ for some dimension d.
For any function f, define the *loss* $L(f)$ to be the expectation of $|f(X) - Y|$. Similarly, define the *empirical loss* $\hat{L}(f)$ as the loss observed on the examples : $\hat{L}(f) = \frac{1}{s}\sum_i |f(x_i) - y_i|$.
Finally, define L^*, the *Bayes error*, as the smallest possible generalization error for any mapping from X to $\{0, 1\}$.

The following 4 theorems are well-known in the Statistical Learning community :

Theorem A (Devroye *et al.*, 1997, Th. 12.8, p206) :
Consider $\mathcal{F}$ a family of functions from a domain X to $\{0, 1\}$ and V its VC-dimension. Then, for any $\epsilon > 0$

$$P(\sup_{P\in\mathcal{F}} |L(P) - \hat{L}(P)| \geq \epsilon) \leq 4\exp(4\epsilon + 4\epsilon^2)s^{2V}\exp(-2s\epsilon^2)$$

and for any $\delta \in]0, 1]$

$$P(\sup_{P\in\mathcal{F}} |L(P) - \hat{L}(P)| \geq \epsilon(s, V, \delta)) \leq \delta$$

where $\epsilon(s, V, \delta) = \sqrt{\frac{4-\log(\delta/(4s^{2V}))}{2s-4}}$.

Other forms of this theorem have no $\log(n)$ factor ; they are known as Alexander's bound, but the constant is so large that this result is not better than the result above unless s is huge ((Devroye *et al.*, 1997, p207)) : if $s \geq 64/\epsilon^2$,

$$P(\sup_{P\in\mathcal{F}} |L(P) - \hat{L}(P)| \geq \epsilon) \leq 16(\sqrt{s}\epsilon)^{4096V} \exp(-2s\epsilon^2)$$

We classically derive the following result from theorem A :

Theorem A' :

Consider $\mathcal{F}_s$ for $s \geq 0$ a family of functions from a domain X to $\{0,1\}$ and V_s its VC-dimension. Then,

$$\sup_{P\in\mathcal{F}_s} |L(P) - \hat{L}(P)| \to 0 \text{ as } s \to \infty$$

almost surely whenever $V_s = o(s/\log(s))$.

Proof :

We use the classical Borell-Cantelli lemma[1], for any $\epsilon \in [0,1]$:

$$\sum_{s\geq 64/\epsilon^2} P(|L(P) - \hat{L}(P)| > \epsilon) \leq 16 \sum_{s\geq 64/\epsilon^2} (\sqrt{s}\epsilon)^{4096V_s} \exp(-2s\epsilon^2)$$

$$\leq 16 \sum_{s\geq 64/\epsilon^2} \exp(4096V_s(\log(\sqrt{s}) + \log(\epsilon)) - 2s\epsilon^2)$$

which is finite as soon as $V_s = o(s/\log(s))$.

Theorem B in (Devroye *et al.*, 1997, Th. 18.2, p290) :

Let $\mathcal{F}_1$, ..., $\mathcal{F}_k$ ... with finite VC-dimensions V_1, ..., V_k, ... Let $\mathcal{F} = \cup_n \mathcal{F}_n$. Then, being given s examples, consider $\hat{P} \in \mathcal{F}_s$ minimizing the empirical risk $\hat{L}$ among F_s. Then, if $V_s = o(s/log(s))$ and $V_s \to \infty$,

$$P(L(\hat{P}) \leq \hat{L}(\hat{P}) + \epsilon(s, V_s, \delta)) \geq 1 - \delta$$

$$P(L(\hat{P}) \leq \inf_{P\in\mathcal{F}_s} L(P) + 2\epsilon(s, V_s, \delta)) \geq 1 - \delta$$

$$\text{and } L(\hat{P}) \to \inf_{P\in\mathcal{F}} L(P) \text{ a.s.}$$

Note that for a well chosen family of functions (typically, programs), $\inf_{P\in\mathcal{F}} L(P) = L^*$ for any distribution ; so, theorem B leads to universal consistency (asymptotic minimization of the error rate for any distribution), for a well-chosen family of functions.

Theorem C (8.14 and 8.4 in (Antony & Bartlett, 1999)) :

Let $H = \{x \mapsto h(a,x); a \in R^{d'}\}$ where h can be computed with at most t' operations among

- $\alpha \mapsto \exp(\alpha)$;
- $+, -, \times, /$;
- jumps conditioned on $>, \geq, =, \leq, =$;
- output 0 ;
- output 1.

Then : $$VCdim(H) \leq t'^2 d'(d' + 19\log_2(9d'))$$.

Furthermore, if $\exp(.)$ is used at most q' times, and if there are at most t' operations executed among arithmetic operators, conditional jumps, exponentials,

$$\pi(H,m) \leq 2^{(d'(q'+1))2/2}(9d'(q'+1)2^t)^{5d'(q'+1)}(em(2^{t'}-2)/d')^{d'}$$

where $\pi(H,m)$ is the m^{th} shattering coefficient of H, and hence

$$VCdim(H) \leq (d'(q'+1))^2 + 11d'(q'+1)(t' + \log_2(9d'(q'+1)))$$

[1] If $\sum_n P(X_n > \epsilon)$ is finite for any $\epsilon > 0$ and $X_n > 0$, then $X_n \to 0$ almost surely.

Finally, if $q = 0$ then $VCdim(H) \leq 4d'(t' + 2)$.

Theorem D : structural risk minimization, (Devroye *et al.*, 1997) p. 294
Let $\mathcal{F}_1$, ..., $\mathcal{F}_k$ ... with finite VC-dimensions V_1, ..., V_k, ... Let $\mathcal{F} = \cup_n \mathcal{F}_n$. Assume that all distribution lead to $L_{\mathcal{F}} = L^*$ where L^* is the optimal possible error. Then, given s examples, consider $f \in \mathcal{F}$ minimizing $\hat{L}(f) + \sqrt{\frac{32}{s} V(f) \log(e \times s)}$, where $V(f)$ is V_k with k minimal such that $f \in \mathcal{F}_k$. Then :

• if additionally one optimal function belongs to $\mathcal{F}_k$,
then for any s and ϵ such that $V_k \log(e \times s) \leq s\epsilon^2/512$,
generalization error is lower than ϵ with probability at most
$\Delta \exp(-s\epsilon^2/128) + 8s^{V_k} \times \exp(-s\epsilon^2/512)$
where $\Delta = \sum_{j=1}^{\infty} \exp(-V_j)$ is assumed finite.

• the generalization error, with probability 1, converges to L^*.

5 Results

This section presents in details results that have been already surveyed in Section 3. They make an intensive use of the results of Statistical Learning Theory presented in the previous section.

More precisely, Lemma 1 and Lemma 1' define precisely the space of program considered here, and carefully show that it satisfies the hypotheses of Theorems A-C of section 3. This allows us to evaluate the VC-dimension of sets of programs, stated in Theorem 2. Then, announced results are derived.

Finally, next we propose a new approach combining an a priori limit on VC-dimension (i.e. *size limit*) and a complexity penalization (i.e. *parsimony pressure*) and state in theorem 6 that this leads to both universal consistency and convergence to an optimal complexity of the program (i.e. *no-bloat*).

We first recall some classical results of learning theory.

Lemma 1 :

Let F be the set of functions which can be computed with at most t operations among :

- operations $\alpha \mapsto \exp(\alpha)$ (at most q times) ;
- operations $+$, $-$, $\times$, $/$;
- jumps conditioned on $>$, $\geq$, $=$, $\leq$, $=$;

and

- output 0 ;
- output 1 ;
- labels for jumps ;
- at mosts m constants ;
- at most z variables

by a program with at most n lines.

We note $\log_2(x)$ the integer part (ceil) of $\log(x)/\log(2)$. Then F is included in H as defined in theorem C, for a given P with $t' = t + t\ max(3 + \log_2(n) + \log_2(z), 7 + 3\log_2(z)) + n(11 + max(9log_2(z), 0) + max(3log_2(z) - 3, 0))$, $q' = q$, $d' = 1 + m$.

Proof :
We define a program as in theorem above that can emulate any of these programs, with at most $t' = t + t\,max(3 + \log_2(n) + \log_2(z), 7 + 3\log_2(z)) + n(11 + max(9log_2(z), 0) + max(3log_2(z) - 3, 0))$, $q' = q$, $d' = 1 + m$.
The program is as follows :

- label "inputs"
- initialize $variable(1)$ at value $x(1)$
- initialize $variable(2)$ at value $x(2)$
- ...
- initialize $variable(dim(x))$ at value $x(dim(x))$
- label "constants"
- initialize $variable(dim(x) + 1)$ at value a_1
- initialize $variable(dim(x) + 2)$ at value a_2
- ...
- initialize $variable(dim(x) + m)$ at value a_m
- label "Decode the program into c"
- operation decode c
- label "Line 1"
- operation $c(1, 1)$ with variables $c(1, 2)$ and $c(1, 3)$ and $c(1, 4)$
- label "Line 2"
- operation $c(2, 1)$ with variables $c(2, 2)$ and $c(2, 3)$ and $c(2, 4)$
- ...
- label "Line n"
- operation $c(n, 1)$ with variables $c(n, 2)$and $c(n, 3)$ and $c(n, 4)$
- label "output 0"
- output 0
- label "output 1"
- output 1

"operation decode c" can be developed as follow. Indeed, we need m real numbers, for parameters, and $4n$ integers $c(.,.)$, that we will encode as only one real number in $[0, 1]$ as follows :
1. let $y \in [0, 1]$
2. for each $i \in [1, \ldots n]$:

- $c(i, 1) = 0$
- $y = y * 2$
- if $(y > 1)$ then $\{\ c(i, 1) = 1\,;\, y = y - 1\ \}$
- $y = y * 2$
- if $(y > 1)$ then $\{\ c(i, 1) = c(i, 1) + 2\,;\, y = y - 1\ \}$
- $y = y * 2$
- if $(y > 1)$ then $\{\ c(i, 1) = c(i, 1) + 4\,;\, y = y - 1\ \}$

3. for each $j \in [2, 4]$ and $i \in [1, \ldots n]$:

- $c(i, j) = 0$
- $y = y * 2$
- if $(y > 1)$ then $\{\ c(i, j) = 1\,;\, y = y - 1\ \}$

- $y = y * 2$
- if $(y > 1)$ then $\{\ c(i,j) = c(i,j) + 2\,;\, y = y - 1\ \}$
- $y = y * 2$
- if $(y > 1)$ then $\{\ c(i,j) = c(i,j) + 4\,;\, y = y - 1\ \}$
- ...
- $y = y * 2$
- if $(y > 1)$ then $\{\ c(i,j) = c(i,j) + 2^{log_2(z)-1}\,;\, y = y - 1\ \}$

The cost of this is $n \times (3 + max(3 \times log_2(z), 0))$ "if then", and $n \times (3 + max(3 \times log_2(z), 0))$ operators $\times$, and $n(2 + max(3(log_2(z) - 1), 0))$ operators $+$, and $n \times (3 + max(3 \times log_2(z), 0))$ operators $-$. The overall sum is bounded by $n(11 + max(9\log_2(z), 0) + max(3log_2(z) - 3, 0))$.

Lemma 1' : "operation $c(i,1)$ with variables c(i,2) and c(i,3)" can be developed as follows :

- if $c(i,1) == 0$ then goto "output1"
- if $c(i,1) == 1$ then goto "output 0"
- if $c(i,2) == 1$ then $c = variable(1)$
- if $c(i,2) == 2$ then $c = variable(2)$
- ...
- if $c(i,2) == z$ then $c = variable(z)$
- if $c(i,1) == 7$ then goto "Line c" (must be encoded by dichotomy with $log_2(n)$ lines)
- if $c(i,1) == 6$ then goto "exponential(i)"
- if $c(i,3) == 1$ then $b = variable(1)$
- if $c(i,3) == 2$ then $b = variable(2)$
- ...
- if $c(i,3) == z$ then $b = variable(z)$
- if $c(i,1) == 2$ then $a = c + b$
- if $c(i,1) == 3$ then $a = c - b$
- if $c(i,1) == 4$ then $a = c \times b$
- if $c(i,1) == 5$ then $a = c/b$
- if $c(i,4) == 1$ then $variable(1) = a$
- if $c(i,4) == 2$ then $variable(2) = a$
- ...
- if $c(i,4) == z$ then $variable(z) = a$
- label "endOfInstruction(i)"

For each such instruction, at the end of the program, we add three lines of the following form :

- label "exponential(i)"
- $a = exp(c)$
- goto "endOfInstruction(i)"

Each sequence of the form "if x=... then" (p times) can be encoded by dichotomy with $log_2(p)$ tests "if ... then goto".

Theorem 2 :

Let F be the set of programs as in lemma 1, where $q' \geq q$, $t' \geq t + t\ max(3 + \log_2(n) +$

$\log_2(z), 7+3\log_2(z))+n(11+max(9log_2(z),0)+max(3log_2(z)-3,0))$, $d' \geq 1+m$.

$$VCdim(H) \leq t'^2 d'(d' + 19\log_2(9d'))$$

$$VCdim(H) \leq (d'(q'+1))^2 + 11d'(q'+1)(t' + \log_2(9d'(q'+1)))$$

If $q = 0$ (no exponential) then $VCdim(H) \leq 4d'(t'+2)$.

Proof : Just plug Lemmas 1 and 1' in Theorem C ■

Theorem 3 :

Consider q_f, t_f, m_f, n_f and z_f integer sequences, non-decreasing functions of f. Define $V_f = VCdim(H_f)$, where H_f is the set of programs with at most t_f lines executed, with z_f variables, n_f lines, q_f exponentials, and m_f constants.

Then with $q'_f = q_f$, $t'_f = t_f + t_f\, max(3 + \log_2(n_f) + \log_2(z_f), 7 + 3\log_2(z_f)) + n_f(11 + max(9log_2(z_f), 0) + max(3log_2(z_f) - 3, 0))$, $d'_f = 1 + m_f$,

$$V_f = (d'_f(q'_f + 1))^2 + 11d'_f(q'_f + 1)(t'_f + \log_2(9d'_f(q'_f + 1)))$$

or, if $\forall f\ q_f = 0$ then define $V_f = 4d'_f(t'_f + 2)$.

Then, being given s examples, consider $f \in \mathcal{F}$ minimizing $\hat{L}(f) + \sqrt{\frac{32}{s}V(f)\log(e \times s)}$, where $V(f)$ is the min of all k such that $f \in \mathcal{F}_k$.

Then, if $\Delta = \sum_{j=1}^{\infty} \exp(-V_j)$ is finite,

- the generalization error, with probability 1, converges to L^*.
- if one optimal rule belongs to $\mathcal{F}_k$, then for any s and ϵ such that $V_k \log(e \times s) \leq s\epsilon^2/512$, the generalization error is lower than ϵ with probability at most $\Delta \exp(-s\epsilon^2/128) + 8s^{V_k} \times \exp(-s\epsilon^2/512)$ where $\Delta = \sum_{j=1}^{\infty} \exp(-V_j)$ is assumed finite.

Proof : Just plug theorem D in theorem 2. ■

We now prove the non-surprising fact that if it is possible to approximate the optimal function (the Bayesian classifier) without reaching it exactly, then the "complexity" of the program runs to infinity as soon as there is convergence of the generalization error to the optimal one.

Proposition 4 :

Consider P_s a sequence of functions such that $P_s \in \mathcal{F}_{V(s)}$, with $\mathcal{F}_1 \subset \mathcal{F}_2 \subset \mathcal{F}_3 \subset \ldots$, where $\mathcal{F}_V$ is a set of functions from X to $\{0,1\}$ with VC-dimension bounded by V.
Define $L_V = \inf_{P \in \mathcal{F}_V} L(P)$ and $V(P) = \inf\{V / P \in \mathcal{F}_V\}$
and suppose that $\forall V\ L_V > L^*$.

Then

$$(L(P_s) \overset{s\to\infty}{\longrightarrow} L^*) \Longrightarrow (V(P_s) \overset{s\to\infty}{\longrightarrow} \infty)$$

Proof :

Define $\epsilon(V) = L_V - L^*$. Assume that $\forall V\ \epsilon(V) > 0$. ϵ is necessarily non-increasing.

Consider V_0 a positive integer; let us prove that if n is large enough, then $V(P_s) \geq V_0$.

There exists ϵ_0 such that $\epsilon(V_0) > \epsilon_0 > 0$.

For s large enough, $L(P_s) \leq L^* + \epsilon_0$,
hence $L_{V_s} \leq L^* + \epsilon_0$,
hence $L^* + \epsilon(V_s) \leq L^* + \epsilon_0$,

hence $\epsilon(V_s) \leq \epsilon_0$,
hence $V_s > V_0$. ■

We now show that the usual procedure defined below, consisting in defining a maximum VC-dimension depending upon the sample size (as usually done in practice and as recommended by theorem B) and then using a moderate family of functions, leads to bloat. With the same hypotheses as in theorem B, we can state

Theorem 5 (bloat theorem for empirical risk minimization with relevant VC-dimension) :
Let $\mathcal{F}_1$, ..., $\mathcal{F}_k$ ... non-empty sets of functions with finite VC-dimensions V_1, ..., V_k, ... Let $\mathcal{F} = \cup_n \mathcal{F}_n$. Then, given s examples, consider $\hat{P} \in \mathcal{F}_s$ minimizing the empirical risk $\hat{L}$ in F_s.
From Theorem B we already know that
if $V_s = o(s/log(s))$ and $V_s \to \infty$,
then $P(L(\hat{P}) \leq \hat{L}(\hat{P}) + \epsilon(s, V_s, \delta)) \geq 1 - \delta$,
and $L(\hat{P}) \to \inf_{P \in \mathcal{F}} L(P)$ a.s..

We will now state that if $V_s \to \infty$,
and noting $V(f) = min\{V_k / f \in \mathcal{F}_k\}$, then
$\forall V_0, P_0 > 0$
$\exists P$, distribution of probability on X and Y, such that
$\exists g \in \mathcal{F}_1$ such that $L(g) = L^*$
and for s sufficiently large $P(V(\hat{P}) \leq V_0) \leq P_0$.

Remarks :
The result in particular implies that for any V_0, there is a distribution of examples such that for some g with $V(g) = V_1$ and $L(g) = L^*$, with probability 1, $V(\hat{f}) \geq V_0$ infinitely often as s increases.

Proof (of the part which is not theorem B) :
Consider $V_0 > 0$ and $P_0 > 0$. Consider α such that $(e\alpha/2^\alpha)^{V_0} \leq P_0/2$. Consider s such that $V_s \geq \alpha V_0$. Let $d = \alpha V_0$.

Consider $x_1, \ldots, x_d$ d points shattered by $\mathcal{F}_d$; such a family of d points exist, by definition of $\mathcal{F}_d$.

Define the probability measure P by the fact that X and Y and independent and $P(Y = 1) = \frac{1}{2}$ and $P(X = x_i) = \frac{1}{d}$.

Then, the following holds, with Q the empirical distribution (the average of Dirac masses on the x_i's) :

1. no empty x_i's :
$$P(E_1) \to 0$$
where E_1 is the fact that $\exists i / Q(X = x_i) = 0$, as $s \to \infty$.
2. no equality :
$$P(E_2) \to 0$$
where E_2 is the fact that E_1 occurs or $\exists i / Q(Y = 1 | X = x_i) = \frac{1}{2}$.
3. the best function is not in $\mathcal{F}_{V_0}$:
$$P(E_3 | E_2 \text{ does not hold}) \leq S(d, d/\alpha)/2^d$$

where E_3 is the fact that $\exists g \in \mathcal{F}_{d/\alpha=V_0} / \hat{L}(g) = inf_{\mathcal{F}_d} \hat{L}$, with $S(d, d/\alpha)$ the relevant shattering coefficient, i.e. the cardinal of $\mathcal{F}_{d/\alpha}$ restricted to $\{x_1, \ldots, x_d\}$.

We now only have to use classical results. It is well known in VC-theory that $S(a,b) \leq (ea/b)^b$ (see for example (Devroye *et al.*, 1997, chap.13)), hence $S(d, d/\alpha) \leq (ed/(d/\alpha))^{d/\alpha}$ and

$$P(E_3|E_2 \text{ does not hold}) \leq (e\alpha)^{d/\alpha}/2^d \leq P_0/2$$

and if n is sufficiently large to ensure that $P(E_2) \leq P_0/2$ (we have proved above that $P(E_2) \to 0$ as $s \to \infty$) then

$$P(E_3) \leq P(E_3|\neg E_2) \times P(\neg E_2) + P(E_2)$$

$$\leq P(E_3|\neg E_2) + P(E_2) \leq P_0/2 + P_0/2 \leq P_0 \quad \blacksquare$$

We now show that, on the other hand, it is possible to optimize a compromise between optimality and complexity in an explicit manner (e.g., replacing 1 % precision with 10 lines of programs or 10 minutes of CPU) :

Theorem 5' (bloat-control theorem for regularized empirical risk minimization with relevant VC-dimension) :
Let $\mathcal{F}_1$, ..., $\mathcal{F}_k$ ... be non-empty sets of functions with finite VC-dimensions V_1, ..., V_k, ... Let $\mathcal{F} = \cup_n \mathcal{F}_n$. Consider W a user-defined complexity penalization term. Then, being given s examples, consider $P \in \mathcal{F}_s$ minimizing the regularized empirical risk $\hat{\tilde{L}}(P) = \hat{L}(P) + W(P)$ among F_s. If $V_s = o(s/log(s))$ and $V_s \to \infty$, then $\tilde{L}(\hat{P}) \to \inf_{P \in \mathcal{F}} \tilde{L}(P)$ a.s. where $\tilde{L}(P) = L(P) + W(P)$.

Proof :

$$\sup_{P \in \mathcal{F}_s} |\hat{\tilde{L}}(P) - \tilde{L}(P)|$$
$$\leq \sup_{P \in \mathcal{F}_s} |\hat{L}(P) - L(P)|$$
$$\leq \epsilon(s, V_s) \to 0 \text{ almost surely, by theorem A'}$$

Hence the expected result. $\blacksquare$

Theorem 5' shows that, using a relevant a priori bound on the complexity of the program and adding a user-defined complexity penalization to the fitness, can lead to convergence toward a user-defined compromise between classification rate and program complexity (i.e. we ensure almost sure convergence to a compromise of the form "λ_1 CPU time + λ_2 misclassification rate + λ_3 number of lines", where the λ_i are user-defined.

Remark : the drawback of this approach is that we have lost universal consistency and consistency (in the general case, the misclassification rate in generalization will not converge to the Bayes error, and whenever an optimal program exists, we will not necessarily converge to its efficiency).

We now turn our attention to a more complicated case where we want to ensure universal consistency, but we want to avoid a non-necessary bloat ; e.g., we require that if an optimal program exists in our family of functions, then we want to converge to its error rate, without increasing the complexity of the program.

We are now going to consider a merge between regularization and bounding of the VC-dimension ; we penalize the complexity (e.g., length) of programs by a penalty term $R(s, P) = R(s)R'(P)$ depending upon the sample size and upon the program ; $R(.,.)$

is user-defined and the algorithm will look for a classifier with a small value of both R' and L.

We study both the universal consistency of this algorithm (i.e. $L \rightarrow L^*$) and the no-bloat theorem (i.e. $R' \rightarrow R'(P^*)$ when P^* exists).

Theorem 6 :

Let $\mathcal{F}_1$, ..., $\mathcal{F}_k$ ... with finite VC-dimensions V_1, ..., V_k, ... Let $\mathcal{F} = \cup_n \mathcal{F}_n$. Define $V(P) = V_k$ with $k = \inf\{t | P \in \mathcal{F}_t\}$. Define $L_V = inf_{P \in \mathcal{F}_V} L(P)$. Consider $V_s = o(log(s))$ and $V_s \rightarrow \infty$. Consider $\hat{P}$ minimizing $\hat{\tilde{L}}(P) = \hat{L}(P) + R(s,P)$ in $\mathcal{F}_s$ and assume that $R(s,.) \geq 0$.

Then (consistency), whenever $\sup_{P \in \mathcal{F}_{V_s}} R(s,P) = o(1)$, $L(\hat{P}) \rightarrow \inf_{P \in \mathcal{F}} L(P)$ almost surely (note that for well chosen family of functions, $\inf_{P \in \mathcal{F}} L(P) = L^*$)

Assume that $\exists P^* \in \mathcal{F}_{V^*}$ $L(P^*) = L^*$. Then with $R(s,P) = R(s)R'(P)$ and with $R'(s) = \sup_{P \in \mathcal{F}_{V_s}} R'(P)$:

1. **non-asymptotic no-bloat theorem :** $R'(\hat{P}) \leq R'(P^*) + (1/R(s))2\epsilon(s, V_s, \delta)$ with probability at least $1 - \delta$ (this result is in particular interesting for $\epsilon(s, V_s, \delta)/R(s) \rightarrow 0$, what is possible for usual regularization terms as in theorem D,
2. **almost-sure no-bloat theorem :** if $R(s)s^{(1-\alpha)/2} = O(1)$, then almost surely $R'(\hat{P}) \rightarrow R'(P^*)$ and if $R'(P)$ has discrete values (such as the number of instructions in P or many complexity measures for programs) then for s sufficiently large, $R'(\hat{P}) = R'(P^*)$.
3. **convergence rate :** with probability at least $1 - \delta$,

$$L(\hat{P}) \leq inf_{P \in \mathcal{F}_{V_s}} L(P) + \underbrace{R(s)R'(s)}_{=o(1) \text{ by hypothesis}} + 2\epsilon(s, V_s, \delta)$$

   where $\epsilon(s, V, \delta) = \sqrt{\frac{4 - \log(\delta/(4s^{2V}))}{2s-4}}$ is an upper bound on $\epsilon(s,V) = \sup_{f \in \mathcal{F}_V} |\hat{L}(f) - L(f)|$ (given by theorem A), true with probability at least $1 - \delta$.

Remarks : The usual $R(s,P)$ as used in theorem D or theorem 3 provides consistency and non-asymptotic no-bloat. A stronger regularization leads to the same results, plus almost sure no-bloat. The asymptotic convergence rate depends upon the regularization. The result is not limited to genetic programming and could be used in other areas.

As shown in proposition 4, the no-bloat results require the fact that $\exists V^* \exists P^* \in \mathcal{F}_{V^*}$ $L(P^*) = L^*$.

Interestingly, the convergence rate is reduced when the regularization is increased in order to get the almost sure no-bloat theorem.

Proof :

Define $\epsilon(s,V) = \sup_{f \in \mathcal{F}_V} |\hat{L}(f) - L(f)|$.

Let us prove the consistency : For any P,

$$\hat{L}(\hat{P}) + R(s,\hat{P}) \leq \hat{L}(P) + R(s,P)$$

On the other hand,

$$L(\hat{P}) \leq \hat{L}(\hat{P}) + \epsilon(s, V_s)$$

So :

$$L(\hat{P}) \leq (inf_{P \in \mathcal{F}_{V_s}}(\hat{L}(P) + R(s,P))) - R(s,\hat{P}) + \epsilon(s, V_s)$$

$$\leq (inf_{P \in \mathcal{F}_{V_s}}(L(P) + \epsilon(s,V_s) + R(s,P))) - R(s,\hat{P}) + \epsilon(s, V_s)$$

$$\leq (inf_{P \in \mathcal{F}_{V_s}}(L(P) + R(s,P))) + 2\epsilon(s, V_s)$$

as $\epsilon(s, V_s) \to 0$ almost surely[2] and $(inf_{P \in \mathcal{F}_{V_s}}(L(P) + R(s,P))) \to \inf_{P \in \mathcal{F}} L(P)$, we conclude that $L(\hat{P}) \to \inf_{P \in \mathcal{F}} L(P)$ a.s.

We now focus on the proof of the "no bloat" result :
By definition of the algorithm, for s sufficiently large to ensure $P^* \in \mathcal{F}_{V_s}$,

$$\hat{L}(\hat{P}) + R(s,\hat{P}) \leq \hat{L}(P^*) + R(s,P^*)$$

hence with probability at least $1 - \delta$,

$$R'(\hat{P}) \leq R'(P^*) + (1/R(s))(L^* + \epsilon(s, V_s, \delta) - L(\hat{P}) + \epsilon(s, V_s, \delta))$$

hence

$$R'(\hat{P}) \leq R'(V^*) + (1/R(s))(L^* - L(\hat{P}) + 2\epsilon(s, V_s, \delta))$$

As $L^* \leq L(\hat{P})$, this leads to the non-asymptotic version of the no-bloat theorem.

The almost sure no-bloat theorem is derived as follows.

$$R'(\hat{P}) \leq R'(P^*) + 1/R(s)(L^* + \epsilon(s, V_s) - L(\hat{P}) + \epsilon(s, V_s))$$

hence

$$R'(\hat{P}) \leq R'(P^*) + 1/R(s)(L^* - L(\hat{P}) + 2\epsilon(s, V_s))$$

$$R'(\hat{P}) \leq R'(P^*) + 1/R(s)2\epsilon(s, V_s)$$

All we need is the fact that $\epsilon(s, V_s)/R(s) \to 0$ a.s.

For any $\epsilon > 0$, we consider the probability of $\epsilon(s, V_s)/R(s) > \epsilon$, and we sum over $s > 0$. By the Borel-Cantelli lemma, the finiteness of this sum is sufficient for the almost sure convergence to 0.

The probability of $\epsilon(s, V_s)/R(s) > \epsilon$ is the probability of $\epsilon(s, V_s) > \epsilon R(s)$. By theorem A, this is bounded above by $O(\exp(2V_s \log(s) - 2s\epsilon^2 R(s)^2))$. This has finite sum for $R(s) = \Omega(s^{-(1-\alpha)/2})$.

Let us now consider the convergence rate. Consider s sufficiently large to ensure $L_{V_s} = L^*$. As shown above during the proof of the consistency,

$$L(\hat{P}) \leq (inf_{P \in \mathcal{F}_{V_s}}(L(P) + R(s,P))) + 2\epsilon(s, V_s)$$

$$\leq (inf_{P \in \mathcal{F}_{V_s}}(L(P) + R(s)R'(P))) + 2\epsilon(s, V_s)$$

$$\leq inf_{P \in \mathcal{F}_{V_s}} L(P) + R(s)R'(s) + 2\epsilon(s, V_s)$$

so with probability at least $1 - \delta$,

$$\leq inf_{P \in \mathcal{F}_{V_s}} L(P) + R(s)R'(s) + 2\epsilon(s, V_s, \delta) \quad \blacksquare$$

[2] See theorem A'

6 conclusion

In this paper, we have proposed a theoretical study of an important issue in Genetic Programming known as code bloat. We have shown that GP trees used in symbolic regression (involving the four arithmetic operations, the exponential function, and ephemeral constants, as well as test and jump instructions) could be applied some classical results from Statistical Learning Theory. This has lead to two kinds of original outcomes : some results about Universal Consistency of GP, i.e. some guarantee that if GP converges to some (empirical) function, this function will be close from the optimal one if sufficiently enough examples are used ; and results about the bloat, both the unavoidable structural bloat in case the target ideal function is not included in the search space, and the functional bloat, for which we proved that it can – theoretically – be avoided by simultaneously bounding the length of the programs with some *ad hoc* bound) and using some parsimony pressure in the fitness function. Some negative results have been obtained, too, such as the fact though structural bloat was know to be unavoidable, functional bloat might indeed happen even when the target function does lie in the search space, but no parsimony pressure is used.

Interestingly enough, all those results (both positive and negative) about bloat are also valid in different contexts, such as for instance that of Neural Networks (the number of neurons replaces the complexity of GP programs). Moreover, results presented here are not limited to the scope of regression problems, but may be applied to variable length representation algorithms in different contexts such as control or identification tasks.

Further research will first be concerned with experimental validations of those theoretical results, emphasizing their usefulness for practitioners (see the discussion Section 3.4). However, we are aware that the balance between both parsimony factors (the bound on the complexity of the search space, and the penalization factor in the fitness) might be tricky to tune, and the solution might prove to be highly problem-dependent.

Another extension of those results concerns noisy and dynamic fitness-es since most of them could probably be easily generalized to ϵ-convergence instead of actual convergence.

Finally, going back to the debate about the causes of bloat in practice, it is clear that our results can only partly explain the actual cause of bloat in a real GP run – and tends to give arguments to the "fitness causes bloat" explanation (Langdon & Poli, 1997). It might be possible to study the impact of size-preserving mechanisms (e.g. specific variation operators, like size-fair crossover (Langdon, 2000) or fair mutations (Langdon *et al.*, 1999)) as somehow contributing to the regularization term in our final result ensuring both Universal Consistency and no-bloat.

Acknowledgements

This work was supported in part by the PASCAL Network of Excellence.

Références

ANTONY M. & BARTLETT P. (1999). Neural network learning : Theoretical foundations,

cambridge university press.

BANZHAF W. & LANGDON W. B. (2002). Some considerations on the reason for bloat. *Genetic Programming and Evolvable Machines*, **3**(1), 81–91.

BLICKLE T. & THIELE L. (1994). Genetic programming and redundancy. In J. HOPF, Ed., *Genetic Algorithms Workshop at KI-94*, p. 33–38 : Max-Planck-Institut für Informatik.

DAIDA J. M. (2001). What makes a problem gp-hard ? analysis of a tunably difficult problem in genetic programming. *Genetic Programming and Evolvable Machines*, **2**(2), 165 – 191.

DEVROYE L., GYÖRFI L. & LUGOSI G. (1997). A probabilistic theory of pattern recognition, springer.

GUSTAFSON S., EKART A., BURKE E. & KENDALL G. (2004). Problem difficulty and code growth in Genetic Programming. *GP and Evolvable Machines*, **4**(3), 271–290.

KOZA J. R. (1992). *Genetic Programming : On the Programming of Computers by Means of Natural Selection*. Cambridge, MA, USA : MIT Press.

LANGDON W. B. (1998). The evolution of size in variable length representations. In *ICEC'98*, p. 633–638 : IEEE Press.

LANGDON W. B. (2000). Size fair and homologous tree genetic programming crossovers. *Genetic Programming And Evolvable Machines*, **1**(1/2), 95–119.

LANGDON W. B. & POLI R. (1997). Fitness causes bloat : Mutation. In J. KOZA, Ed., *Late Breaking Papers at GP'97*, p. 132–140 : Stanford Bookstore.

LANGDON W. B., SOULE T., POLI R. & FOSTER J. A. (1999). The evolution of size and shape. In L. SPECTOR, W. B. LANGDON, U.-M. O'REILLY & P. ANGELINE, Eds., *Advances in Genetic Programming III*, p. 163–190 : MIT Press.

LUKE S. & PANAIT L. (2002). Lexicographic parsimony pressure. In W. B. L. ET AL., Ed., *GECCO 2002 : Proceedings of the Genetic and Evolutionary Computation Conference*, p. 829–836 : Morgan Kaufmann Publishers.

MCPHEE N. F. & MILLER J. D. (1995). Accurate replication in genetic programming. In L. ESHELMAN, Ed., *Genetic Algorithms : Proceedings of the Sixth International Conference (ICGA95)*, p. 303–309, Pittsburgh, PA, USA : Morgan Kaufmann.

NORDIN P. & BANZHAF W. (1995). Complexity compression and evolution. In L. ESHELMAN, Ed., *Genetic Algorithms : Proceedings of the Sixth International Conference (ICGA95)*, p. 310–317, Pittsburgh, PA, USA : Morgan Kaufmann.

RATLE A. & SEBAG. M. (2001). Avoiding the bloat with probabilistic grammar-guided genetic programming. In P. C. ET AL., Ed., *Artificial Evolution VI* : Springer Verlag.

SILVA S. & ALMEIDA J. (2003). Dynamic maximum tree depth : A simple technique for avoiding bloat in tree-based gp. In E. C.-P. ET AL., Ed., *Genetic and Evolutionary Computation – GECCO-2003*, volume 2724 of *LNCS*, p. 1776–1787 : Springer-Verlag.

SOULE T. (2002). Exons and code growth in genetic programming. In J. A. F. ET AL., Ed., *EuroGP 2002*, volume 2278 of *LNCS*, p. 142–151 : Springer-Verlag.

SOULE T. & FOSTER J. A. (1998). Effects of code growth and parsimony pressure on populations in genetic programming. *Evolutionary Computation*, **6**(4), 293–309.

VAPNIK V. (1995). The nature of statistical learning theory, springer.

Introduction à l'extraction de l'information à partir des flux de données

Toufik Boudellal, David William Pearson

Equipe Universitaire de Recherche en Informatique de St-Etienne,
Faculté des Sciences et Techniques, Département Informatique
23, Rue Paul Michelon, 42023, St-Etienne, Cedex 2
toufik.boudellal@univ-st-etienne.fr
david.peasron@univ-st-etienne.fr

Le présent travail concerne l'extraction de l'information à partir des flux de données (EIFD) 'Mining Data Streams'. En effet,l'EIFD est un nouveau domaine principalement lié aux besoins réels d'applications qui analysent prestement et de manière continue ces flux de données (fd(s)). Les sources de ces données peuvent être par exemple des capteurs industriels qui génèrent périodiquement des observations (données sismiques, données atmosphériques, ... etc.). Les travaux actuels s'intéressent à : La description continue d'un certain ensemble de flux de sortie (fs) en fonction d'un certain ensemble de flux d'entrée (fe), identifier les données expirées, ... etc (Wang *et al.*, 2003; Fan *et al.*, 2004; Aggarwal *et al.*, 2004). Les principales difficultés dans ce domaine sont l'aspect infini des données, la multiplicité des flux et les types des données des flux. Nous contribuons dans ce domaine en introduisant des mesures de similarités adaptées au cas multi flux de données pour la description continue de données symboliques. Ces mesures ne supposent pas un domaine de définition initialement connu ni un précodage de ces données, possèdent un pouvoir d'extraction d'informations comparatif aux mesures existantes et sont réduites en complexité de calculs. Le paragraphe suivant présente un scénario possible.

Un nouveau paquet de données arrive au système, ce paquet de données est considéré comme le contexte informatif courant et constitué de l'ensemble des observations des flux d'entrée et de sortie. Un deuxième paquet arrive au système sous forme d'une requête, constitué uniquement des observations liées aux flux d'entrée. Les deux paquets forment la fenêtre d'informations. L'objectif principal est l'extraction des différentes similarités pour pouvoir prédire approximativement les observations liées aux flux de sortie. Les mesures de similarités qui existent actuellement se basent sur la théorie des statistiques ou la théorie de l'information. L'article (Tan *et al.*, 2004) propose une synthèse des différentes mesures, ces mesures présentent une grande corrélation entre elles. A titre comparatif, nous avons choisi les mesures de similarités de base et les plus pertinentes. La mesure de Pearson (P) considérée comme la base de plusieurs variantes, elle permet la construction des tables de contingences entre les différents flux pour découvrir des règles d'associations (Tan *et al.*, 2004). La mesure de Smyth et Goodman (S-G) (Tan *et al.*, 2004) est une entropie relative qui permet d'évaluer les différentes règles

d'association.
Nous proposons trois mesures de similarités (mesures de Boudellal-Pearson). Si on considère un seul flux de sortie alors la similarité sur la capacité de décision (DCP) exprime l'information qui lie les sous ensembles de symboles du flux de sortie par rapport aux flux d'entrée, sachant l'ordre des données sur la fenêtre d'information courante. La similarité sur l'incertitude de décision (DIS) mesure la contiguïté des symboles du flux de sortie, cette contiguïté est relative à chaque flux d'entrée. La similarité sur l'aptitude de décision (DAP) permet de mesurer le lien de chaque symbole du flux d'entrée aux symboles du même flux et au flux de sortie. La considération de la complexité de calcul des différentes mesures est d'une importance principale. La mesure de Smyth-Goodman et la corrélation de Pearson se basent sur la jonction des différents flux d'entrée, ensuite une jonction avec le flux de sortie. Contrairement à nos mesures qui se basent sur un critère additif entre les différents flux d'entrée. Pour comparer les différentes mesures, on a injecté les jeux des données standards (UCI) [1] dans un module aléatoire. Ce module permet de choisir aléatoirement des exemples pour former un paquet de données pour une taille donnée. Nous avons essayé l'expérience sur plusieurs tailles (10–100). Les jeux de données utilisés sont les suivants :"Tic-Tac-Toe", "Connect", "Champignons", "Cancer de seins". Le plan d'expériences utilisé cherche à présenter le pouvoir d'extraction d'informations des mesures suivant la taille de la fenêtre. Nous avons choisi pour les premières expériences deux flux d'entrée et un flux de sortie (vue la complexité de calcul des deux mesures (S-G),(P)). Ensuite, nous avons choisi d'autres expériences pour une comparaison locale à nos mesures. Le tableau suivant résume le classement ($\star\star\star$:meilleur) des différentes mesures . Pour conclure, On a proposé de nouvelles mesures dans le domaine EIFD, qui présentent un avantage considérable au niveau de la complexité de calcul et le pouvoir d'extraction. Notre perspective est le traitement des flux bruités.

	TicTacToe				Connect		Champignon	Cancer
	1-2	3-4	5-6	7-8	1-2	3-4	tous	tous
S-G	$\star\star$	$\star\star$	$\star\star\star$	$\star\star$	$\star\star$	$\star\star$	$--$	$--$
P	$\star$	$\star\star\star$	$\star$	$\star$	$\star$	$\star$	$--$	$--$
DCP	$\star\star\star$	$\star\star\star$	$\star\star\star$	$\star\star\star$	$\star\star\star$	$\star\star$	$\star\star\star$	$\star\star\star$
DIS	$\star\star$	$\star\star\star$	$\star\star$	$\star\star\star$	$\star\star$	$\star\star\star$	$\star\star$	$\star\star\star$
DAP	$\star\star\star$	$\star\star\star$	$\star\star$	$\star\star$	$\star\star$	$\star\star$	$\star\star\star$	$\star\star\star$

Références

AGGARWAL C. C., HAN J., WANG J. & YU P. S. (2004). On demand classification of data streams. In *KDD '04*, p. 503–508.

FAN W., AN HUANG Y., WANG H. & YU P. S. (2004). Active mining of data streams. In *SDM*, p. 457–461.

TAN P.-N., KUMAR V. & SRIVASTAVA J. (2004). Selecting the right objective measure for association analysis. *Information Systems*, **29**(4), 293–313.

WANG H., FAN W., YU P. S. & HAN J. (2003). Mining concept-drifting data streams using ensemble classifiers. In *KDD '03*, p. 226–235.

[1] UCI : 'Knowledge Discovery in Databases', http ://kdd.ics.uci.edu

Apprentissage non supervisé de motifs temporels, multidimensionnels et hétérogènes Application à la télésurveillance médicale

Florence Duchêne[1], Catherine Garbay[1] et Vincent Rialle[1,2]

[1] Laboratoire TIMC-IMAG, Faculté de médecine de Grenoble, France
[2] Département d'Informatique Médicale de l'hôpital Michallon, Grenoble, France
Florence.Duchene@sophia.inria.fr,
Catherine.Garbay@imag.fr, Vincent.Rialle@imag.fr

Résumé : On propose une méthode générique d'extraction non supervisée de motifs temporels dans des séries multidimensionnelles et hétérogènes, appliquée à l'apprentissage des comportements récurrents d'une personne à domicile.

Mots-clés : Fouille de séries temporelles, multidimensionnelles et hétérogènes, Motifs temporels, Apprentissage non supervisé, Télésurveillance médicale.

Pour la détection des évolutions critiques à long terme de personnes à domicile, on propose un système d'apprentissage d'un profil comportemental dans la vie quotidienne. Un écart de comportement par rapport à ce profil peut être inquiétant car significatif d'une dégradation de l'état de santé. Il s'agit d'extraire des *motifs* "haut niveau" de séquences "bas niveau" collectées de capteurs installés au domicile. Un *motif* est le représentant d'une classe de sous-séquences récurrentes, et correspond à un comportement type de la personne. Ce problème d'apprentissage a les caractéristiques suivantes :

1. **Séquences analysées –** Les séquences analysées sont multidimensionnelles, hétérogènes, et *mixtes* : elles contiennent à la fois des sous-séquences représentatives de *motifs* et des "*non motifs*".
2. **Méthode utilisée –** L'extraction de motifs est *non supervisée* pour s'adapter aux spécificités individuelles et au manque de connaissances *a priori*.
3. **Séquences temporelles –** Les séquences analysées sont multidimensionnelles, hétérogènes, et *mixtes* – i.e. contenant à la fois des *motifs* et des "*non motifs*".
4. **Motifs extraits –** On recherche des *motifs multidimensionnels* afin d'éviter une sur-simplification du système observé, et la non détection de certaines évolutions critiques. Les instances d'un motif ont les caractéristiques suivantes :
   – **Variabilité dans les valeurs**, due à celle des comportements humains.
   – **Présence d'interruptions** dans la réalisation d'une activité (toilettes, etc.).
   – **Déformations et translation dans le temps**, car une même activité se répète à des instants et sur des durées variables.

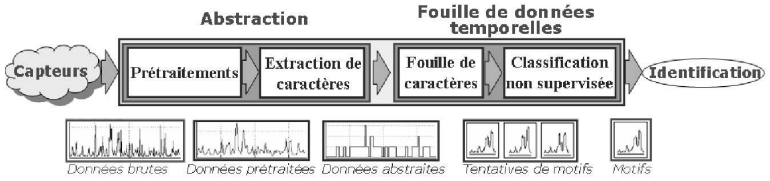


FIG. 1 – Méthode d'identification non supervisée de motifs temporels. Les signaux représentés illustrent sur une dimension le type des données disponibles après chaque étape.

Une originalité de ce travail est la considération de séquences multidimensionnelles et hétérogènes, dans l'objectif d'extraction de motifs multidimensionnels. Il est en particulier nécessaire de définir une mesure de similarité adaptée à la comparaison de séquences de ce type. La mesure proposée est basée sur la plus longue sous-séquence commune – *Longest Common Subsequence (LCSS)* (Duchêne *et al.*, 2004).

La première étape de la méthode proposée (voir figure 1) consiste en l'**abstraction** des données brutes issues des capteurs, pour leur donner un sens au regard de l'objectif de l'analyse. Il s'agit de mettre en évidence les tendances à plus ou moins long terme, en résumant les situations "stationnaires" observées. Une séquence est ainsi représentée par une succession de symboles estampillés. La **fouille de données** comprend d'abord une étape de **fouille de caractères**, basée sur la méthode des projections aléatoires (Buhler & Tompa, 2002; Chiu *et al.*, 2003), pour l'identification des sous-séquences récurrentes. Les sous-séquences significatives sont sélectionnées selon deux critères maximum de distance et minimum de collisions. Une méthode de synthèse basée sur une classification divisive des sous-séquences récurrentes permet ensuite de générer un ensemble de sous-séquences disjointes – les *tentatives de motifs*. Leur **classification** ascendante hiérarchique en *motifs* est enfin réalisée sur la base d'une mesure de distance.

L'approche proposée est expérimentée dans le cadre de la télésurveillance médicale à domicile, à partir des déplacements, postures, niveau d'activité, fréquence cardiaque moyenne d'une personne. De manière générale, on constate de bonnes performances des étapes d'extraction et de classification des motifs. À partir de séquences simulées pour une personne dans des conditions habituelles de vie, on vérifie qu'il est possible d'extraire des sous-séquences représentatives de comportements qu'on sait interpréter *a posteriori* en terme de la réalisation de certaines activités de la vie quotidienne.

Références

BUHLER J. & TOMPA M. (2002). Finding motifs using random projections. *Journal of Computational Biology*, **9**(2), 225–242.

CHIU B., KEOGH E. & LONARDI S. (2003). Probabilistic discovery of time series motifs. In *Proceedings of the 9^{th} ACM International Conference on Knowledge Discovery and Data Mining (KDD'03), Washington DC., USA*, p. 493–498.

DUCHÊNE F., GARBAY C. & RIALLE V. (2004). Similarity measure for heterogeneous multivariate time-series. In *Proceedings of the 12^{th} European Signal Processing Conference (EUSIPCO), Vienna, Austria.*

Taylor-based pseudo-metrics for random process fitting in dynamic programming : expected loss minimization and risk management

Sylvain Gelly[1], Jérémie Mary[1], Olivier Teytaud[12]

[1] TAO-inria
TAO, LRI, UMR 8623, CNRS - Univ. Paris-Sud, bat 490, F-91405 Orsay

[2] Artelys
215 rue J.J. Rousseau, F-92136 Issy-Les-Moulineux

Abstract : L'optimisation stochastique est la recherche de x optimisant l'espérance $EC(x, A)$, où A est une variable aléatoire. Typiquement $C(x, a)$ est le coût associé à une stratégie x qui fait face à une réalisation a du processus aléatoire. De nombreux problèmes d'optimisation stochastique traitent de plusieurs pas de temps, et conduisent à des temps de calcul importants ; des solutions efficaces existent, par exemple via le principe de décomposition de Bellman, mais seulement si le processus stochastique est représenté de manière bien structuré, typiquement par un modèle de Markov ou un arbre de scénarios. Le problème est que dans le cas général, A est loin d'être markovien ou bien structuré. Aussi, on cherche A', "ressemblant à A", mais appartenant à une famille donnée $\mathcal{A}'$ qui ne contient pas A. Le problème est alors la quantification du fait que A' ressemble à A, i.e. la définition d'une mesure de similarité pertinente entre A et A'.

Une solution classique est l'utilisation de la distance de Kantorovitch-Rubinstein (Gröwe-Kuska et al, 2003), justifiée par des bornes sur la différence $|EC(x, A) - EC(x, A')|$ via la distance de Kantorovitch-Rubinstein et des conditions de Lipschitz. Nous proposons d'autres (pseudo-)distances, basées sur des inégalités affinées, garantissant un bon choix de A'. De plus, comme dans beaucoup de cas on préfère en fait l'optimisation avec gestion du risque, i.e. l'optimisation de $E\ C(x, bruit(A))$ où $bruit(.)$ est un bruit alèatoire modélisant notre ignorance sur la variable aléatoire réelle, nous proposons une distance telle que A' optimisant la distance entre A' et A conduise à de bons coûts, en moyenne, pour $bruit(A)$. Des tests sont menés sur des données artificielles aux fonctions de coût réalistes pour montrer la pertinence de la méthode.

L'article complet est disponible à `http://www.lri.fr/~teytaud/alea_english.pdf`.

The stochastic optimization considered below typically consists in : i) choose A'

compatible with your optimizer close to A for your preferred distance; ii) optimize $C(s, A')$; iii) verify that $C(s, A)$ is not too large.

Instead of the Kantorovitch-Rubinstein distance between a random variable A and a random variable $\pi(A)$, we define

$$distance2 = E_A(|\nabla_A C(s_0, A)(\pi(A) - A)| + \frac{1}{2}(\pi(A) - A)^t H_A C(s_0, A)(\pi(A) - A))$$

which is dependent upon s_0, that we assume to be a strategy not too far from the optimal.

We show under mild hypotheses that choosing π minimizing this distance is better than minimizing the Kantorovitch-Rubinstein distance. In particular, figure 1 shows a much smaller resulting cost for the 50 random processes A' which are the closest to A for our distance (among 200 random processes), than for the 50 random processes A' which are the closest to A for the Kantorovitch-Rubinstein distance. This shows that $E\ C(\arg\min E\ C(., A'), A)$ is better with $A' = \arg\min\ distance2(A', A)$ than with $A' = \arg\min\ dk(A', A)$ where dk is the Kantorovitch-Rubinstein distance.

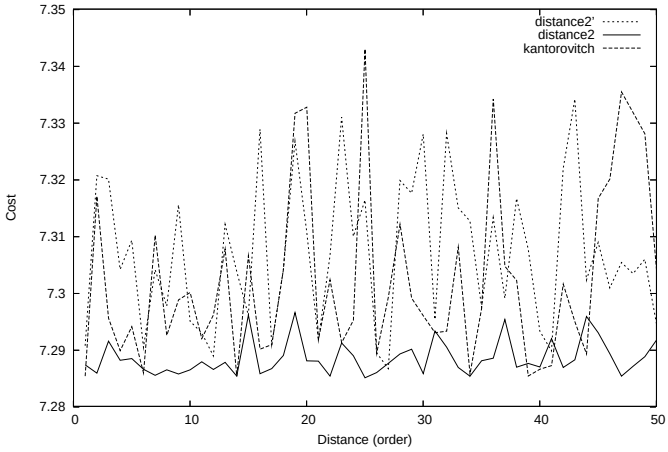


Figure 1: Results in **dimension 2**. We see that the Kantorovitch-Rubinstein distance is not correlated to the cost, whereas the $distance^{(2)}$ leads to good results. Hence, choosing a random process A' "close" to A in the Kantorovitch-Rubinstein distance leads to have a very bad estimation of the best cost, and leads to get an apparently-optimal solution (when looking at the cost for A') very far from the real optimum.

An adaptation to the case with risk has also been performed.

Acknowledgements

This work was supported in part by the Pascal Network of Excellence.

References

N. Gröwe-Kuska, H. Heitsch, W. Römisch, Scenario Reduction and Scenario Tree Construction for Power Management Problems, IEEE Bologna Power Tech Proceedings (A. Borghetti, C.A. Nucci, M. Paolone eds.), 2003.

Acquisition de contraintes ouvertes par apprentissage de solveurs

Andreï Legtchenko, Arnaud Lallouet

Université d'Orléans — LIFO
Laboratoire d'Informatique Fondamentale d'Orléans
BP 6759 — F-45067 Orléans cedex 2
prenom.nom@lifo.univ-orleans.fr

Résumé : Nous présentons une technique d'apprentissage d'un objet très particulier qui est un solveur de contraintes. Étant donné un sous-ensemble de solutions et un autre sous-ensemble de non-solutions pour une contrainte, nous cherchons une représentation de la contrainte entière sous la forme d'un solveur. Un solveur permet de distinguer les solutions des non-solutions, et ainsi il effectue la tache de classification. Mais il permet également réduire le domaine des variables lors du processus de résolution. Les résultats de tests démontrent la qualité de classification obtenue par l'apprentissage de solveurs, mais aussi l'efficacité de la réduction de domaines.

Mots-clés : Contraintes ouvertes ; Apprentissage de solveurs ; Algorithmes de classification.

1 Introduction

La programmation par contraintes est un outil très puissant de modélisation et de résolution de problèmes. Dans ce paradigme, un problème est représenté par un certain nombre de variables, avec leurs domaines (les valeurs possibles) et un ensemble de contraintes sur ces variables. Un exemple classique bien connu de problème avec contraintes (Constraint Satisfaction Problem, CSP) est un système d'équations mathématiques, par exemple $X^3 + \cos(Y + Z) = Z^2 \wedge X + Y + Z = 0$. Une solution du problème est une affectation de toutes les variables qui satisfait toutes les contraintes. Dans le domaine de la Programmation par Contraintes, il existe deux principales difficultés. La première concerne la modélisation par contraintes du problème à résoudre. L'utilisateur doit composer le CSP en utilisant une bibliothèque de contraintes disponibles (par exemple, des contraintes arithmétiques). Cette tâche peut s'avérer assez complexe, car la bibliothèque des contraintes disponibles peut être insuffisante pour un problème particulier. La deuxième difficulté consiste à résoudre efficacement le CSP formé, c'est-à-dire à trouver une (les) affectation(s) de variables satisfaisant à toutes les contraintes. Évidemment, la résolution d'un problème avec contraintes peut en théorie être faite par un algorithme général de recherche, qui parcourrait l'arbre des affecta-

tions possibles. Mais en pratique cette méthode est inutilisable vu la taille de l'espace de recherche. Ainsi, plusieurs techniques plus ou moins générales ont été développées afin de rendre la recherche de solutions la plus rapide possible.

Notre travail étend les possibilités de modélisation par contraintes tout en permettant une résolution efficace. L'idée de départ était de proposer une méthode qui permet à l'utilisateur de définir ses propres contraintes en donnant des exemples de solutions et de non-solutions. En effet, dans plusieurs problèmes réels interviennent les contraintes spécifiques dont l'expression formelle n'est pas connue. Ces contraintes expriment des préférences (”être une bonne configuration”), des concepts (”être un arbre”) ou encore les habitudes. On dispose généralement d'un certain nombre d'observations étiquetées (”solution” ou ”non-solution”) et on doit produire une représentation de toute la contrainte (qui sépare l'espace *total* des n-uplets en deux parties). On appelle *une contrainte ouverte* l'ensemble des exemples connus. On ”devine” la contrainte à partir des exemples. On appelle ce processus l'acquisition ou la fermeture de contrainte ouverte. Mais il ne suffit pas d'avoir une représentation quelconque de la contrainte, même si elle est juste (au sens qu'elle représente bien le concept). Une fois la contrainte obtenue, elle sera mise dans un CSP à résoudre, avec un certain nombre d'autres contraintes. Selon la technique de résolution utilisée, la représentation des contraintes joue plus ou moins sur la rapidité du processus de résolution. Ainsi on cherche une représentation qui :

- corresponde bien à la contrainte, c'est-à-dire permet de distinguer avec le minimum d'erreurs les solutions des non-solutions,
- facilite le processus de résolution, dans le cadre de la technique de résolution choisie à l'avance.

En clair, il ne suffit pas de classer bien les n-uplets, il faut aussi participer de façon la plus efficace possible à la résolution. Cette dernière condition fait de l'acquisition d'une contrainte à partir des exemples un problème plus complexe que l'apprentissage de classificateurs. Un arbre de décision peut classer correctement les n-uplets (en solutions ou en non-solutions), mais comment intégrer efficacement cet arbre dans la résolution de CSP, par exemple, par la réduction de domaines ? A notre connaissance, il n'y a pas de réponse à cette question. Au lieu d'essayer d'adapter un classificateur connu à la résolution efficace, nous proposons le schéma inverse : partir d'une représentation particulièrement intéressante pour la résolution et arriver à proposer une technique d'apprentissage pour cette représentation. La suite de l'article se compose comme suit. Dans la section 2, on donne un certain nombre de définitions importantes (contrainte, consistance, résolution par réduction de domaines, opérateurs de réduction, etc). On donne également une représentation générale pour les contraintes sous la forme d'un ensemble d'opérateurs de réduction de domaines (qui jouent un rôle actif pendant la résolution). Dans la section 3, on présente la méthode d'apprentissage des opérateurs de réduction à partir des exemples. Cette méthode permet d'acquérir la contrainte à partir des exemples sous la forme d'un ensemble de classificateurs particuliers. On compare la qualité de classification de notre système avec l'algorithme C5.0 et C5.0 avec boosting sur quelques jeux de données réels. Dans la section 4 nous montrons comment la représentation de la section précédente est transformée en un ensemble d'opérateurs de réduction pour la contrainte apprise. Nous présentons aussi les performances des

opérateurs de réduction obtenus pour quelques cas réels. Enfin, l'article se termine par une discussion et une conclusion.

2 Préliminaires : Contraintes et Résolution

Dans cette partie on donne les définitions relatives aux contraintes et à la résolution des CSP utiles à la compréhension du cadre général. Soit V un ensemble de variables et $D = (D_X)_{X \in V}$ leurs domaines finis. On note Π le produit cartésien des ensembles. Pour un sous-ensemble $W \subseteq V$, on note D^W l'ensemble de n-uplets sur W, c'est-à-dire l'ensemble $\Pi_{X \in W} D_X$. La projection d'un n-uplet ou d'un ensemble de n-uplets sur une variable ou un ensemble de variables est notée $|$, la jointure de deux ensembles de n-uplets est notée $\bowtie$. Si A est un ensemble, alors $\mathcal{P}(A)$ désigne l'ensemble des parties de A et $|A|$ son cardinal.

Définition 1 (Contrainte)

Une contrainte c *est un couple* (W, T) *où :*
- $W \subseteq V$ *est* l'arité[1] *de la contrainte* c *;*
- $T \subseteq D^W$ *est l'ensemble de* solutions *de* c*.*

La *jointure* de deux contraintes est définie comme une extension naturelle de la jointure des n-uplets : si $c = (W, T)$ et $c' = (W', T')$, alors $c \bowtie c' = (W \cup W', T \bowtie T')$. La table $T \bowtie T'$ contient tous les n-uplets $t \in D^{W \cup W'}$ tels que $t_{|W} \in T$ (la projection sur les variables de W est un n-uplet de T) et $t_{|W'} \in T'$ (la projection sur les variables de W' est un n-uplet de T').

Un CSP est un ensemble fini de contraintes. La jointure est naturellement étendue aux CSPs et les solutions d'un CSP $C = \{c_1, .., c_n\}$ sont $c_1 \bowtie ... \bowtie c_n$. Le calcul direct de cette jointure est trop complexe pour être faisable en pratique. D'où l'intérêt particulier des cadres d'approximation. Le cadre qui a le plus de succès est celui de réduction de domaines. Ainsi pour un ensemble $W \subseteq V$ un état de recherche est un ensemble des valeurs encore possibles pour chaque variable : $S_W = (s_X)_{X \in W}$ tel que $s_X \subseteq D_X$. L'espace de recherche est $S_W = \Pi_{X \in W} \mathcal{P}(D_X)$. L'ensemble S_W ordonné par l'inclusion point à point est un treillis complet. De même, l'inclusion et l'intersection des états de recherche sont définies aussi point à point. L'espace de recherche entier S_V est noté simplement S. Des états de recherche particuliers, appelés *singletoniques* jouent un rôle important dans notre cadre. Un état de recherche singletonique ne contient qu'une seule valeur pour chaque variable, et ainsi représente un n-uplet. Un n-uplet est transformé en un état singletonique par l'opérateur $\lceil\ \rceil$: pour $t \in D^W$, on a $\lceil t \rceil = (\{t_X\})_{X \in W} \in S_W$. Inversement, un état de recherche singletonique est transformé en un ensemble de n-uplets en prenant le produit cartésien Π : pour $s \in S_W$, $\Pi s = \Pi_{X \in W} s_X \subseteq D^W$. On note $Sing_W$ l'ensemble $\lceil D^W \rceil$ des états de recherche singletoniques. Par définition, $\lceil D^W \rceil \subseteq S_W$.

[1] Dans notre formalisme ensembliste, arité a un sens plus précis que le simple nombre de variables et contient l'information de typage.

La résolution du CSP est faite par un enchaînement d'étapes de propagation et d'étapes de recherche (figure 1). Une étape de propagation consiste à faire une itération chaotique des opérateurs de réduction de domaines jusqu'à un point fixe (Apt, 1999). Un opérateur de réduction est une fonction de $f : S_W \rightarrow S_W$ croissante (pour l'inclusion) et contractante ($\forall s \in S_W$, $f(s) \subseteq s$). Malheureusement, la réduction de domaines ne permet pas toujours de trouver directement une affectation de variables (les domaines ne deviennent pas toujours singletoniques). Pour remédier à ce problème, cette étape de propagation (qui n'a pas abouti à une affectation) est suivie par une étape de recherche (appelée labeling) qui, pour une variable du CSP, choisit et affecte une valeur parmi les valeurs encore possibles pour cette variable. Ensuite on recommence la réduction de domaines par propagation et ainsi de suite jusqu'à trouver une solution ou jusqu'à arriver à un domaine vide (absence de solutions). Si un choix pendant une étape de recherche conduit à l'échec (absence de solutions), une autre valeur pour la même variable sera essayée.

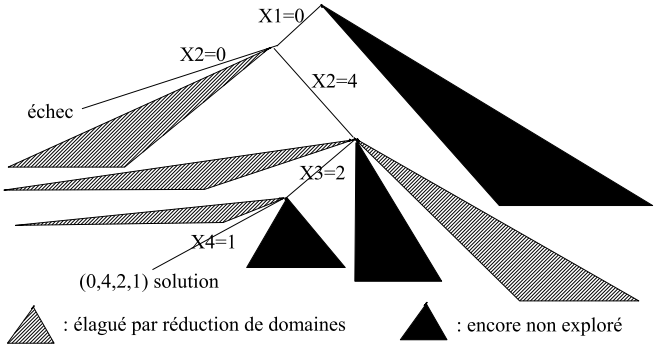


FIG. 1 – Résolution d'un CSP

Ainsi, la résolution d'un CSP est le parcours de l'arbre des affectations possibles, mais l'arbre est dynamiquement élagué par la réduction de domaines. Notons que ce schéma de résolution n'est pas le seul dans le domaine de la résolution des CSP (Tsang, 1993). Néanmoins, la résolution par itération des opérateurs de réduction de domaines et labeling a déjà fait ses preuves et elle est très largement répandue.

Les opérateurs de réduction sont généralement associés aux contraintes elle-mêmes. Chaque contrainte de la bibliothèque possède son propre opérateur de réduction. Cet opérateur est une représentation "active" de la contrainte. On appelle un tel opérateur une *consistance* :

Définition 2 (Consistance)
Un opérateur $f : S_W \rightarrow S_W$ *est une* consistance *pour une contrainte* $c = (W, T)$ *si :*
- f *est croissante :* $\forall s, s' \in S_W, s \subseteq s' \Rightarrow f(s) \subseteq f(s')$.
- f *est contractante :* $\forall s \in S_W, f(s) \subseteq s$.
- f *est correcte :* $\forall s \in S_W, \Pi s \cap T \subseteq \Pi f(s) \cap T$.
- f *est* associée *à la contrainte* c *:* $Fix(f) \cap Sing_W = \lceil T \rceil$ *où* $Fix(f)$ *est l'ensemble des points fixes de* f.

La correction veut dire qu'une consistance ne réduit jamais les domaines de telle sorte qu'une solution de la contrainte serait rejetée. La dernière propriété de la définition est très importante : une consistance est une représentation fonctionnelle de la contrainte, car les seuls n-uplets acceptés par l'opérateur sont exactement les solutions de la contrainte. *La représentation de la contrainte par son opérateur de consistance est une représentation idéale dans le cadre de résolution de CSPs par la réduction de domaines et la recherche*. On appelle *solveur* l'opérateur de consistance pour une contrainte, car la combinaison de l'opérateur avec un algorithme de parcours d'arbre donne un solveur pour la contrainte. L'*arc-consistance* est la consistance probablement la plus connue et l'une des plus utilisées. Bien qu'elle soit définie comme une propriété, il est possible de la calculer avec un opérateur. L'arc-consistance est définie par $\forall s \in S_W$, $ac_c(s) = s'$ avec $\forall X \in W$, $s'_X = (\Pi s \cap T)_{|X}$. En clair, on supprime des domaines les valeurs avec lesquelles on ne peut pas former de solutions de c. Pour cela, on réduit chaque domaine avec les projections de la contrainte sur la variable en question. L'arc consistance est la consistance la plus contractante qui peut exister pour une contrainte c indépendamment du contexte, c'est-à-dire sans considérer plusieurs contraintes à la fois (d'après la définition de ac). Supposons maintenant que chaque domaine D_X est équipé d'un ordre total $\leq$. On appelle l'intervalle $[a..b]$ l'ensemble $\{e \in D_X | a \leq e \leq b\}$. On appelle Int_X le treillis d'intervalles construit sur D_X. On note aussi $Int_W = \prod_{X \in W} Int_X$. Pour tout $A \subseteq D_X$ on note $[A]$ l'intervalle $[min(A)..max(A)]$. Alors on définit la *consistance de bornes* par : $\forall s \in S_W$, $bc_c(s) = s'$ avec $\forall X \in W$, $s'_X = s_X \cap [(\Pi s \cap T)_{|X}]$. La consistance de bornes supprime les valeurs en faisant bouger les bornes des intervalles. En général, c'est une consistance plus faible que l'arc-consistance (moins contractante) mais au même temps plus rapide à calculer. La rapidité de calcul et la possibilité de traitement de très grands domaines (pour approximer le continu) rend la consistance de bornes largement utilisée elle-aussi. On peut ordonner les consistances selon leur pouvoir de réduction : $f_1 \subseteq f_2 \Leftrightarrow \forall s \in S_W$, $f_1(s) \subseteq f_2(s)$. Alors on a $ac \subseteq bc$.

Soit $c = (W, T)$ une contrainte. Soit un opérateur de consistance op pour c. Soit $s \in S_W$ un état de calcul (les valeurs encore possibles pour les variables de W). L'application de l'opérateur op à s a pour l'effet la suppression (éventuelle) de certaines valeurs dans s. Il est possible de modéliser l'action de op sur s par le test de la présence des valeurs dans les domaines des variables. Soit une variable $X \in W$ et une valeur $a \in D_X$. On appelle *Fonction de Réduction Élémentaire* (FRE) la fonction $f_{X=a} : S_{W-\{X\}} \rightarrow \mathbb{B}$ qui renvoie **true** si la valeur a doit rester dans le domaine D_X, et **false** si elle doit être supprimée. L'action de l'opérateur op sur un état de calcul $s \in S_W$ est obtenu par la suppression dans tous les domaines courants s_X des valeurs a pour lesquelles leur fonction respective $f_{X=a}$ répond **false**. L'avantage de représenter un opérateur de consistance avec les FREs réside dans la simplicité des objets en présence. Un opérateur op est une fonction qui prend en paramètre un produit cartésien d'ensembles pour retourner un autre objet de même type. Une Fonction de Réduction Élémentaire est une fonction qui, certes, prend en paramètre toujours un produit cartésien d'ensembles, mais retourne un booléen. Nous avons donc $\sum_{X \in W} |D_X|$ fonctions et chacune est plus simple qu'un opérateur de réduction. Le type d'une FRE rend sa construction automatique plus simple que la construction automatique d'un opérateur de réduction.

Dans la section suivante, on introduit le concept de contrainte ouverte, ainsi qu'une méthode d'acquisition inspirée par la forme et le sens des fonctions de réduction élémentaires.

3 Contraintes ouvertes : acquisition

La modélisation par contraintes des problèmes réels peut être considérablement enrichie par la possibilité donnée à l'utilisateur de définir ses propres contraintes. Nous nous intéressons au cas où l'utilisateur fournit un certain nombre d'exemples de solutions et de non-solutions pour la contrainte, et la totalité de la contrainte est obtenue par apprentissage. On appelle la *contrainte ouverte* la partie connue de la contrainte :

Définition 3 (Contrainte Ouverte)
Une contrainte ouverte *est un triplet* (W, T^+, T^-) *où*
- $T^+ \subseteq D^W$ *et* $T^- \subseteq D^W$
- $T^+ \neq \emptyset$ *et* $T^- \neq \emptyset$
- $T^+ \cap T^- = \emptyset$

Une contrainte ouverte est difficilement utilisable dans un CSP classique, car on est incapable de faire le test de satisfiabilité (décider si le n-uplet est une solution ou pas) pour les n-uplets non étiquetés. Nous devons d'abord "fermer" la contrainte ouverte, c'est-à-dire pouvoir étiqueter l'ensemble de n-uplets de D^W et ainsi définir une contrainte "classique". Nous appelons cette phase de "fermeture" la phase d'acquisition de la contrainte. Quels sont les cas pratiques où on a besoin d'acquérir une contrainte à partir d'une contrainte ouverte ? Nous pouvons aujourd'hui suggérer deux grandes familles :

- Les contraintes "être un bon plan" ou "être n états consécutifs d'un système dynamique", où chaque variable X_i représente l'action à effectuer au pas numéro i. Les applications de ce type de contraintes sont la robotique (Lallouet *et al.*, 2004), programmation de processus industriels, les traitements médicaux, les plans d'investissement, un agenda coopératif qui essaie de modéliser les emplois de temps des autres personnes.
- Les contraintes de type "être un bon objet", "être une configuration préférée" ou "être un exemple du concept", où chaque variable représente un degré de liberté de configuration et les valeurs des variables représentent les choix possibles (ex : variable "processeur" et les types de processeurs possibles comme valeurs ; variable "couleur de sièges" et les couleurs disponibles comme valeurs).

Voici un exemple d'utilisation de contrainte ouverte :

Exemple 4 (Salades)
On suppose qu'une société possède une cantine qui sert un grand nombre de repas par jour. Dans chaque plateau servi, il y a une salade. Le problème que le chef doit traiter, c'est quelle recette choisir et combien de portions préparer de chaque recette. Les contraintes sont : le nombre de repas à servir, la quantité des ingrédients disponibles, le coût des salades à minimiser etc. Les variables correspondent aux différents ingrédients,

et les valeurs - les quantités à utiliser. Le chef dispose d'une base de recettes qui, dans cette modélisation, correspond à la contrainte "être une bonne salade". Nous proposons, en ajoutant un certain nombre de "mauvaises recettes", considérer l'ensemble de n-uplets obtenus comme une contrainte ouverte. La fermeture de cette contrainte ouverte permettra d'étendre la base de recettes, mais il est souhaitable que la performance de résolution ne soit pas compromise. Voici la description détaillée de cette contrainte ouverte "salades" (que nous avons constituée avec un expert en la matière) : il y a 22 ingrédients, comme les tomates, la mayonnaise, le thon en conserve, l'huile d'olive ou encore la laitue. Le domaine de chaque variable est entre 2 ou 4 valeurs. Il y a 64 valeurs en tout ($\sum_{i=1}^{22} |D_i|$). L'ensemble des solutions connues comporte 53 recettes ; l'ensemble des non-solutions connues (qui sont des recettes pas bonnes ou qui ne sont pas du tout des recettes) comporte 281 exemples.

Nous proposons une méthode d'acquisition qui est basée sur les puissantes techniques d'apprentissage connues pour obtenir un bon résultat. En même temps, la représentation obtenue est facilement et automatiquement transformable en un opérateur de consistance pour la contrainte fermée (c'est-à-dire celle obtenue par apprentissage à partir de la contrainte ouverte).

Soit (W, T^+, T^-) une contrainte ouverte. Nous allons apprendre $\sum_{X \in W} |D_X|$ classificateurs à deux classes spécifiques, c'est-à-dire autant qu'il y a de FREs pour représenter l'opérateur de consistance pour une contrainte sur les variables de W. Soit $X \in W$ et $a \in D_X$. Le classificateur $n_{X=a} : \prod_{W-\{X\}} D_X \rightarrow \mathbb{B}$ prend en paramètres les variables *instanciées* de W, autres que X et retourne **true** si la valeur a est possible pour la variable X, et **false** si la valeur a est impossible. En fait, un classificateur $n_{X=a}$ est la version singletonique (qui fonctionne seulement sur les variables instanciées) de la fonction de réduction élémentaire $f_{X=a}$. C'est la particularité de notre méthode, que d'utiliser les classificateurs inspirés par les fonctions de réduction élémentaires pour apprendre un concept. Dans la section suivante on verra que la transformation du classificateur $n_{X=a}$ en FRE $f_{X=a}$ est formelle et ne fait pas d'appel aux techniques d'apprentissage. L'ensemble d'exemples et de contre-exemples pour le classificateur $n_{X=a}$ est obtenu par la sélection des n-uplets t tels que $t_X = a$ suivie de la projection sur $W - \{X\}$. Les n-uplets de T^+ (les exemples de solutions) dont la projection sur X est a nous donnent les exemples pour $n_{X=a}$:

$$Ex^+_{X=a} = \{t_{|W-\{X\}} | t \in T^+,\ t_{|X} = a\}.$$

De même, les n-uplets de T^- dont la projection sur X est a donnent les contre-exemples pour $n_{X=a}$:

$$Ex^-_{X=a} = \{t_{|W-\{X\}} | t \in T^-,\ t_{|X} = a\}.$$

Nous avons réalisé le système `Solar` qui effectue l'apprentissage de classificateurs $n_{X=a}$ pour une contrainte ouverte donnée par l'utilisateur. Pour représenter les classificateurs nous avons choisi les perceptrons avec une couche cachée (voir (Mitchell, 1997) pour plus de références). Le choix a été suggéré, d'une part, par la puissance de cette technique, et par la facilité de transformation en FRE d'un perceptron multi-couches, d'autre part. L'architecture du réseau utilisé est tout à fait classique (figure 2). La fonction d'activation est la fonction sigmoïde. Chaque neurone de la couche cachée est

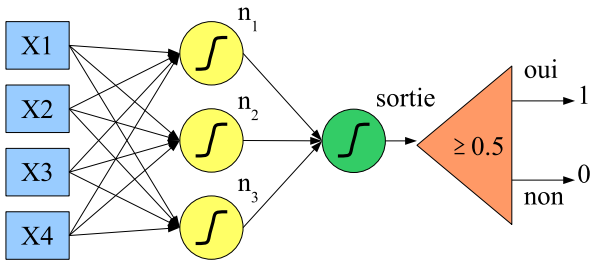


FIG. 2 – Architecture du réseau de neurones utilisé

connecté à chacun des neurones d'entrée. De même, le neurone de sortie est connecté à tous les neurones de la couche cachée. La sortie des neurones est dans l'intervalle $]0, 1[$. En supposant les domaines de variables totalement ordonnés, on fait correspondre à chaque valeur un entier naturel : par exemple, à chaque $a_i \in D_X$ on associe i. On effectue ensuite une conversion vers l'intervalle $[0, 1]$. Le neurone d'entrée correspondant à X reçoit la quantité $i/|D_X|$ quand la variable X reçoit a_i. Après le neurone de sortie, il y a une unité de décision : si la valeur de sortie de ce neurone est supérieure ou égal à 0.5, la réponse du classificateur est **true**, et **false** sinon. Le nombre de neurones dans la couche cachée est le même pour tous les classificateurs, pour une contrainte ouverte donnée. Ceci n'est pas obligatoire et d'ailleurs ce nombre est un paramètre de la méthode. Nous l'avons fixé pour nos exemples entre 3 et 5. L'apprentissage des poids est fait par l'algorithme de rétropropagation du gradient. L'apprentissage de chaque classificateur est arrêté quand l'ensemble des exemples est bien classé, ou après un délai raisonnable.

Une fois les classificateurs appris, on les combine pour obtenir un classificateur global de n-uplets qui effectue le test de satisfiabilité (détermine si un n-uplet est une solution ou non). Soit un n-uplet $t \in D^W$. Soit $X \in W$. Nous supposons que $t_{|X} = a$. Nous allons exécuter le classificateur $n_{X=a}$ sur $t_{|W-\{X\}}$. Si la valeur retournée est **false**, cela veut dire que selon le classificateur $n_{X=a}$ la valeur $t_{|X} = a$ est impossible vu les affectations des autres variables. Nous allons exécuter tous les classificateurs $n_{X=a}$ tels que $X \in W$ et $t_{|X} = a$. Le classificateur associé à une valeur du n-uplet t vérifie la légitimité de la présence de cette valeur dans le n-uplet. Il y a en tout $|W|$ classificateurs qui sont sollicités pour tester un n-uplet. Les $|W|$ réponses sont combinées pour donner un avis sur le n-uplet selon deux méthodes :

- la méthode "vote majoritaire" : si le nombre de valeurs légitimes dépasse la moitié des votants, le n-uplet est déclaré comme une solution, sinon c'est une non-solution ;
- la méthode "vote avec veto" : un n-uplet est considéré comme une solution seulement si toutes les valeurs sont légitimes, et comme une non-solution si au moins une valeur n'est pas légitime.

La deuxième méthode est issue du monde des contraintes, où un état de calcul $s \in S_W$ est rejeté si au moins un domaine de variables est vide. Comme les classificateurs $n_{X=a}$ sont les versions singletoniques des fonctions de réduction élémentaires, la méthode de

vote avec veto est seulement la traduction du comportement des FREs sur un état de calcul singletonique, quand il y a une seule valeur dans chaque domaine).

Nous avons testé la qualité de notre méthode d'acquisition comme une méthode de classification à 2 classes. Nous avons pris 4 jeux de données différents : la base "recettes de salades" de l'exemple 4 et trois bases trouvées dans *UCI Machine Learning Repository*[2]. Pour avoir une idée de la performance relative de notre technique nous avons testé sur les mêmes bases l'algorithme d'apprentissage d'arbres de décision C5.0 (qui est une version améliorée du très populaire C4.5 (Quinlan, 1993)) et C5.0 avec boosting (RuleQuest Research, 2004) avec $|W|$ votants pour contrebalancer le nombre de nos classificateurs. Notons au passage que notre méthode de construction de classificateurs est différente de celle du boosting (Freund & Shapire, 1999) par la façon de construire les ensembles d'exemples pour les classificateurs. Nous avons utilisé la validation croisée classique avec 10 blocs. Le jeu de données est divisé de manière aléatoire en 10 blocs de même taille. Un bloc est utilisé pour la validation, et les autres pour l'apprentissage. Après avoir fait 10 tests, à chaque fois avec un bloc de validation différent, on calcule la moyenne. Cinq sessions différentes de validation croisée ont été effectuées pour augmenter la fiabilité de résultats (il y a donc 50 tests au total). Les résultats des tests ainsi que la description des bases sont regroupés dans le tableau 1. Dans ce tableau, "Solar veto" et "Solar maj" signifient les performances de notre système avec les différentes modes de vote. Le nombre optimal de neurones dans les couches cachées a été à chaque fois déterminé empiriquement. Le temps d'apprentissage est le temps nécessaire pour apprendre tous les classificateurs. Le temps d'apprentissage pour le système C5.0 avec ou sans boosting est moins d'une seconde. Mais le temps d'apprentissage n'a absolument aucune importance dans l'application aux contraintes, car le gain apporté par l'utilisation des propagateurs obtenus compense très largement les minutes nécessaires pour apprendre les classificateurs élémentaires. Ainsi, la comparaison de notre technique avec les algorithmes connus, comme les arbres de décision, sert seulement à signaler la qualité d'acquisition de concepts, sans se mettre en compétition avec les autres technique d'apprentissage, car les techniques d'apprentissage connues ne fournissent pas de solveur pour le concept appris.

Comme on le constate, la qualité de classification défie celle des algorithmes réputés comme très efficaces. On constate que la méthode de vote avec veto est visiblement moins performante que la méthode avec vote majoritaire. Mais il faut comprendre que lors de la classification par le vote avec veto, l'erreur est principalement faite (en proportion écrasante) sur la classe "solution". En fait, cette méthode accepte seulement les n-uplets qui sont des solutions "irréprochables". Ainsi cette méthode tend à "purifier" la classe de solutions, ce qui peut même être recherché par l'utilisateur (dans les cas où il vaut mieux avoir une solution sûre quitte à en avoir moins au total). Par exemple, dans le cas de la base "salades", on passe par apprentissage de 53 recettes à 7.4E4 recettes (le nombre de n-uplets acceptés unanimement). Probablement, la vraie classe "bonne salade" est encore plus grande, mais 7.4E4 recettes sont déjà largement suffisantes. En plus, les 7.4E4 recettes sont des recettes "sûres" car acceptées unanimement (un expert confirme qu'il y a moins de 3% de mauvaises recettes dans un échantillon de 100 recettes prélevées aux hasard dans les 7.4E4 solutions).

[2] `http ://www.ics.uci.edu/~mlearn`

Database	**salades**	**mushroom**	**br-cancer**	**votes-84**
Arité	22	22	9	16
Taille de domaines	2-4	2-12	10	3
#Classificateurs	64	116	90	48
#examples	334	8124	699	435
#neurones dans CC	3	3	5	5
Temps d'app.	$55''$	$2'30''$	$8'30''$	$4'30''$
Erreur Solar veto	11.9%	6.9%	4.6%	25.9%
Erreur Solar maj	**3.6**%	0.7%	**3.5**%	3.8%
Erreur C5.0	9.9%	0.8%	5.5%	**3.7**%
Erreur C5.0 boost	4.8%	**0.2%**	3.7%	4.4%

TAB. 1 – Tests de classification.

Nous pensons que notre technique d'acquisition de contraintes est intéressante, vue les résultats de tests, même dans son application immédiate à la classification, sans parler de la possibilité d'obtenir un solveur pour la contrainte apprise par la transformation des classificateurs. Nous allons voir dans la partie suivante la méthode de transformation et les propriétés du solveur obtenu.

4 Contraintes ouvertes : obtenir les opérateurs

La méthode que nous avons proposé pour l'acquisition de contraintes, telle qu'elle a été exposée, est une méthode de classification de n-uplets en deux classes. Même si les classificateurs $n_{X=a}$ sont inspirés par les fonctions de réduction élémentaires, ils ne peuvent pas être directement utilisés pour la réduction de domaines au cours de la résolution. Les classificateurs $n_{X=a}$ représentent la contrainte au sens qu'ils distinguent les solutions de non-solutions à partir des variables instanciées. Soit $X \in W$ une variable et $a \in D_X$ une valeur. Le classificateur $n_{X=a} : (\prod_{W-\{X\}} D_X) \rightarrow \mathbb{B}$ prend en paramètres les variables de $W - \{X\}$ instanciées pour renvoyer la décision. Comment à partir de ce classificateur définir une fonction de réduction élémentaire $f_{X=a}$ qui prendrait en paramètre des domaines, c'est-à-dire des variables non complètement instanciées ? Supposons que la contrainte est apprise sous la forme d'un ensemble de classificateurs $n_{X=a}$, pour tout $X \in W$ et tout $a \in D_X$. Soit $s \in S_W$ un état de calcul. La valeur a peut être supprimée du domaine D_X si et seulement si $\forall t \in s_{|W-\{X\}}$ on a $n_{X=a}(t) = \mathbf{false}$. Dit autrement, la valeur a est supprimée du s_X si et seulement si pour tout n-uplet t de l'état de calcul courant s tel que $t_{|X} = a$ le classificateur $n_{X=a}$ trouve la valeur $X = a$ impossible (on pourrait dire illégitime ou encore inconsistante). Voici la définition de la fonction de réduction élémentaire $f^1_{X=a}$ à partir du classificateur $n_{X=a}$, qui est une FRE possible construite à partir du $n_{X=a}$:

Définition 5 (FRE via classificateur (I))

Un classificateur $n_{X=a}$ *:*$\prod_{W-\{X\}} D_X \rightarrow \mathbb{B}$ *est transformé en fonction de réduction élémentaire* $f^1_{X=a}$ *:*$S_{W-\{X\}} \rightarrow \mathbb{B}$ *par :*

$$\forall s \in S_{W-\{X\}},\ f^1_{X=a}(s) = \bigvee_{t \in \Pi s} n_{X=a}(t).$$

Cette façon de définir les FREs n'a rien d'un hasard : l'opérateur modélisé par ces FREs est l'opérateur de consistance pour la contrainte définie par les classificateurs votant avec veto.

Proposition 6 (Consistance via classificateurs (I))
Soit une contrainte ouverte (W, T^+, T^-)*. On suppose que la contrainte a été fermée par l'acquisition sous la forme de* $\sum_{X \in W} |D_X|$ *classificateurs. On suppose que les classificateurs votent avec le droit de veto pour définir l'ensemble de solutions (seuls les n-uplets acceptés unanimement sont considérés comme solutions). On note* $c = (W, T)$ *la contrainte obtenue. Alors l'opérateur représenté par les FREs de la définition 5 est un opérateur de consistance pour la contrainte* $c = (W, T)$*.*

Preuve On rappelle l'utilisation des FREs pour représenter un opérateur de réduction : étant donné un état de calcul $s \in S_W$, toutes les FREs concernées (c'est-à-dire les $f^1_{X=a}$ telles que $a \in s_X$) sont exécutées, et dans les cas où $f^1_{X=a}(s_{|W-\{X\}}) = \mathbf{false}$ la valeur a est supprimée du domaine courant s_X. Les nouveaux domaines ainsi formés donnent le nouvel état de calcul s'. On appelle op^1 :$S_W \rightarrow S_W$ l'opérateur ainsi défini. On suppose que la contrainte ouverte (W, T^+, T^-) a été fermée (généralisée) pour donner la contrainte $c = (W, T)$ par l'apprentissage des classificateurs $n_{X=a}$, pour tout $X \in W$ et tout $a \in D_X$. On suppose que les classificateurs votent avec le droit de veto pour définir l'ensemble de solutions T. Montrons d'abord que op^1 est une consistance pour la contrainte $c = (W, T)$.

- **Croissance** : D'après la définition 5, les fonctions FREs sont croissantes. L'opérateur op^1 est donc lui aussi croissant.
- **Contractance** : L'opérateur op^1 est contractant car les FREs ne font que supprimer les valeurs (sans jamais en rajouter).
- **Correction** : Soit $s \in S_W$. On suppose que $\Pi s \cap T \neq \emptyset$. Soit $t \in \Pi s \cap T$. On a donc $t \in T$. Cela veut dire que tous les classificateurs sont d'accord sur ce n-uplet (tous retournent **true**). Alors par la définition 5 toutes les FREs retournent **true** sur $\lceil t \rceil$: l'état singletonique est stable par toutes les FREs. Comme de plus l'opérateur op^1 est croissant, pour tout $s \in S_W$ tel que $\lceil t \rceil \subseteq s$ on a $\lceil t \rceil \subseteq op^1(s)$. Par conséquent, $\forall s \in S_W, \Pi s \cap T \subseteq \Pi op^1(s) \cap T$: les solutions ne sont jamais rejetées par op^1.
- **Association à la contrainte** : Soit $s \in Fix(op) \cap Sing_W$ un état de calcul singletonique (une seule valeur dans chaque domaine) stable par l'opérateur op^1 ($op^1(s) = s$). D'après la définition de l'opérateur op^1, l'état s est stable parce que toutes les FREs l'acceptent. Mais d'après la définition 5 un état singletonique sera stable seulement si tous les classificateurs acceptent le n-uplet Πs. Or, c'est la définition d'un n-uplet solution. Donc $\Pi s \in T$. Soit $t \in T$ un n-uplet. Par correction, l'état $\lceil t \rceil$ est stable par op^1, et donc $\lceil t \rceil \subseteq Fix(op^1) \cap Sing_W$. Il en résulte que $\lceil t \rceil \subseteq Fix(op^1) = \lceil T \rceil$: l'opérateur op^1 représente exactement la contrainte c.

L'opérateur op^1 est donc une consistance pour la contrainte c. A cause du fait que l'opérateur de l'arc-consistance ac est une consistance pour c, et qu'en plus il est le plus contractant, on a $Fix(ac) \subseteq Fix(op^1) : ac \subseteq op^1$. $\square$

La définition 5 de la fonction de réduction élémentaire propose une méthode de calcul de cette fonction, à partir du classificateur correspondant. Malheureusement, cette méthode n'est pas utilisable en pratique car elle demande la génération et le parcours de tous les n-uplets de Πs, pour un $s \in S_{W-\{X\}}$ donné. Par exemple, 10 domaines de 10 valeurs chacun représentent 10^{10} n-uplets. Quand on sait que l'opérateur est exécuté plusieurs fois (même plusieurs milliers de fois) pendant la résolution, on comprend que cette façon de calculer est inacceptable en pratique.

Nous proposons d'utiliser les techniques issues de l'Analyse des Intervalles (Moore, 1966) pour résoudre ce problème. Et plus précisément, nous allons utiliser l'*Arithmétique des Intervalles* pour construire les extensions naturelles aux intervalles des classificateurs. Rappelons brièvement le principe de l'arithmétique des intervalles de réels. Au lieu de travailler avec les variables réelles, on fait du calcul avec les variables intervalles. Les objets manipulés sont les intervalles fermés de réels. On note $I_{\mathbb{R}} = \{[a,b] | a,b \in \mathbb{R},\ a \leq b\}$ l'ensemble des intervalles fermés de réels. Quelques opérations élémentaires sur les intervalles $[a,b]$, $[c,d] \in I_{\mathbb{R}}$ dont définies dans la table 2. Ce sont les extensions canoniques aux intervalles des opérations sur les nombres réels. Soit une fonction $f : \mathbb{R} \rightarrow \mathbb{R}$ dont l'expression syntaxique est un arbre composé exclusivement des opérations $+$, $-$, $*$ etc. Alors l'extension naturelle aux intervalles de f est la fonction $F : I_{\mathbb{R}} \rightarrow I_{\mathbb{R}}$ qui est obtenue à partir de l'expression de la fonction f en remplaçant chaque variable réelle par une variable intervalle et chaque opérateur par son extension aux intervalles (exemple 7).

$$
\begin{array}{lcl}
[a,b] + [c,d] & = & [a+c, b+d] \\
[a,b] - [c,d] & = & [a-c, b-d] \\
[a,b] * [c,d] & = & [\min(P)), \max(P)] \\
 & & \text{avec } P = \{ac, ad, bc, bd\} \\
\exp([a,b]) & = & [\exp(a), \exp(b)]
\end{array}
$$

TAB. 2 – Opérations sur les intervalles.

Exemple 7
Soit $f : \mathbb{R} \rightarrow \mathbb{R}$, $f(x) = x * (exp(x) + 1)$ a pour extension naturelle aux intervalles la fonction : $F : I_{\mathbb{R}} \rightarrow I_{\mathbb{R}}$, définie par $F(X) = X * (exp(X) + [1,1])$ où X est une variable intervalle.

L'extension naturelle aux intervalles est une fonction croissante ($A \subseteq B \Rightarrow F(A) \subseteq F(B)$). D'après le "Théorème Fondamental de l'Analyse des Intervalles" (Moore, 1966) on a la propriété suivante :

Proposition 8 (Correction de l'extension naturelle)
Soit une fonction $f : \mathbb{R} \rightarrow \mathbb{R}$ et $F : I_{\mathbb{R}} \rightarrow I_{\mathbb{R}}$ son extension naturelle aux intervalles.

Alors on a $\forall I \in I_{\mathbb{R}}$, $\forall x \in I$, $f(x) \in F(I)$.

Cette propriété traduit la correction au sens des contraintes : si x était dans l'intervalle I, alors l'image de x par f se retrouve dans l'image de l'intervalle I par F. Le même principe d'extension et la proposition sont valables pour les fonctions du type $\mathbb{R}^n \rightarrow \mathbb{R}$.

Soit un classificateur $n_{X=a}$. Comme nous l'avons présenté, c'est un perceptron à 3 couches avec une unité de décision à seuil à la sortie. Ce perceptron est une fonction de type $\mathbb{R}^{|W|-1} \rightarrow \mathbb{R}$ dont l'expression est formée à partir des opérations de base $+$, $*$, $-$, $/$ et $\exp$. Pour tout $X \in W$ et tout $a \in D_X$ on définit la fonction $N_{X=a}$: $\mathbb{R}^{|W|-1} \rightarrow \mathbb{R}$ comme étant l'extension naturelle aux intervalles du perceptron $n_{X=a}$. La fonction $N_{X=a}$ prend en paramètres les domaines des variables de $W-\{X\}$ sous la forme d'intervalles, et elle retourne l'intervalle image des domaines par le classificateur $n_{X=a}$. Comme l'unité de décision à la sortie du perceptron $n_{X=a}$ a un seuil de 0.5, nous utilisons le même seuil pour l'extension $N_{X=a}$: si la borne de l'intervalle de sortie est supérieure ou égal à 0.5, la valeur est gardé dans le domaine, sinon elle est supprimée. La fonction $N_{X=a}$ munie de l'unité de décision à la sortie (avec le seuil 0.5) est une FRE :

Définition 9 (FRE via classificateur (II))

Un classificateur perceptron $n_{X=a}$ *:*$\prod_{W-\{X\}} D_X \rightarrow \mathbb{B}$ *est transformé en fonction de réduction élémentaire* $f^2_{X=a}$ *:*$S_{W-\{X\}} \rightarrow \mathbb{B}$ *en composant l'extension* $N_{X=a}$ *avec une unité de décision :*

$$\forall s \in S_{W-\{X\}},\ f^2_{X=a}(s) = \begin{cases} \textbf{true},\ \max(N_{X=a}([s])) \geq 0.5 \\ \textbf{false},\ sinon. \end{cases}$$

avec $\forall s \in S_W$, $[s] = ([s_X])_{X \in W}$.

L'opérateur représenté par les FREs du type $f^2_{X=a}$ est un opérateur de consistance pour la contrainte apprise :

Proposition 10 (Consistance via classificateurs (II))
Soit une contrainte ouverte (W, T^+, T^-). *On suppose que la contrainte a été fermée par l'acquisition sous la forme de* $\sum_{X \in W} |D_X|$ *classificateurs. On suppose que les classificateurs votent avec le droit de veto pour définir l'ensemble de solutions (seuls les n-uplets acceptés unanimement sont considérés comme solutions). On note* $c = (W, T)$ *la contrainte obtenue. Alors l'opérateur représenté par les FREs de la définition 9 est un opérateur de consistance pour la contrainte* $c = (W, T)$. *On appelle* op^2 *cet opérateur.*

Preuve La même preuve que pour la proposition 6. La monotonie et la correction des $f^2_{X=a}$ résulte des propriétés de l'extension naturelle aux intervalles. □

Le calcul de cet opérateur est par contre très rapide, car l'exécution de chaque fonction de réduction élémentaire $f^2_{X=a}$ est du même ordre que l'évaluation d'un perceptron simple $n_{X=a}$. C'est un grand avantage pour la résolution des CSP par rapport au calcul lourd des fonctions $f^1_{X=a}$. Mais cette facilité de calcul est payante : la fonction $f^2_{X=a}$

réduit moins que la fonction $f^1_{X=a}$:$\forall s \in S_{W-\{X\}}$, $f^1_{X=a}(s) \leq f^2_{X=a}(s)$. La fonction obtenue par l'extension naturelle aux intervalles du classificateur $n_{X=a}$ répond par **false** moins souvent que la fonction définie directement. Cet inconvénient provient des propriétés de l'extension naturelle aux intervalles, qui ne donne pas toujours l'image exacte de l'intervalle argument, mais plutôt un intervalle plus grand incluant la vraie image. L'Analyse des Intervalles et le *Théorème des simples occurrences* (Moore, 1966) nous apprend que si dans l'expression de la fonction à étendre toutes les variables n'ont qu'une seule occurrence, alors l'extension calcule l'intervalle image exact.

Néanmoins, malgré une puissance de réduction moindre que l'arc-consistance (la consistance la plus forte pour une contrainte prise toute seule), nos tests montrent que l'opérateur de consistance obtenu est assez efficace. L'utilisation d'une consistance plus faible que l'arc-consistance est pleinement justifiée dans le cas des grandes contraintes du monde réel (comme les cas traités ici), car dans (Bessière *et al.*, 2004b) on montre que le calcul *exact* du support (c'est-à-dire le calcul d'une FRE la plus forte) est NP-complet.

Nous avons effectué une série de tests pour estimer la puissance de réduction de l'opérateur op^2. Pour cela nous considérons les bases de données "salades", "mushroom", "breast-cancer-wisconsin" et "house-votes-84" comme des contraintes ouvertes. Une des classes est considérée comme la classe solution. Le système `Solar` construit l'ensemble des classificateurs représentant la contrainte donnée, ensuite les transforme automatiquement en fonctions de réduction élémentaires en prenant l'extension aux intervalles. L'opérateur obtenu est ajouté dans notre propre solveur de contraintes. Le CSP à résoudre est simple : c'est la contrainte toute seule. On demande au solveur de générer toutes les solutions de la contrainte donnée. Le tableau 3 regroupe les résultats des tests. On y trouve : le nombre de solutions trouvées (c'est exactement le nombre de solution de la contrainte apprise car le solveur trouve toutes les solutions) ; le nombre d'échecs pendant la recherche de toutes les solutions (faire un échec, c'est d'arriver à un état où au moins un des domaines est vide) ; le nombre moyen d'échecs par solutions ; le temps de génération de toutes les solutions. Notons que l'arc-consistance ne ferait aucun échec pour une contrainte isolée (mais elle est indisponible pour une contrainte ouverte !).

Database	**salades**	**mushroom**	**br-cancer**	**votes-84**
#Solutions	7.4E5	$\geq$ 4.1E6	1.27E5	1.27E5
#Echecs	1.34E5	$\geq$ 3.1E6	1.28E5	3.47E5
Echecs/Solution	1.8	0.75	0.99	2.86
Temps	$5'15''$	$\geq$ 2h	$3'00''$	$7'30''$

TAB. 3 – Tests des solveurs obtenus.

Les tests démontrent la différence essentielle entre un classificateur de n-uplets simple et un opérateur de consistance. Pour générer les solutions avec un classificateur, le seul moyen est de parcourir l'espace des n-uplets et de les tester un par un avec le classificateur. Mais, par exemple, pour la contrainte "mushroom" nous avons pu récupérer avec l'opérateur de consistance op^2 plus de 4.1E6 solutions en 2h, alors que en parcou-

rant l'espace des n-uplets et en faisant le test de classification nous n'avons que 7.7E4 solutions en 3h. Dans le cas de résolution d'un vaste CSP, le gain apporté par l'utilisation de l'opérateur de consistance permet de conserver les performances de résolution, tout en élargissant les possibilités de modélisation de problèmes grâce aux contraintes ouvertes.

5 Discussion et conclusion

Les contraintes ouvertes ont été introduites dans (Faltings & Macho-Gonzalez, 2002) dans le contexte du raisonnement distribué, où le but est la minimisation du nombre de requêtes nécessaires pour compléter la définition de la contrainte. Les contraintes ouvertes sont utilisées également dans le cadre de *Interactive Constraint Solving* (Alberti *et al.*, to appear) travail qui ne concerne pas l'apprentissage. L'apprentissage des préférences *molles* apparaît dans (Rossi & Sperduti, 2004), mais la construction du solveur n'est pas non plus proposée. Dans l'article (Bessière *et al.*, 2004a) les auteurs présentent un algorithme pour l'apprentissage de contraintes basé sur l'espace de versions. Cet algorithme construit un CSP (trouve un sous-ensemble de contraintes prédéfinies) pour représenter la contrainte cible. L'opérateur correspondant à la contrainte cible est la composition par itération chaotique des opérateurs associés aux contraintes du CSP obtenu. Dans cette approche les auteurs supposent que la bibliothèque des contraintes de base utilisée pour la modélisation est bien adaptée, ce qu'il est difficile d'assurer pour les contraintes ouvertes du monde réel, souvent mal connues. D'autre part la technique proposée est très sensible au bruit, fréquent dans les jeux de données rencontrés en pratique. L'apprentissage de solveurs pour les contraintes classiques (fermées) a été introduit par (Apt & Monfroy, 1999). Leur algorithme construit un solveur à base de règles, mais le traitement de grandes contraintes est impossible pour les raisons de complexité. Ce travail a été étendu par (Abdennadher & Rigotti, 2004) et (Lallouet *et al.*, 2003) toujours dans le contexte des contraintes fermées. Aucune de ces approches ne combine la possibilité de généralisation avec les performances du solveur obtenu. A notre connaissance il n'y a pas d'autres techniques alternatives de construction de solveurs pour une contrainte ouverte provenant du monde réel.

Les contraintes ouvertes permettent d'élargir considérablement les possibilités de modélisation par contraintes de problèmes réels. Plusieurs problèmes de décision ou/et d'optimisation font intervenir les contraintes connues en partie, comme ensemble d'exemples et de contre-exemples. Dans ce travail, nous proposons une nouvelle technique d'apprentissage de solveur pour une contrainte ouverte. Cette technique basée sur l'apprentissage d'un ensemble de classificateurs particuliers est directement inspirée par le formalisme des contraintes et la nature des opérateurs de propagation. La technique permet non seulement la construction d'un classificateur à deux classes aux performances intéressantes, mais aussi la construction d'un solveur puissant pour la contrainte apprise. La technique est applicable aux contraintes de grande arité et les domaines finis de taille assez importante, ce qui la rend particulièrement intéressante. Les développements envisagés de ce travail sont :

– l'utilisation d'autres méthodes de classification que les réseaux de neurones avec ensuite l'extension aux intervalles ou aux ensembles

– l'application du principe général aux contraintes continues et mixtes.

Ces travaux sont en ce moment en cours.

Remerciement.

Merci à Anna, notre expert culinaire, pour la préparation de la base "salades".

Références

ABDENNADHER S. & RIGOTTI C. (2004). Automatic generation of rule-based constraint solvers over finite domains. *ACM TOCL*, **5**(2).

ALBERTI M., GAVANELLI M., LAMMA E., MELLO P. & MILANO M. (to appear). A chr-based implementation of known arc-consistency. *Theory and Practice of Logic Programming*.

APT K. (1999). The essence of constraint propagation. *Theoretical Computer Science*, **221**(1-2), 179–210.

APT K. & MONFROY E. (1999). Automatic generation of constraint propagation algorithms for small finite domains. In J. JAFFAR, Ed., *International Conference on Principles and Practice of Constraint Programming*, volume 1713 of *LNCS*, p. 58–72, Alexandria, Virginia, USA : Springer.

BESSIÈRE C., COLETTA R., FREUDER E. C. & O'SULLIVAN B. (2004a). Leveraging the learning power of examples in automated constraint acquisition. In M. WALLACE, Ed., *Principles and Practice of Constraint Programming*, volume 3258 of *LNCS*, p. 123–137, Toronto, Canada : Springer.

BESSIÈRE C., HEBRARD E., HNICH B. & WALSH T. (2004b). The complexity of global constraints. In D. L. MCGUINNESS & G. FERGUSON, Eds., *National Conference on Artificial Intelligence*, p. 112–117, San Jose, CA, USA : AAAI Press / MIT Press.

FALTINGS B. & MACHO-GONZALEZ S. (2002). Open constraint satisfaction. In P. VAN HENTENRYCK, Ed., *International Conference on Principles and Practice of Constraint Programming*, volume 2470 of *LNCS*, p. 356–370, Ithaca, NY, USA : Springer.

FREUND Y. & SHAPIRE R. (1999). A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, **14**(5), 771–780.

LALLOUET A., DAO T.-B.-H., LEGTCHENKO A. & ED-DBALI A. (2003). Finite domain constraint solver learning. In G. GOTTLOB, Ed., *International Joint Conference on Artificial Intelligence*, p. 1379–1380, Acapulco, Mexico : AAAI Press. Poster.

LALLOUET A., LEGTCHENKO A., MONFROY E. & ED-DBALI A. (2004). Solver learning for predicting changes in dynamic constraint satisfaction problems. In K. B. CHRIS BECK & G. VERFAILLIE, Eds., *Changes'04, International Workshop on Constraint Solving under Change and Uncertainty*, Toronto, CA.

MITCHELL T. M. (1997). *Machine Learning*. McGraw-Hill.

MOORE R. E. (1966). *Interval Analysis*. Prentice Hall.

QUINLAN J. (1993). *C4.5 : Programs for Machine Learning*. Morgan Kaufmann.

ROSSI F. & SPERDUTI A. (2004). Acquiring both constraint and solution preferences in interactive constraint system. *Constraints*, **9**(4).

RULEQUEST RESEARCH (2004). See5 : An informal tutorial. http ://www.rulequest.com/see5-win.html.

TSANG E. (1993). *Foundations of Constraint Satisfaction*. Academic Press.

Policy gradient in continuous time

Rémi Munos

Centre de Mathématiques Appliquées,
Ecole Polytechnique, 91128 Palaiseau, France.
remi.munos@polytechnique.fr
www.cmap.polytechnique.fr/∼munos

Abstract : We consider the approach of solving approximately a deterministic optimal control problem by searching a good controller in a given class of parameterized policies.

When the dynamics of the system is known from the decision maker, an explicit representation of the sensitivity of the performance measure with respect to the control parameters is easily derived using pathwise derivation, which enables to use gradient methods for solving the parametric optimization problem.

This paper is concerned with the case of an *unknown state dynamics* (such as in reinforcement learning). It introduces a method for computing the policy gradient only from the observable. The underlying idea consists in approximating the continuous deterministic process by a stochastic discrete one and using stochastic policies to estimate the unknown coefficients by quantities that depend solely on the state and the policy. Almost sure convergence to the policy gradient is proved. The method is illustrated on a (6 dimensional) target problem.

Résumé : Nous considérons la résolution approchée d'un problème de contrôle optimal par un problème d'optimisation dans un espace de politiques paramétrées.

Lorsque la dynamique d'état du système est connue de l'agent décisionnel, une représentation explicite de la sensibilité (le gradient) de la mesure de performance par rapport aux paramètres de contrôle se déduit facilement par dérivation trajectorielle, ce qui permet d'utiliser une méthode de gradient pour résoudre le problème d'optimisation paramétrique.

Dans cet article, nous considérons le cas d'une *dynamique d'état inconnue* (cadre de l'apprentissage par renforcement). Nous décrivons une méthode pour calculer le gradient uniquement à partir des observables. L'idée sous-jacente consiste à approcher le processus déterministe continu par un processus stochastique discret et d'utiliser une politique stochastique pour remplacer les coefficients inconnus par des grandeurs qui ne dépendent que de l'état et de la politique. La convergence presque-sure vers le gradient est montrée. L'approche est illustrée sur un problème de cible (en dimension 6).

1 Introduction and statement of the problem

We consider an optimal control problem with continuous state $(x_t \in I\!R^d)_{t\geq 0}$ whose dynamics is defined by the differential equation:

$$\frac{dx_t}{dt} = f(x_t, u_t), \tag{1}$$

where the control $(u_t)_{t\geq 0}$ is a Lebesgue measurable function with values in a control space U, assumed to be finite (extension to a continuous control space is straightforward). The functional J to be maximized has finite-time horizon T. For simplicity, in the following, we illustrate the case of a terminal reward only:

$$J(x; (u_t)_{t\geq 0}) = r(x_T), \tag{2}$$

where $r : I\!R^d \to I\!R$ is the reward function. Extension to the general case of a functional of the form

$$J(x; (u_t)_{t\geq 0}) = \int_0^T r(t, x_t)dt + R(x_T), \tag{3}$$

with r and R being current and terminal reward functions, easily follows from linearity, as indicated in remark 1.

The optimal control problem of finding a control $(u_t)_{t\geq 0}$ that maximizes the functional is replaced by a parametric optimization problem in which we search for a good feed-back control law in a given class of parameterized policies $\{\pi_\alpha\}_\alpha$, where $\alpha \in I\!R^m$ represents the parameter. The control (or action) may be written $u_t = \pi_\alpha(t, x_t)$, and the dynamics of the resulting feed-back system is

$$\frac{dx_t}{dt} = f_\alpha(x_t), \tag{4}$$

where $f_\alpha(x_t) = f(x, \pi_\alpha(t, x))$. We assume that f_α is $\mathcal{C}^2$ with bounded derivatives. Let us write the performance measure

$$V(\alpha) = J(x; \pi_\alpha(t, x_t)_{t\geq 0})$$

to emphasize the dependency with respect to (w.r.t.) the parameter α. One may also consider an average performance measure defined by some distribution μ for the initial state: $V(\alpha) = \mathbb{E}[J(x; \pi_\alpha(t, x_t)_{t\geq 0})|x \sim \mu]$.

In order to find a local maximum of $V(\alpha)$, one may perform a gradient ascent method:

$$\alpha \leftarrow \alpha + \eta \nabla_\alpha V(\alpha), \tag{5}$$

with η being an adequate step (see for example (Polyak, 1987; Kushner & Yin, 1997)). Of course, many much powerful variants of gradient-based methods exists (Bonnans *et al.*, 2003).

The computation of the gradient $\nabla_\alpha V(\alpha)$ is the object of this paper.

Pathwise estimation of the gradient.

Define the gradient of the state with respect to the parameter: $z_t = \nabla_\alpha x_t$, which solves

$$\frac{dz_t}{dt} = \nabla_\alpha f_\alpha(x_t) + \nabla_x f_\alpha(x_t) z_t, \tag{6}$$

with initial condition $z_0 = 0$. Here, z_t is an $d \times m$-matrix whose (i,j)-component is the derivative of the ith component of x_t w.r.t. α_j. Similarly $\nabla_\alpha f_\alpha$ and $\nabla_x f_\alpha$ are, respectively, the derivatives of f w.r.t. the parameter (matrix of size $d \times m$) and the state (matrix of size $d \times d$).

When the reward function r is smooth, one may apply a pathwise differentiation to derive a gradient formula:

Proposition 1
(Yang & Kushner, 1991). If r is continuously differentiable then

$$\nabla_\alpha V(\alpha) = \nabla_x r(x_T) z_T. \tag{7}$$

Remark 1
In the more general setting of a functional (3), the gradient is:

$$\nabla_\alpha V(\alpha) = \int_0^T \nabla_x r(t, x_t) z_t dt + \nabla_x R(x_T) z_T.$$

What is known from the agent?

The decision maker (call it the agent) that intends to design a good controller for the dynamical system may or may not know a model of the state dynamics f. In case the dynamics is known, the state gradient $z_t = \nabla_\alpha x_t$ may be computed from (6) along the trajectory and the gradient of the performance measure w.r.t. the parameter α may be deduced at time T from (7), which allows to perform the gradient ascent step (5).

In this paper we investigate the case of an a priori unknown dynamics: the agent only observes the response of the system to its control. This specific framework is often referred to as *model-free reinforcement learning* (Sutton & Barto, 1998).

The available information to the agent at time t are its own control policy π_α and the trajectory $(x_s)_{0 \leq s \leq t}$ up to time t. At time T, it observes the reward $r(x_T)$ and, in this paper, we also assume that the gradient $\nabla r(x_T)$ is known.

From this point of view, it seems impossible to derive the state gradient z_t from (6), since $\nabla_\alpha f$ and $\nabla_x f$ are unknown. The term $\nabla_x f(x_t)$ may be approximated by least squares methods from the observation of past states $(x_s)_{s \leq t}$, as this will be explained later on in subsection 3.2. However the term $\nabla_\alpha f(x_t)$ cannot be computed analogously.

In this paper, we introduce the idea of using stochastic policies to approximate the state x_t and the state gradient z_t by discrete-time stochastic processes X_t^Δ and Z_t^Δ (with Δ being some time-discretization step). We show how Z_t^Δ may be computed without the knowledge of $\nabla_\alpha f$, but instead with likelihood ratios (such as $\nabla_\alpha \log \pi_\alpha$ and $\nabla_x \log \pi_\alpha$) of the policy.

We prove the convergence (with probability one) of the gradient estimate $\nabla_x r(X_T^\Delta) Z_T^\Delta$ derived from the stochastic processes to $\nabla_\alpha V(\alpha)$ when $\Delta \to 0$.

Remark 2
It is worth mentioning that this strong convergence result contrasts with usual likelihood ratio methods in discrete time (Williams, 1992; Baxter & Bartlett, 2001; Sutton et al.*, 2000; Marbach & Tsitsiklis, 2003) for which the policy gradient estimate would be subject to variance explosion when the number of discretization steps (thus the number of decisions before getting the reward) goes to infinity (i.e.* $\Delta \to 0$*).*

The paper is organized as follows. In Section 2, we state a general result for discretization of continuous deterministic dynamics by stochastic discrete processes and apply it to prove the convergence of the approximate state and state gradient. In Section 3, we state the convergence of the policy gradient estimate and describe the model-free reinforcement learning algorithm. In the last Section, we illustrate the method on a (6 dimensional) target problem. Appendices A and B provide all proofs.

2 Discretized stochastic processes

We first state a general result for approximating a deterministic continuous process by a stochastic discrete one. Then, we apply this result to the convergence analysis of processes (the state X_t^Δ and the state gradient Z_t^Δ) related to the introduction of stochastic policies.

2.1 A general convergence result

Let $(x_t)_{0\leq t\leq T}$ be a deterministic continuous process defined by some dynamics

$$\frac{dx_t}{dt} = f(x_t)$$

with initial condition $x_0 = x$. We assume that f is of class $\mathcal{C}^2$ with bounded derivatives. Let $\Delta = T/N$ be a time-discretization step (with N being the number of time-steps) and denote $\{t_n = n\Delta\}_{0\leq n\leq N}$ the discrete times.

Let $(X_{t_n}^\Delta)_{0\leq n<N}$ be a discrete stochastic process, such that $X_0^\Delta = x$, and which satisfies the consistency property:

$$\mathbb{E}[X_{t+\Delta}^\Delta - X_t^\Delta | X_t^\Delta = x] = f(x)\Delta + o(\Delta) \tag{8}$$

and the following bound on the jumps:

$$X_{t+\Delta}^\Delta - X_t^\Delta = O(\Delta). \tag{9}$$

The following theorem establishes the convergence of (X_t^Δ) to (x_t) when $\Delta \to 0$.

Theorem 1
We have

$$\lim_{\Delta\to 0} X_T^\Delta = x_T, \text{ with probability } 1.$$

Appendix A gives a proof of this result. Note that a weaker convergence result (i.e. convergence in probability) may be obtained from general results in approximation of diffusion processes by Markov chains (Kloeden & Platen, 1995). Here, almost sure convergence is obtained using the *concentration of measure phenomenon* (Talagrand, 1996; Ledoux, 2001), detailed in Appendix A.

Remark 3
If we assume a slightly better consistency error of $O(\Delta^2)$ instead of $o(\Delta)$ in (8), then we may prove (straightforwardly from the Appendix) that $\mathbb{E}[X_T^\Delta] = x_T + O(\Delta)$ and $\mathbb{E}[||X_T^\Delta - x_T||^2] = O(\Delta)$.

2.2 Discretization of the state

Let us go back to our initial control problem (1). We define a **stochastic policy** π_α as a random choice of an action (or control) according to some probabilities: $\pi_\alpha(u|t,x)$ denotes the probability of choosing action $u \in U$ at time t in state x. We write:

$$u_t \sim \pi_\alpha(\cdot|t, x_t)$$

a choice of an action u_t at time t and state x_t, according to such a stochastic policy.

Now, we define the stochastic discrete process $(X_{t_n}^\Delta)_{0\leq n\leq N}$ (using the same notations for the time-steps (t_n) as in the previous subsection) according to:

- Initialize $X_0^\Delta = x$.
- At time $t \in \{(t_n)_{0\leq n<N}\}$, we choose an action $u_t \sim \pi_\alpha(\cdot|t, X_t^\Delta)$. Then, $X_{t+\Delta}^\Delta$ is the state of the system at time $t+\Delta$ resulting from keeping the action u_t constant for a period of time Δ. We write:

$$\left\{ \begin{array}{rcl} u_t & \sim & \pi_\alpha(\cdot|t, X_t^\Delta) \\ X_{t+\Delta}^\Delta & = & y(t, X_t^\Delta; u_t, \Delta) \end{array} \right. \tag{10}$$

  where $y(t,x;u,s)$ represents the state resulting from the state dynamics (1) with initial condition $x_t = x$ and using a constant control u for a period of time s.

When Δ is small, this process is close to a deterministic process $(x_t)_{0\leq t\leq T}$ defined by the dynamics (4) with

$$f_\alpha(x) := \sum_{u\in U} \pi_\alpha(u|t,x) f(x,u).$$

and initial condition $x_0 = x$. Indeed, the discrete stochastic process (X_t^Δ) converges to (x_t) when $\Delta \to 0$ as an immediate consequence of Theorem 1. To see that, we use Taylor formula,

$$X_{t+\Delta}^\Delta = X_t^\Delta + f(X_t^\Delta, u_t)\Delta + O(\Delta^2),$$

to derive the property on the average jumps:

$$\mathbb{E}[X_{t+\Delta}^\Delta - X_t^\Delta | X_t^\Delta = x] = \sum_{u\in U} \pi_\alpha(u|t,x) f(x,u)\Delta + O(\Delta^2) = f_\alpha(x)\Delta + O(\Delta^2),$$

and the consistency conditions (8) holds, as well as the bound on the jumps (9).

2.3 Discretization of the state gradient

Now, we discretize the state gradient $z_t = \nabla_\alpha x_t$. We build the discrete stochastic process $(Z^\Delta_{t_n})_{0\leq n\leq N}$ according to:

- Initialize $Z^\Delta_0 = 0$.
- At time $t \in \{(t_n)_{0\leq n<N}\}$, let (u_t) and (X^Δ_t) be defined according to (10). Then

$$\begin{aligned} Z^\Delta_{t+\Delta} &= Z^\Delta_t + f(X^\Delta_t, u_t)\big[l_\alpha(t, X^\Delta_t, u_t)' + l_x(t, X^\Delta_t, u_t)' Z^\Delta_t\big]\Delta \\ &\quad + \nabla_x f(X^\Delta_t, u_t) Z^\Delta_t \Delta, \end{aligned} \tag{11}$$

  where

$$l_\alpha(t,x,u) := \frac{\nabla_\alpha \pi_\alpha(u|t,x)}{\pi_\alpha(u|t,x)} \text{ and } l_x(t,x,u) := \frac{\nabla_x \pi_\alpha(u|t,x)}{\pi_\alpha(u|t,x)}$$

  are the likelihood ratios of π_α w.r.t. α and x (defined as vectors of size m and d respectively), and $'$ denotes the transpose operator.

Here again, as a consequence of Theorem 1, the process (Z^Δ_t) converges almost surely to (z_t) when $\Delta \to 0$. Indeed, from the property

$$\begin{aligned} \mathbb{E}[Z^\Delta_{t+\Delta} - Z^\Delta_t | X^\Delta_t = x, Z^\Delta_t = z] &= \sum_{u\in U} \pi_\alpha(u|t,x)\big(f(x,u)[l_\alpha(t,x,u)' + l_x(t,x,u)' z] \\ &\quad + \nabla_x f(x,u) z\big)\Delta \\ &= \big[\nabla_\alpha f_\alpha(x) + \nabla_x f_\alpha(x) z\big]\Delta, \end{aligned}$$

we deduce that the coupled process (X^Δ_t, Z^Δ_t) is consistent with (x_t, z_t) in the sense of (8):

$$\mathbb{E}\left[\begin{pmatrix} X^\Delta_{t+\Delta} \\ Z^\Delta_{t+\Delta}\end{pmatrix} - \begin{pmatrix} X^\Delta_t \\ Z^\Delta_t\end{pmatrix} \,\middle|\, \begin{pmatrix} X^\Delta_t \\ Z^\Delta_t\end{pmatrix} = \begin{pmatrix} x \\ z\end{pmatrix}\right] = \begin{pmatrix} f_\alpha(x) \\ \nabla_\alpha f_\alpha(x) + \nabla_x f_\alpha(x) z\end{pmatrix}\Delta + o(\Delta) \tag{12}$$

and $X^\Delta_{t+\Delta} - X^\Delta_t = O(\Delta)$ and $Z^\Delta_{t+\Delta} - Z^\Delta_t = O(\Delta)$.

3 Model-free reinforcement learning algorithm

We show how to use the approximation results of the previous section to design a model-free reinforcement learning algorithm for estimating the policy gradient $\nabla_\alpha V(\alpha)$ using one trajectory. First, we state the convergence of the policy gradient estimate computed from the discrete process, then show how to approximate the unknown coefficient $\nabla_x f$ using least-squares regression from the observed trajectory, and finally describe the reinforcement learning algorithm.

3.1 Convergence of the policy gradient estimate

One may use formula (7) to define a gradient estimate of the performance measure w.r.t. the parameter α based on the discrete process (X_t^Δ, Z_t^Δ):

$$g(\Delta) := \nabla_x r(X_T^\Delta) Z_T^\Delta. \tag{13}$$

Proposition 2
Assume that r is continuously differentiable. Then

$$\lim_{\Delta \to 0} g(\Delta) = \nabla_\alpha V(\alpha) \text{ with probability } 1.$$

Proof. This is a direct consequence of the almost sure convergence of (X_T^Δ, Z_T^Δ) to (x_T, z_T) and the continuity of $\nabla_x r$. □

Now, let us illustrate how Z_t^Δ may be approximated with quantities available to the agent. The definition (11) of Z_t^Δ requires the term $\nabla_x f(X_t^\Delta, u)$. We now explain how to built a consistent approximation $\widehat{\nabla_x f}(X_t^\Delta, u)$ of this term based on the past of the trajectory $(X_s^\Delta)_{0 \le s \le t}$.

3.2 Least-squares approximation of $\nabla_x f(X_t^\Delta, u)$

For clarity, we omit references to Δ, for example writing X_s instead of X_s^Δ. Let us write $\Delta X_t = X_{t+\Delta} - X_t$ the jumps of the state. Define $S(t) := \{s \le t, u_s = u_t, X_s - X_t = O(\Delta)\}$ the set of past times $s \le t$ when action u_t have been chosen, and such that the distance between the states X_s and X_t is bounded by a constant times Δ. From Taylor formula, we have for all $s \in S(t)$,

$$\Delta X_s = X_{s+\Delta} - X_s = f(X_s, u_t)\Delta + \nabla_x f(X_s, u_t) f(X_s, u_t) \frac{\Delta^2}{2} + O(\Delta^3). \tag{14}$$

Now, since

$$f(X_s, u_t) = f(X_t, u_t) + \nabla_x f(X_t, u_t)(X_s - X_t) + O(\Delta^2),$$

we deduce (using the fact that $\nabla_x f(X_s, u_t) = \nabla_x f(X_t, u_t) + O(\Delta)$) that

$$\begin{aligned}
\Delta X_s &= \Delta X_t + \left[\nabla_x f(X_s, u_t) f(X_s, u_t) - \nabla_x f(X_t, u_t) f(X_t, u_t)\right] \frac{\Delta^2}{2} \\
&\quad + \nabla_x f(X_t, u_t)(X_s - X_t)\Delta + O(\Delta^3) \\
&= \Delta X_t + \nabla_x f(X_t, u_t)[X_s - X_t + \frac{1}{2}(\Delta X_s - \Delta X_t)]\Delta + O(\Delta^3) \quad (15) \\
&= b + A(X_s + \frac{1}{2}\Delta X_s)\Delta + O(\Delta^3)
\end{aligned}$$

with $b := \Delta X_t - \nabla_x f(X_t, u_t)(X_t + \frac{1}{2}\Delta X_t)\Delta$ and $A := \nabla_x f(X_t, u_t)$. Based on the observation of several jumps $\{\Delta X_s\}_{s \in S(t)}$, one may derive an approximation of

$\nabla_x f(X_t, u_t)$ by solving the least-squares problem:

$$\min_{A,b} \frac{1}{n_t} \sum_{s \in S(t)} \left\| \Delta X_s - b - A\big(X_s + \frac{1}{2}\Delta X_s\big)\Delta \right\|^2, \tag{16}$$

where n_t is the cardinality of $S(t)$. Write $X_s^+ := X_s + \frac{1}{2}\Delta X_s = \frac{1}{2}(X_s + X_{s+\Delta})$ and use the simplified notations: $\overline{X}$, $\overline{X\,X'}$, $\overline{\Delta X}$, and $\overline{\Delta X\,X'}$, to denote the average values, when $s \in S(t)$, of X_s^+, $X_s^+(X_s^+)'$, ΔX_s, and $\Delta X_s(X_s^+)'$, respectively. For example,

$$\overline{X} := \frac{1}{n_t} \sum_{s \in S(t)} X_s^+.$$

The optimality condition of (16) holds when the matrix $Q_t := \overline{X\,X'} - \overline{X}\,\overline{X}'$ is invertible, and in that case, the least squares solution provides the approximation $\widehat{\nabla_x f}(X_t, u_t)$ of $\nabla_x f(X_t, u_t)$:

$$\widehat{\nabla_x f}(X_t, u_t) = \frac{1}{\Delta}\big(\overline{\Delta X\,X'} - \overline{\Delta X}\,\overline{X}'\big)\big(\overline{X\,X'} - \overline{X}\,\overline{X}'\big)^{-1}. \tag{17}$$

This optimality condition does not hold when the set of points $(X_s^+)_{s \in S(t)}$ lies in a vector space of dimension $< d$ (then Q_t is degenerate). In order to circumvent this problem, we assume that the eigenvalues of the matrix Q_t are bounded away from 0, in the sense given in the following proposition (whose proof in given in Appendix B).

Proposition 3
The matrix $Q_t = \overline{X\,X'} - \overline{X}\,\overline{X}'$ is symmetric non-negative. Let $\nu(\Delta) \geq 0$ be the smallest eigenvalue of Q_t, for all $0 \leq t \leq T$. Then, if $\nu(\Delta) > 0$ and $\nu(\Delta)$ satisfies

$$\frac{1}{\nu(\Delta)} = o(\Delta^4), \tag{18}$$

then, for all $0 \leq t \leq T$, the least squares estimate $\widehat{\nabla_x f}(X_t, u_t)$ defined by (17) is consistent with the gradient $\nabla_x f(X_t, u_t)$, that is:

$$\lim_{\Delta \to 0} \widehat{\nabla_x f}(X_t, u_t) = \nabla_x f(X_t, u_t).$$

The condition (18) is not easy to check, since it depends on the state dynamics and the policy. However, it seems (at least in our experiments) that its holds in general, specially since we used stochastic policies. Future work should focus on its well-foundedness.

3.3 The reinforcement learning algorithm

Here, we derive an policy gradient estimate from (13) in which all information required to build the state gradient Z_t is the past trajectory $(X_s)_{0 \leq s \leq t}$.

Choose a time step Δ. For a given stochastic policy π_α, the algorithm proceeds as follows:

- At time $t = 0$, initialise $X_0 = x$ and $Z_0 = 0$.

- At time $t \in \{(t_n)_{0 \leq n < N}\}$, choose a random action u_t according to the stochastic policy π_α and keep this action for a period of time Δ, which moves the system from X_t to $X_{t+\Delta}$ (summarized by the dynamics (10)). Update Z_t according to

$$\begin{aligned} Z_{t+\Delta} &= Z_t + \Delta X_t \big[l_\alpha(t, X_t, u_t)' + l_x(t, X_t, u_t)' Z_t \big] \Delta \\ &\quad + \widehat{\nabla_x f}(t, X_t, u_t) Z_t \Delta \end{aligned} \tag{19}$$

  where $\widehat{\nabla_x f}(t, X_t, u_t)$ is processed from (17).

- At time T, return the policy gradient estimate $\nabla_x r(X_T) Z_T$.

This algorithm returns a consistent approximation of the policy gradient $\nabla_\alpha V(\alpha)$, as stated in the next Proposition.

Proposition 4
Assume that the property (18) of Proposition 3 holds, and that r is continuously differentiable. Then the estimate $\nabla_x r(X_T) Z_T$ returned by the algorithm converges to $\nabla_\alpha V(\alpha)$ with probability 1, when $\Delta \to 0$.

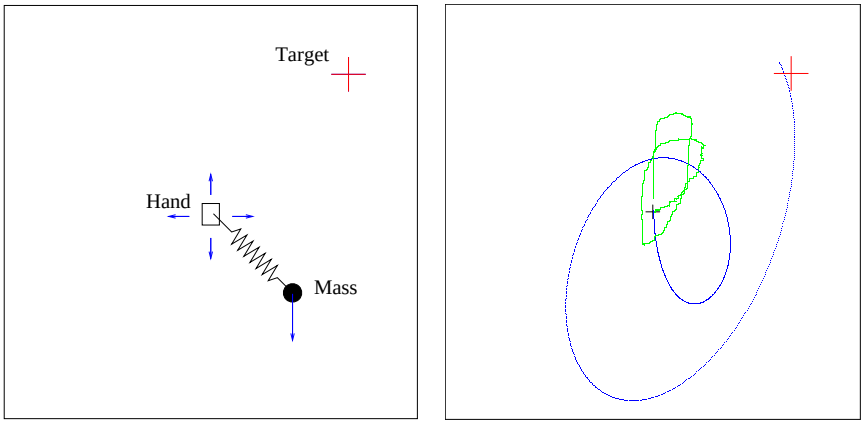
Proof. From Proposition 3, $\widehat{\nabla_x f}$ is a consistent approximation of $\nabla_x f$, thus the process (Z_t) built from (19) also satisfies the consistency condition (12), and the proof follows like in Proposition 2. □

Notice that a simple on-line way for approximating $\nabla_x f$ is to use an exponentially decreasing trace with coefficient $\lambda \in (0, 1)$. For that purpose, we may define a table with values $\overline{Y}(u)$ (where Y means X, XX', ΔX, and $\Delta X\, X'$) for all $u \in U$. The values are initialized (at the first time t each action u is chosen) by Y_t, where Y_t means X_t^+, $X_t^+ (X_t^+)'$, ΔX_t, and $\Delta X_t (X_t^+)'$, respectively. Then, the values are updated at time t, according to

$$\begin{array}{lll} \overline{Y}(u) \leftarrow \lambda \overline{Y}(u) + (1 - \lambda) Y_t & \text{for} & u = u_t, \\ \overline{Y}(u) \text{ stays unchanged} & \text{for} & u \neq u_t. \end{array}$$

Thus, the quantities $\overline{X}$, $\overline{X\, X'}$, $\overline{\Delta X}$, and $\overline{\Delta X\, X'}$ are easily updated and the term $\widehat{\nabla_x f}$ may be advantageously computed from (17) by an iterative matrix inversion, such as with Sherman-Morrison formula (see for example (Golub & Loan, 1996)).

Remark 4
Notice that for the initial discrete times t, the matrix $\overline{X\, X'} - \overline{X}\, \overline{X}'$ may not be invertible, simply because there are not enough points $(X_s)_{s<t}$ to define a subspace of dimension d. Then, we may simply set $\widehat{\nabla_x f}$ to 0, which has no impact on the general convergence result.



(a) The physical system

(b) A trajectory (the mass and the hand) starting from the origin

Figure 1: (a) the physical system. (b) A trajectory obtained after 1000 gradient steps.

4 Numerical experiment

We illustrate the algorithm described in the previous section on a 6 dimensional system $(x_0, y_0, x, y, v_x, v_y)$ that represents a hand ((x_0, y_0) position) holding a spring to which is attached a mass (defined by its position (x, y) and velocity (v_x, v_y)) subject to gravitation. The control is the movement of the hand, in any 4 possible directions (up, down, left, right). The goal is to reach a target (x_G, y_G) with the mass at a specific time T (see Figure 1a), while keeping the hand close to the origin. For that purpose, the terminal reward function is defined by

$$r = -x_0^2 - y_0^2 - (x - x_G)^2 - (y - y_G)^2.$$

The dynamics of the system is:

$$\begin{array}{lll} \dot{x_0} = u_x, & \dot{x} = v_x, & \dot{v_x} = -\frac{k}{m}(x - x_0), \\ \dot{y_0} = u_y, & \dot{y} = v_y, & \dot{v_y} = -\frac{k}{m}(y - y_0) - g, \end{array}$$

with k being the spring constant, m the mass, g the gravitational constant, and $(u_x, u_y) = u \in U := \{(1,0),(0,1),(-1,0),(0,-1)\}$ the control. We consider a stochastic policy of the form

$$\pi_\alpha(u|t,x) = \frac{\exp Q_\alpha(t,x,u)}{\sum_{u' \in U} \exp Q_\alpha(t,x,u')}$$

with a linear parameterization of the Q_α values: $Q_\alpha(t,x,u) = \alpha_0^u + \alpha_1^u t + \alpha_2^u x_0 + \alpha_3^u y_0 + \alpha_4^u x + \alpha_5^u y + \alpha_6^u v_x + \alpha_7^u v_y$, for each 4 possible actions $u \in U$. Thus the parameter $\alpha \in I\!R^{32}$. We initialized α with uniform random values in the range $[-0.01, 0.01]$. In

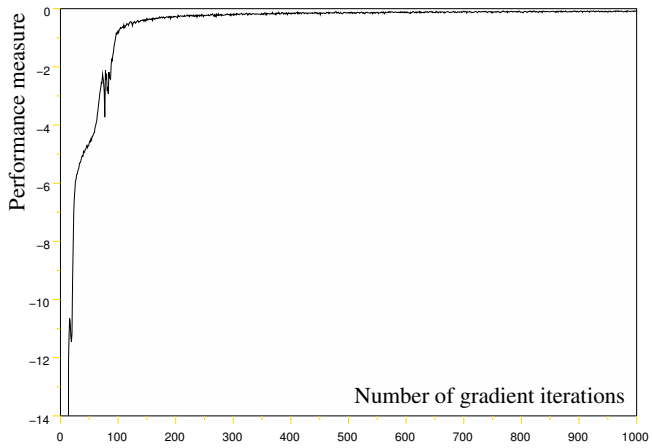


Figure 2: Performance of successive parameterized controllers (performance for the first iterations are below -14).

our experiments we chose $k = 1$, $m = 1$, $g = 1$, $x_G = y_G = 2$, $\lambda = 0.9$, $\Delta = 0.01$, $T = 10$.

At each iteration, we run one (stochastic) trajectory $(X_t)_{0 \leq t \leq T}$ using the stochastic policy, and calculate the policy gradient estimate according to the algorithm described in Section 3.3. We then perform a gradient ascent step (5) (with a fixed $\eta = 0.01$). Figure 2 shows the performance of the parameterized controller as a function of the number of gradient iterations.

For that problem, we chose initial states uniformly sampled from the domain $[-0.1, 0.1]^6$. We found that the randomness introduced in the choice of the initial state helped in not getting stuck in local minima. Here, convergence of the gradient method to an optimal controller (for which $r = 0$) occurs. We illustrate in Figure 1b the trajectory (where only the hand and the mass positions are showed) obtained after 1000 gradient steps, starting from the initial state $(x_0, y_0, x, y, v_x, v_y)_{t=0} = 0$.

5 Conclusion

We described a method for approximating the gradient of the performance measure of a continuous-time deterministic problem, with respect to the control parameters. This was obtained by discretizing the continuous dynamics by a consistent stochastic process built from using a stochastic policy. We showed how a consistent estimation of the gradient may be computed only from the observable.

In future work, it would be useful to extend this idea to stochastic dynamics, and to

non-smooth reward, or when its gradient is unknown (maybe using integration-by-part formula for the gradient estimate, such as in the *likelihood ratio method* of (Yang & Kushner, 1991) or the *martingale approach* of (Gobet & Munos, 2005)).

A Appendix: proof of Theorem 1

For convenience, we write x_n for x_{t_n}, X_n for $X^\Delta_{t_n}$, and u_n for u_{t_n}, $0 \leq n \leq N$. Let us define the average approximation errors $m^\Delta_n = \mathbb{E}[||X_n - x_n||]$ and the squared errors $v^\Delta_n = \mathbb{E}[||X_n - x_n||^2]$.

A.1 Convergence of the squared error $\mathbb{E}[||X^\Delta_T - x_T||^2]$:

We use the decomposition:

$$\begin{aligned} v^\Delta_{n+1} &= \mathbb{E}[||X_{n+1} - X_n||^2] + \mathbb{E}[||X_n - x_n||^2] + \mathbb{E}[||x_n - x_{n+1}||^2] \\ &\quad +2\mathbb{E}[(X_n - x_n)'(X_{n+1} - X_n + x_n - x_{n+1})] \\ &\quad +2\mathbb{E}[(X_{n+1} - X_n)'(x_n - x_{n+1})]. \end{aligned} \tag{20}$$

From the bounded jumps property (9), $\mathbb{E}[||X_{n+1} - X_n||^2] = O(\Delta^2)$. From Taylor formula,

$$x_{n+1} - x_n = f(x_n)\Delta + O(\Delta^2), \tag{21}$$

thus $\mathbb{E}[||x_n - x_{n+1}||^2] = O(\Delta^2)$ (since f is Lipschitz, and x_t and $f(x_t)$ are uniformly bounded on $[0, T]$) and from Cauchy-Schwarz inequality, $|\mathbb{E}[(X_{n+1} - X_n)'(x_n - x_{n+1})]| = O(\Delta^2)$. From (8) and (21),

$$\mathbb{E}[X_{n+1} - X_n + x_n - x_{n+1}|X_n] = [f(X_n) - f(x_n)]\Delta + o(\Delta). \tag{22}$$

Now, from (9) we deduced that $||X_n - x|| = O(1)$ thus X_n is bounded (for all n and N), as well as x_n. Let B a constant such that $||X_n|| \leq B$ and $||x_n|| \leq B$ for all $n \leq N$, $N \geq 0$. Since f is $\mathcal{C}^2$, from Taylor formula, there exists a constant k, such that, for all $n \leq N$,

$$||f(X_n) - f(x_n) - \nabla_x f(x_n)(X_n - x_n)|| \leq k||X_n - x_n||^2. \tag{23}$$

We deduce that

$$\begin{aligned} &|\mathbb{E}[(X_n - x_n)'(X_{n+1} - X_n + x_n - x_{n+1})]| \\ &= \left|\mathbb{E}\left[(X_n - x_n)'(f(X_n) - f(x_n))\right]\right| \Delta + o(\Delta) \\ &\leq \left|\mathbb{E}\left[(X_n - x_n)'\nabla_x f(x_n)(X_n - x_n)\right]\right| \Delta + 2kBv_n\Delta + o(\Delta) \\ &\leq Mv_n\Delta + o(\Delta) \end{aligned}$$

with $M = \sup_{||x|| \leq B} ||\nabla_x f(x)|| + 2kB$. Thus, (20) leads to the recurrent bound

$$v^\Delta_{n+1} \leq (1 + M\Delta)v^\Delta_n + o(\Delta).$$

This actually means that there exists a function $e(\Delta) \to 0$ when $\Delta \to 0$, such that $v_{n+1}^{\Delta} \leq (1 + M\Delta)v_n^{\Delta} + e(\Delta)\Delta$. Thus,

$$v_N^{\Delta} \leq \frac{(1+M\Delta)^N - 1}{(1+M\Delta) - 1} e(\Delta)\Delta \leq (e^{NM\Delta} - 1)\frac{1}{M} e(\Delta)$$

thus $v_N^{\Delta} = o(1)$, that is $\mathbb{E}[||X_T^{\Delta} - x_T||^2] \stackrel{\Delta \to 0}{\longrightarrow} 0$.

A.2 Convergence of the mean $\mathbb{E}[||X_T^{\Delta} - x_T||]$:

From (22), we have

$$\mathbb{E}[X_{n+1} - x_{n+1}|X_n] = X_n - x_n + [f(X_n) - f(x_n)]\Delta + o(\Delta).$$

Thus from (23),

$$\begin{aligned} m_{n+1}^{\Delta} = \mathbb{E}[||X_{n+1} - x_{n+1}||] &\leq (1 + ||\nabla_x f(x_n)||\Delta)\mathbb{E}[||X_n - x_n||] + k v_n^{\Delta}\Delta + o(\Delta) \\ &\leq (1 + M'\Delta) m_n^{\Delta} + o(\Delta), \end{aligned}$$

since $v_n^{\Delta} = o(1)$ (with $M' = \sup_{||x|| \leq B} ||\nabla_x f(x)||$). Using the same deduction as above, we obtain that $m_N^{\Delta} = o(1)$, that is $\mathbb{E}[||X_T^{\Delta} - x_T||] \stackrel{\Delta \to 0}{\longrightarrow} 0$.

A.3 Almost sure convergence

Here, we use the *concentration-of-measure phenomenon* (Talagrand, 1996; Ledoux, 2001), which states that under mild conditions, a function (say Lipschitz or with bounded differences) of many independent random variables concentrates around its mean, in the sense that the tail probability decreases exponentially fast.

One may write the state X_N as a function h of some independent random variables $(U_n)_{0 \leq n < N}$:

$$X_N = h(U_0, \ldots, U_{N-1}) := \sum_{n=0}^{N-1} X_{n+1} - X_n. \tag{24}$$

Observe that $h - \mathbb{E}[h] = \sum_{n=0}^{N-1} d_n$ with

$$\begin{aligned} d_n &= \mathbb{E}[h(U_0, \ldots, U_{N-1})|U_0, \ldots, U_n] - \mathbb{E}[h(U_0, \ldots, U_{N-1})|U_0, \ldots, U_{n-1}] \\ &= X_{n+1} - X_n - \mathbb{E}[X_{n+1} - X_n] \end{aligned}$$

being a martingale difference sequence (that is $\mathbb{E}[d_n|U_0, \ldots, U_{n-1}] = 0$). Now, from (Ledoux, 2001, lemma 4.1), one has:

$$\mathbb{P}(|g - \mathbb{E}[g]| \geq \varepsilon) \leq 2e^{-\varepsilon^2/(2D^2)} \tag{25}$$

for any $D^2 \geq \sum_{n=0}^{N-1} ||d_n||_{\infty}^2$. Thus, from (9), and since X_n is bounded, as well as $f(X_n)$ (for all $n < N$ and all $N > 0$), there exists a constant C that does not depend on N such that $d_n \leq C/N$. Thus we may take $D^2 = C^2/N$.

Now, from the previous paragraph, $|\mathbb{E}[X_N] - x_N| \leq e(N)$, with $e(N) \to 0$ when $N \to \infty$. This means that $|h - \mathbb{E}[h]| + e(N) \geq |X_N - x_N|$, thus

$$\mathbb{P}(|h - \mathbb{E}[h]| \geq \varepsilon + e(N)) \geq \mathbb{P}(|X_N - x_N| \geq \varepsilon),$$

and we deduce from (25) that

$$\mathbb{P}(|X_N - x_N| \geq \varepsilon) \leq 2e^{-N(\varepsilon+e(N))^2/(2C^2)}.$$

Thus, for all $\varepsilon > 0$, the series $\sum_{N\geq 0} \mathbb{P}(|X_N - x_N| \geq \varepsilon)$ converges. Now, from Borel-Cantelli lemma, we deduce that for all $\varepsilon > 0$, there exists N_ε such that for all $N \geq N_\varepsilon$, $|X_N - x_N| < \varepsilon$, which proves the convergence $X_N \to x_N$ as $N \to \infty$, with probability 1.

B Proof of proposition 3

First, note that $Q_t = \overline{X\,X'} - \overline{X}\,\overline{X}'$ is a symmetric, non-negative matrix, since it may be rewritten as

$$\frac{1}{n_t} \sum_{s\in S(t)} (X_s^+ - \overline{X})(X_s^+ - \overline{X})'.$$

In solving the least squares problem (16), we deduce $b = \overline{\Delta X} + A\overline{X}\Delta$, thus

$$\begin{aligned}
&\min_{A,b} \frac{1}{n_t} \sum_{s\in S(t)} ||\Delta X_s - b - A(X_s + \frac{1}{2}\Delta X_s)\Delta||^2 \\
&= \min_{A} \frac{1}{n_t} \sum_{s\in S(t)} ||\Delta X_s - \overline{\Delta X} - A(X_s^+ - \overline{X})\Delta||^2 \\
&\leq \frac{1}{n_t} \sum_{s\in S(t)} ||\Delta X_s - \overline{\Delta X} - \nabla_x f(\overline{X}, u_t)(X_s^+ - \overline{X})\Delta||^2. \qquad (26)
\end{aligned}$$

Now, since $X_s = \overline{X} + O(\Delta)$ one may obtain like in (14) and (15) (by replacing X_t by $\overline{X}$) that:

$$\Delta X_s - \overline{\Delta X} - \nabla_x f(\overline{X}, u_t)(X_s^+ - \overline{X})\Delta = O(\Delta^3). \qquad (27)$$

We deduce from (26) and (27) that

$$\frac{1}{n_t} \sum_{s\in S(t)} \left\| \left[\widehat{\nabla_x f}(X_t, u_t) - \nabla_x f(\overline{X}, u_t)\right](X_s^+ - \overline{X})\Delta \right\|^2 = O(\Delta^6).$$

By developping each component,

$$\sum_{i=1}^{d} \left[\widehat{\nabla_x f}(X_t, u_t) - \nabla_x f(\overline{X}, u_t)\right]_{row\,i} Q_t \left[\widehat{\nabla_x f}(X_t, u_t) - \nabla_x f(\overline{X}, u_t)\right]'_{row\,i} = O(\Delta^4).$$

Now, from the definition of $\nu(\Delta)$, for all vector $u \in I\!R^d$, $u'Q_t u \geq \nu(\Delta)||u||^2$, thus

$$\nu(\Delta) \sum_{i=1}^{d} ||\widehat{\nabla_x f}(X_t, u_t) - \nabla_x f(\overline{X}, u_t)||^2 = O(\Delta^4).$$

Condition (18) yields $\widehat{\nabla_x f}(X_t, u_t) = \nabla_x f(\overline{X}, u_t) + o(1)$, and since $\nabla_x f(X_t, u_t) = \nabla_x f(\overline{X}, u_t) + O(\Delta)$, we deduce

$$\lim_{\Delta \to 0} \widehat{\nabla_x f}(X_t, u_t) = \nabla_x f(X_t, u_t).$$

References

BAXTER J. & BARTLETT P. (2001). Infinite-horizon gradient-based policy search. *Journal of Artificial Intelligence Research*, **15**, 319–350.

BONNANS F., GILBERT J., LEMARÉCHAL C. & SAGASTIZABAL C. (2003). *Numerical Optimization. Theoretical and Practical Aspects*. Springer-Verlag.

GOBET E. & MUNOS R. (2005). Sensitivity analysis using itô-malliavin calculus and martingales. application to stochastic optimal control. *To appear in SIAM journal on Control and Optimization*.

GOLUB G. H. & LOAN C. F. V. (1996). *Matrix Computations, 3rd ed.* Baltimore, MD: Johns Hopkins.

KLOEDEN P. E. & PLATEN E. (1995). *Numerical Solutions of Stochastic Differential Equations*. Springer-Verlag.

KUSHNER H. J. & YIN G. (1997). *Stochastic Approximation Algorithms and Applications*. Springer-Verlag, Berlin and New York.

LEDOUX M. (2001). *The concentration of measure phenomenon*. American Mathematical Society, Providence, RI.

MARBACH P. & TSITSIKLIS J. N. (2003). Approximate gradient methods in policy-space optimization of markov reward processes. *Journal of Discrete Event Dynamical Systems*, **13**, 111–148.

POLYAK B. (1987). *Introduction to Optimization*. Optimization Software Inc., New York.

SUTTON R., MCALLESTER D., SINGH S. & MANSOUR Y. (2000). Policy gradient methods for reinforcement learning with function approximation. *Neural Information Processing Systems. MIT Press*, p. 1057–1063.

SUTTON R. S. & BARTO A. G. (1998). Reinforcement learning: An introduction. *Bradford Book*.

TALAGRAND M. (1996). A new look at independence. *Annals of Probability*, **24**, 1–34.

WILLIAMS R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, **8**, 229–256.

YANG J. & KUSHNER H. (1991). A Monte Carlo method for sensitivity analysis and parametric optimization of nonlinear stochastic systems. *SIAM J. Control Optim.*, **29**(5), 1216–1249.

A Multi-Objective Multi-Modal Optimization Approach for Mining Stable Spatio-Temporal Patterns.

Nicolas Tarrisson[1], Michèle Sebag[1], Olivier Teytaud[1], Julien Lefevre[2] & Sylvain Baillet[2]

[1] TAO, CNRS − INRIA − Université Paris-Sud, F-91405 - Orsay, {nom}@lri.fr

[2] LENA, CNRS, La Pitié Salpétrière, F-75651 - Paris

Résumé : Cet article, motivé par des applications en imagerie fonctionnelle cérébrale, traite de la découverte de motifs spatio-temporels stables. Ce problème est formalisé comme une optimisation multi-modale multi-objectif : d'une part, les motifs cibles doivent montrer une bonne stabilité dans une grande région spatio-temporelle (objectifs antagonistes) ; d'autre part, les experts souhaitent trouver tous les motifs de ce type, même des optima locaux.
L'algorithme proposé, appelé *4D-Miner*, est empiriquement validé sur des jeux de données réels et artificiels ; il montre de bonnes performances et une bonne capacité à passer à l'échelle, détectant des motifs spatio-temporels en quelques minutes dans fichiers de plus de 400 Mo.

1 Introduction

Spatio-temporal data mining is concerned with finding specific patterns in databases describing temporally situated spatial objects. Many approaches have been developed in signal processing and computer science to address such a goal, ranging from Fourier Transforms to Independent Component Analysis iciteICA, mixtures of models icitePadraic or string kernel machines iciteJST-strings, to name a few. These approaches aim at particular pattern properties (e.g. independence, generativity) and/or focus on particular data characteristics (e.g. periodicity). However, in some application domains, the above properties are not relevant to extract the patterns of interest.

The approach presented in this paper is motivated by such an application domain, functional brain imaging. Non invasive techniques, specifically magnetoencephalography (MEG) iciteHamalainen, provide measures of the human brain activity with an unprecedented temporal resolution (time step 1 millisecond). This resolution enables researchers to investigate the timing of basic neural processes at the level of cell assemblies iciteBaillet. A key task is to find out which brain areas are active and when, i.e. to detect spatio-temporal regions with highly correlated activities. It is emphasized that such regions, referred to as stable spatio-temporal patterns (STPs), are neither periodic nor necessarily independent. Currently, STPs are manually detected, which (besides being a tedious task) severely hampers the reproducibility of the results.

This paper addresses the automatic detection of stable spatio-temporal patterns, i.e. maximal spatio-temporal regions with maximal correlation. This detection problem cannot be formalized as a multi-objective optimization (MOO) problem iciteDeb :book, because experts are interested in several active brain areas : an STP might be relevant though it corresponds to a smaller spatio-temporal region, with a lesser correlated activity than another STP.

The proposed approach thus extends MOO along the lines of multi-modal optimization iciteMulti-modal, defining a *multi-objective multi-modal optimization* framework (MoMOO). MoMOO is tackled by an evolutionary algorithm termed *4D-Miner*, using a diversity criterion to relax the multi-objective search (as opposed to diversity enforcing heuristics in MOO, e.g. iciteCorne,Laumans ; more on this in section 2.3).

To the best of our best knowledge, both the extension of evolutionary algorithms to MoMOO, and the use of multi-objective optimization within spatio-temporal data mining are new, although MOO attracts increasing attention in the domain of machine learning and knowledge discovery (see, e.g., iciteGhosh,FRAN).

Experimental validation on artificial and real-world datasets demonstrates the good scalability and performances of *4D-Miner* ; sought STPs are found within minutes from medium sized datasets (450 Mo), on PC-Pentium 2.4 GHz.

The paper is organized as follows. Section 2 formalizes the detection of stable spatio-temporal patterns as a multi-modal multi-objective optimization problem, and motivates the use of evolutionary algorithms iciteBaeck-book,Goldberg :02 for their detection. Section 3 gives an overview of *4D-Miner*. Section 4 describes the experiment goals and setting. Section 5 reports on the extensive validation of *4D-Miner* on artificial and real-world data. Section 6 discusses the approach with respect to relevant work, and the paper ends with perspectives for further research.

2 Position of the problem

This section introduces the notations and criteria for stable spatio-temporal pattern detection.

2.1 Notations and definitions

Let N be the number of measure points. To the i-th measure point is attached a spatial position[1] $M_i = (x_i, y_i, z_i)$ and a temporal activity $C_i(t), t \in [1, T]$.

Let $I = [t_1, t_2] \subset [1, T]$ be a time interval, and let $\bar{C}_i^I$ denote the average activity of the i-th measure point during I. The I-alignment $\sigma_I(i, j)$ of measure points i and j over I is defined as :

$$\sigma_I(i,j) = < i,j >_I \times \left(1 - \frac{|\bar{C}_i^I - \bar{C}_j^I|}{|\bar{C}_i^I|}\right), \quad \text{where}$$

$$< i,j >_I = \frac{\sum_{t=t_1}^{t_2} C_i(t).C_j(t)}{\sqrt{\sum_{t=t_1}^{t_2} C_i(t)^2 \times \sum_{t=t_1}^{t_2} C_j(t)^2}}$$

[1]MEG measure points actually belong to a 2D shape (the surface of the skull). However, the approach equally handles 2D or 3D spatio-temporal data.

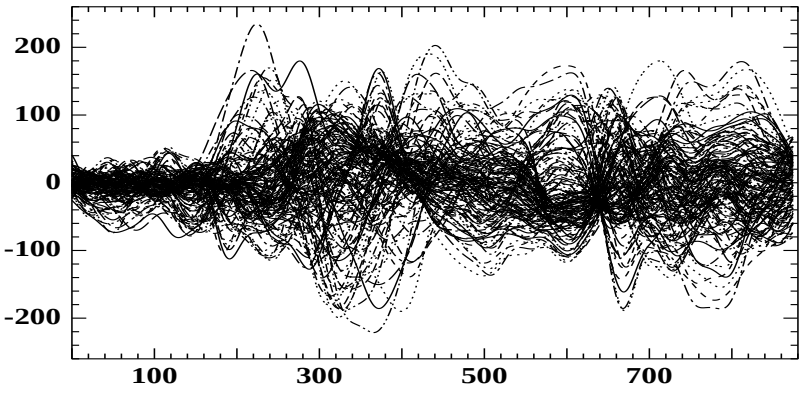


FIG. 1 – Magneto-Encephalography Data (N = 151, T = 875)

As the sought patterns do not need to be spheric, ellipsoidal distances are considered. Only axis-parallel ellipsoids will be considered throughout the paper. For each weight vector $w = (a, b, c)$ ($w \in \mathbb{R}^{+3}$), distance d_w is defined on the measure points as :

$$d_w(i,j) = \sqrt{a(x_i - x_j)^2 + b(y_i - y_j)^2 + c(z_i - z_j)^2}$$

2.2 Multi-objective formalization

A pattern $X = \{i, I, w, r\}$ is characterized by a center point i, $i \in [1, N]$, a time interval I, an ellipsoidal distance d_w, and a radius $r > 0$. The spatial neighborhood $\mathcal{B}(i, w, r)$ is defined as the set of measure points j such that $d_w(i, j)$ is less than r.

The spatial amplitude of X, noted $a(X)$, is the number of measure points in $\mathcal{B}(i, w, r)$. The temporal amplitude of X, noted $\ell(X)$, is the number of time steps in interval I. The spatio-temporal alignment of X, noted $\sigma(X)$ is defined as the average, over all measure points in $\mathcal{B}(i, w, r)$, of their I-alignment with respect to the center i :

$$\sigma(X) = \frac{1}{a(X)} \sum_{j \in \mathcal{B}(i,w,r)} \sigma_I(i, j)$$

Interesting spatio-temporal patterns, according to the expert's first specifications, show maximal spatial and temporal amplitudes together with a maximal spatio-temporal alignment. Specifically, a solution pattern is such that i) increasing its spatial or temporal amplitude would cause the spatio-temporal alignment to decrease ; ii) the only way of increasing its spatio-temporal alignment is by decreasing its spatial or temporal amplitude. It thus comes naturally to formalize the STP detection in terms of multi-objective optimization problem (see iciteDeb :book and references therein ; iciteIGLE,BHAT for related other forms of data-mining).

The three a, ℓ and σ criteria induce a partial order on the set of patterns, referred to as Pareto domination..

Definition 1. (Pareto domination)
Let $c_1, ...c_K$ be K real-valued criteria defined on domain Ω, and let X and Y belong to Ω.
X Pareto-dominates *Y, noted $X \succ Y$, iff $c_k(X)$ is greater or equal $c_k(Y)$ for all $k = 1..K$, and the inequality is strict for at least one k_0.*

$$(X \succ Y) \iff \begin{cases} \forall k = 1..K, (c_k(X) \geq c_k(Y)) & \textit{and} \\ \exists k_0 \ s.t. \ (c_{k_0}(X) > c_{k_0}(Y)) \end{cases}$$

The set of non-dominated solutions after a set of criteria is classically referred to as *Pareto front* iciteDeb :book.

2.3 Multi-modal multi-objective formalization

However, the above criteria only partially account for the expert's expectations : a STP might have a lesser spatio-temporal alignment and amplitude than another one, and still be worthy, provided that it corresponds to another active brain area. Therefore, not all sought STPs belong to the Pareto front.

Multi-modal optimization occurs when we are interested in solutions that are non-dominated locally. What we need here is both multi-objective and multi-modal. A new optimization framework is thus defined, extending multi-objective optimization in the spirit of multi-modal optimization iciteMulti-modal : *multi-modal multi-objective optimization* (MoMOO). Formally, let us first define a relaxed inclusion relationship, noted p-inclusion.

Definition 2. (p-inclusion)
Let A and B be two subsets of set Ω, and let p be a positive real number ($p \in [0, 1]$). A is p-included in B, noted $A \subset_p B$, iff $|A \bigcap B| > p \times |A|$.

Defining adequately the *support* of a candidate solution X (see below) multi-modal Pareto domination can be defined as follows :

Definition 3. (multi-modal Pareto domination)
With same notations as in Def. 1, X mo-Pareto dominates *Y, noted $X \succ_{mo} Y$, iff the support of Y is p-included in that of X, and X Pareto-dominates Y.*

$$X \succ_{mo} Y \iff [(Supp(Y) \subset_p Supp(X)) \textit{ and } (X \succ Y)]$$

In the case of STPs, the support of $X = (i, I, w, r)$ is naturally defined as $Supp(X) = \mathcal{B}(i, d_w, r) \times I$, with $Supp(X) \subset [1, N] \times [1, T]$.

In contrast with iciteCorne,Laumans who use diversity-based heuristics for a better sampling of the Pareto front, diversity is thus used in MoMOO to redefine and extend the set of solutions.

2.4 Discussion

Functional brain imaging sets two specific requirements on spatio-temporal data mining. First, the expert's requirements are subject to change. Much background knowledge is involved in the manual extraction of stable spatio-temporal patterns, e.g. about

the expected localization of the active brain areas. A flexible approach, accommodating further criteria and allowing the user to customize the search (in particular, tuning the thresholds on the minimal spatial or temporal amplitudes, or spatio-temporal alignment) is thus highly desirable.

Second, the approach must be scalable with respect to the data size (number of measure points and temporal resolution). Although the real data size is currently limited, the computational cost must be controllable in order to efficiently adjust the user-supplied parameters ; in other words, the mining algorithm must be an *any-time algorithm* iciteZilberstein.

The approach proposed is therefore based on evolutionary algorithms (EAs) ; these are widely known as stochastic, population-based optimization algorithms iciteBaeckbook,Goldberg :02 that are highly flexible. In particular, EAs address multi-modal optimization iciteMulti-modal and they can be harnessed to sample the whole Pareto front associated to a set of optimization criteria, with a moderate overhead cost compared to the standard approach (i.e., optimizing a weighted sum of the criteria gives a single point of the Pareto front) iciteDeb :book. Last, the computational resources needed by EAs can be controlled in a straightforward way through limiting the number of generations and/or the population size.

3 Overview of *4D-Miner*

This section describes the *4D-Miner* algorithm designed for the detection of stable spatio-temporal patterns.

3.1 Initialization

Following iciteDaida :GECCO99, special care is devoted to the initialization of this evolutionary algorithm. The extremities of the Pareto front, where STPs display a high correlation (respectively, a low correlation) on a small spatio-temporal region (resp. a wide region), do not fulfill the expert's expectations. Accordingly, some user-supplied thresholds are set on the minimal spatio-temporal amplitude and alignment.

In order to focus the search on relevant STPs, the initial population is generated using the initialization operator, sampling patterns $X = (i, w, I, r)$ as follows :

- Center i is uniformly drawn in $[1, N]$;
- Weight vector w is set to $(1, 1, 1)$ (initial neighborhoods are based on Euclidean distance) ;
- Interval $I = [t_1, t_2]$ is such that t_1 is drawn with uniform distribution in $[1, T]$; the length $t_2 - t_1$ of I_j is drawn according to a Gaussian distribution $\mathcal{N}(min_\ell, min_\ell/10)$, where min_ℓ is a time length threshold[2].
- Radius r is deterministically computed from a user-supplied threshold min_σ, corresponding to the minimal I-alignment desired.

$$r = min_k\{d_w(i,k)\ s.t.\ \sigma^I_{i,k} > min_\sigma)\}$$

[2] In case t_2 is greater than T, another interval I is sampled.

All patterns X whose spatial amplitude $a(X)$ is less than a user-supplied threshold min_a are non-admissible ; they will not be selected to generate new offspring. The user-supplied thresholds thus govern the proportion of usable individuals in the initial population.

The computational complexity is in $\mathcal{O}(P \times N \times min_\ell)$, where P is the population size, N is the number of measure points and min_ℓ is the average length of the intervals.

3.2 Variation operators

From parent $X = (i, w, I, r)$, mutation generates an offspring by one among the following operators :

- replacing center i with another measure point in $\mathcal{B}(i, w, r)$;
- mutating w and r using self-adaptive Gaussian mutation iciteBaeck-book ;
- incrementing or decrementing the bounds of interval I ;
- generating a brand new individual (using the initialization operator).

The crossover operator, starting from parent X, first selects the mate pattern $Y = (i', w', I', r')$ by tournament selection, Y minimizing the sum of the euclidean distance between i and i', and the distance between the center of I and I' among K patterns uniformly selected in the population, where K is set to $P/10$ in all the experiments. The offspring is generated by :

- replacing i with the center i' of the mate pattern Y ;
- replacing w (resp. r) using an arithmetic crossover of w and w' (respectively r and r') ;
- replacing I by the smallest interval containing I and I'.

An offspring is said admissible iff it satisfies the user-supplied thresholds mentioned in the initialization step.

3.3 Selection scheme

A Pareto archive is constructed with size L (set to $10 \times P$ in the experiments).

A steady-state scheme is used ; at each step, an admissible parent X is selected among K uniformly drawn individuals in the population, by retaining the one which is dominated by the fewest individuals in the archive (Pareto tournament iciteDeb :book). A single offspring Y is generated from X by applying a variation operator among the ones mentioned above.

Offspring Y is evaluated by computing criteria $a(Y), \ell(Y), \sigma(Y)$. Y is rejected if it is mo-Pareto dominated in the population ; otherwise, it replaces a non-admissible individual in the population if any ; if none it replaces an individual selected after anti-Pareto tournament (the individual out of K randomly selected ones in the population, that is dominated by the most individuals in the archive).

The archive is updated every P generations, replacing the mo-Pareto dominated individuals in the archive with individuals selected from the population after Pareto tournament.

4 Experimental setting and goal

This section presents the goal of the experiments, describes the artificial and real-world datasets used, and finally gives the parameters of the algorithm and the performance measures used to evaluate the algorithm.

4.1 Goals of experiments and datasets

The initial goal is to provide the expert with a usable STP detection algorithm. The real datasets have been collected from people observing a moving ball. Each dataset involves 151 measure points and the number of time steps is 875. As can be noted from Fig. 1, which shows a representative dataset, the amplitude of the activities widely vary along the time dimension.

The other goal is to assess the scalability and the performances of *4D-Miner*, which is done using artificial datasets.

The artificial datasets are generated as follows. N measure points are located in uniformly selected locations in the 3D domain $[0,1]^3$. Activities are initialized from random cumulative uniform noise, with $C_i(t+1) = C_i(t) + \epsilon$, and ϵ is drawn according to $U(0,\tau)$.

Every target STP $S = (i, w, I, r, \delta)$ is defined from a center i, a weight vector w, a time interval I, a radius r, and a fading factor δ, used to bias the activities as detailed below. The activity C_S of the STP S is the average activity in the spatio-temporal region $\mathcal{B}(i, w, r) \times I$.

Thereafter, activities are biased according to the target STPs : for each measure point j, for each STP S such that j is influenced by S ($d_w(i,j) < r$), the activity $C_j(t)$ is smoothed as

$$\begin{array}{ll} & C_j(t) = (1 - e^{-\alpha_i(j,t)})C_j(t) + e^{-\alpha_i(j,t)}C_S \\ \text{where} & \alpha_i(j,t) = d_w(i,j) + \delta \times d(t,I) \end{array}$$

and $d(t,I)$ is the distance of t to interval I (0 if t belongs to I, otherwise the minimum distance of t to the bounds of I).

The scalability of *4D-Miner* is assessed from its empirical complexity depending on the number N of measure points and the number T of time steps.

The performances of *4D-Miner* are measured using the standard criteria in information retrieval, recall and precision. The recall is the fraction of target STPs that are identified by *4D-Miner*, i.e. p-included in an individual in the archive; the precision is the fraction of relevant individuals in the archive (p-included in a target STP). For each experimental setting (see below), the recall and precision are averaged over 21 independent runs.

The number of target STPs is set to 10 in all experiments. Each STP influences a number of measure points varying in [10,20], during intervals uniformly selected in $[1,T]$ with length varying in [15, 25], and δ uniformly selected in $[0,.05]$. Each problem instance thus includes STPs with various levels of difficulty ; the detection is hindered as the spatio-temporal support (r and I) is comparatively low and the δ parameter increases (the regularity is not visible outside interval I, as in Fig 2).

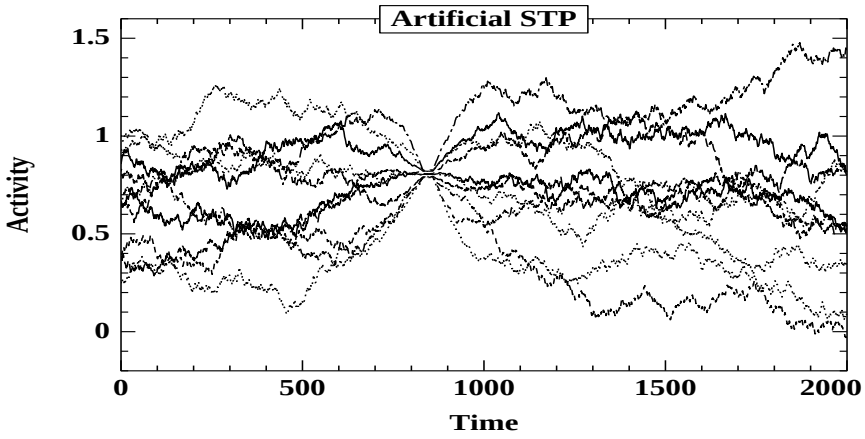


FIG. 2 – An artificial STP (T = 2000, N = 2000)

4.2 Experimental setting

The experiments reported in the next section considers a population size $P = 200$, evolved along 8000 generations (8,000 fitness evaluations per run). A few preliminary runs were used to adjust the operator rates ; the mutation and crossover rates are respectively set to .7 and .3.

The number N of measure points ranges in $\{500, 1000, 2000, 4000\}$. The number T of time steps ranges in $\{1000, 2000, 4000, 8000\}$.

The thresholds used in the initialization are : $min_a = 5$ (minimum number of curves supporting a pattern) ; $min_\ell = 5$ (minimum temporal amplitude of a pattern) ; $min_\sigma = .1$ (minimum spatio-temporal alignment of two curves in a pattern).

For computational efficiency, the p-inclusion is computed as : X is p-included in Y if the center i of X belongs to the spatial support of Y, and there is an overlap between their time intervals.

The maximal size of the datasets ($T = 8000,\ N = 4000$) is 456 Mo. Computational runtimes are given on PC-Pentium IV, 2.4 GHz. *4D-Miner* is written in C^{++}.

5 Experimental validation

This section reports on the experiments done using *4D-Miner*.

5.1 Experiments on real datasets

Typical STPs found in the real datasets are shown in Figs. 3 and 4, showing all curves belonging to the STP plus the time-window of the pattern. The runtime is less than 1 minute (PC Pentium 1.4 GHz). As discussed in section 2.3, many relevant STPs are Pareto-dominated : typically the STP shown in Fig 3 is dominated by the one in Fig 4.

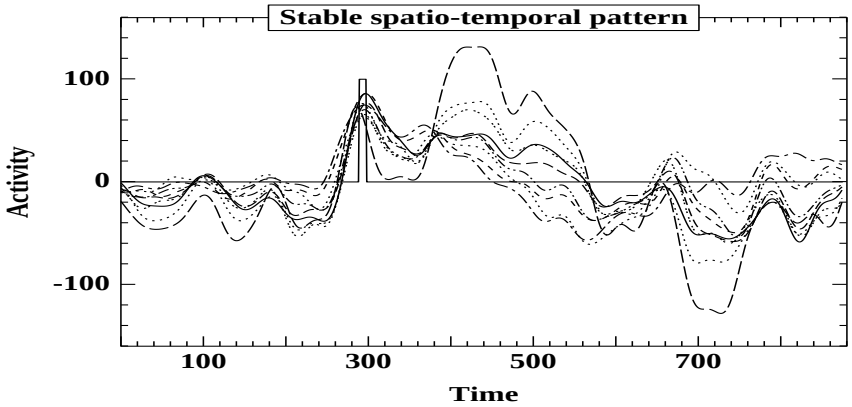


FIG. 3 – A stable spatio-temporal pattern, involving 8 measure points within interval [289,297], alignment .2930

These patterns are considered satisfactory by the expert. All experiments confirm the importance of the user-defined thresholds, controlling the quality of the initial population. Due to the variability of the data, these threshold parameters are adjusted for each new experiment.

The coarse tuning of the parameters can be achieved based on the desired proportion of admissible individuals in the initial population. However, the fine-tuning of the parameters could not be automatized up to now, and it still requires running *4D-Miner* a few times. For this reason, the control of the computational cost through the population size and number of generations is one of the key features of the algorithm.

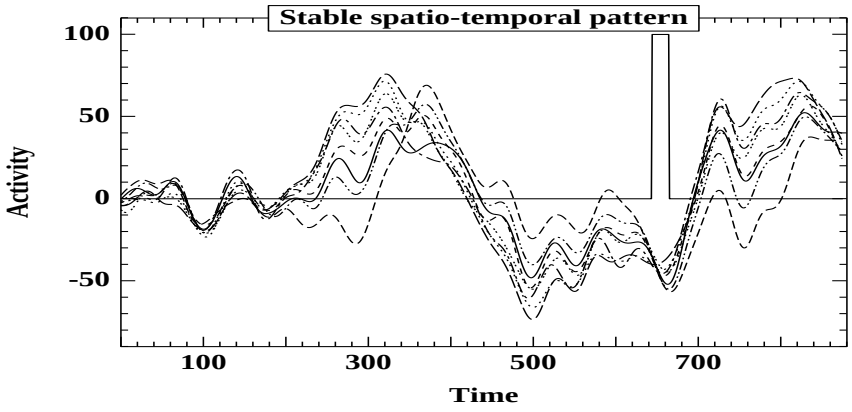


FIG. 4 – Another stable spatio-temporal pattern involving 9 measure points within interval [644,664], alignment .3963

5.2 Multi-objective multi-modal optimization

The good scalability of *4D-Miner* is illustrated in Fig. 5. The empirical complexity of the approach is insensitive to the number of time steps T and linear in the number N of measure points. This computational robustness confirms the analysis (section 3.1), the evaluation of each pattern has linear complexity in N. On-going work is concerned with exploiting additional data structures inspired from iciteYu in order to decrease the complexity in N.

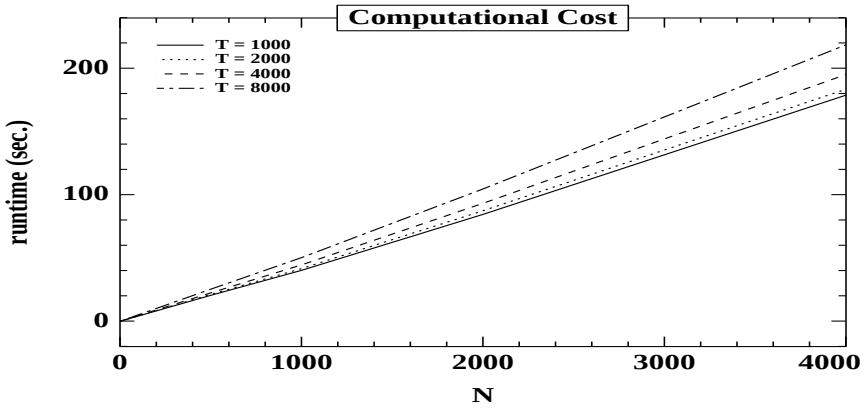


FIG. 5 – Computational cost *vs* N, for $T = 1000, 2000, 4000, 8000$. (in seconds, on PC 2.4 GHz)

Table 1 reports the recall achieved by *4D-Miner* over the range of experiments, averaged over 21 independent runs, together with the standard deviation.

N	T			
	1,000	2,000	4,000	8,000
500	98 ± 5	93 ± 9	92 ± 7	79 ± 16
1000	96 ± 6	96 ± 6	82 ± 14	67 ± 12
2000	96 ± 5	87 ± 12	72 ± 14	49 ± 15
4000	89 ± 10	81 ± 13	56 ± 14	32 ± 16

TAB. 1 – Recall achieved by *4D-Miner* vs N and T (average percentage and standard deviation over 21 runs)

On-line performances are illustrated on Fig 6. These results confirm the robustness of the proposed approach : a graceful degradation of the recall is observed as T and N increase. It must be noted that STPs occupy a negligible fraction of the spatio-temporal domain (circa 10^{-4} for $T = 8000$, $N = 4000$).

The average precision is low, ranging from 12 to 20% (results omitted due to space limitations). However, post-pruning can be used to sort the wheat from the chaff in the final archive, and increase the precision up to 50% without decreasing the recall ; the

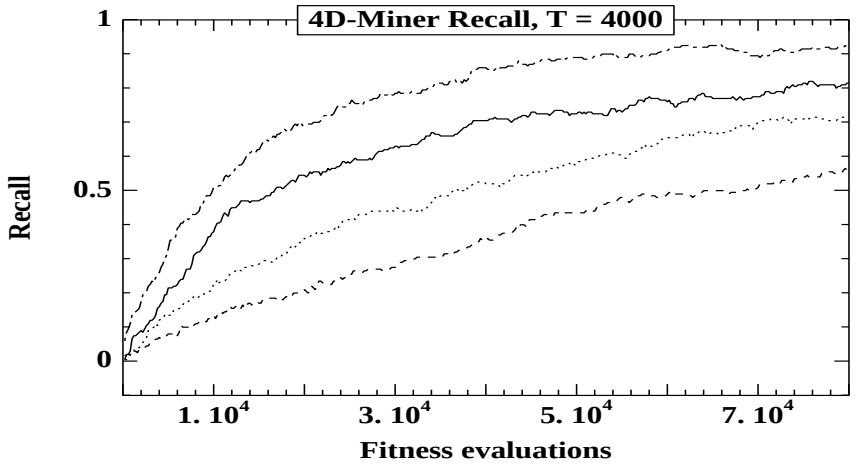


FIG. 6 – Recall achieved by *4D-Miner* vs the number of fitness evaluations. ($T = 4000$, $N = 500,\ 1000,\ 2000,\ 4000$, average over 21 runs).

post-pruning straightforwardly removes the STPs with small spatial or temporal amplitudes.
Quite different effects are obtained when the archive is pruned along the search (e.g. by increasing the thresholds on the minimal spatial and temporal amplitudes), which decreases the overall performances by an order of magnitude ; interestingly, similar effects are observed in constrained evolutionary optimization when the fraction of admissible solutions is very low.

A final remark is that the performances heavily depend upon the user-supplied thresholds (section 3.1), controlling the diversity and effective size of the population. Indeed, a parametric model of the dataset would enable automatic adjustment of these parameters. It might also be viewed as a advantage of *4D-Miner* that it does not require any prior model of the data, and that inexpensive preliminary runs can be used to adjust the needed parameters.

6 Relevant work

This brief review does not claim exhaustiveness ; the interested reader is referred to iciteSpatial,TEMPORAL for comprehensive surveys. Spatio-temporal data mining applications (e.g., remote sensing, environmental studies, medical imaging) are known to be computationally heavy ; as most standard statistical methods do not scale up properly, new techniques have been developed, including randomized variants of statistical algorithms.

Many developments are targeted at efficient access primitives and/or complex data structures (see, e.g., iciteYu) ; another line of research is based on visual and interactive data mining (see, e.g., iciteKeim), exploiting the unrivaled capacities of human eyes for

spotting regularities in 2D-data.

Spatio-temporal data-mining mostly focuses on clustering, outlier detection, denoising, and trend analysis. For instance, icitePadraic used EM algorithms for non-parametric characterization of functional data (e.g. cyclone trajectories), with special care regarding the invariance of the models with respect to temporal translations. The main limitation of such non-parametric models, including Markov Random Fields, is their computational complexity, sidestepped by using randomized search for model estimates.

7 Discussion and Perspectives

This paper has proposed a stochastic approach for mining stable spatio-temporal patterns. Indeed, a very simple alternative would be to discretize the spatio-temporal domain and compute the correlation of the signals in each cell of the discretization grid. However, it is believed that the proposed approach presents several advantages compared to the brute force, discretization-based, alternative.

First of all, *4D-Miner* is a fast and frugal algorithm ; its good performances and scalability have been successfully demonstrated on medium sized artificial datasets.

Second, data mining applications specifically involve two key steps, exemplified in this paper : i) understanding the expert's goals and requirements ; ii) tuning the parameters involved in the specifications. With regard to both steps, the ability of working under bounded resources is a very significant advantage ; any-time algorithms allow the user to check whether the process can deliver useful results, at a low cost.

Further research is concerned with extending *4D-Miner* in a supervised learning perspective (finding the STPs that are complete – active in several persons undergoing the same experiment – and correct – not active in the control experiment). The challenge is to directly handle the additional constraints of completeness and correction in the multi-objective multi-modal optimization framework presented here.

Références

T. Bäck. *Evolutionary Algorithms in theory and practice*. New-York :Oxford University Press, 1995.

S. Bhattacharyya, Evolutionary algorithms in data mining : multi-objective performance modeling for direct marketing. Proceedings of KDD, pp 465-473, 2000.

D. Chudova, S. Gaffney, E. Mjolsness, and P. Smyth. Translation-invariant mixture models for curve clustering. In *KDD'03*, pages 79–88. ACM Press, 2003.

D. Corne, J. D. Knowles, and M. J. Oates. The Pareto envelope-based selection algorithm for multi-objective optimisation. In *PPSN VI*, pages 839–848. Springer Verlag, 2000.

J.M. Daida. Challenges with verification, repeatability, and meaningful comparison in Genetic Programming. In *GECCO'99*, pages 1069–1076. Morgan Kaufmann, 1999.

K. Deb. *Multi-Objective Optimization Using Evolutionary Algorithms*. John Wiley, 2001.

D. Francisci, M. Collard, Multi-Criteria Evaluation of Interesting Dependencies according to a Data Mining Approach, Proceedings of the 2003 Congress on Evolutionary Computation (CEC'2003), Vol. 3, pp. 1568-1574, IEEE Press, Canberra, Australia, 2003.

A. Ghosh and B. Nath. Multi-objective rule mining using genetic algorithms. *Inf. Sci.*, 163(1-3) :123–133, 2004.

D. Goldberg. *The Design of Innovation : Lessons from and for Genetic and Evolutionary Algorithms*. MIT Press, 2002.

A. Hyvarinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. Wiley New York, 2001.

M. Hämäläinen, R. Hari, R. Ilmoniemi, J. Knuutila, and O. V. Lounasmaa. Magnetoencephalography : theory, instrumentation, and applications to noninvasive studies of the working human brain. *Rev. Mod. Phys*, 65 :413–497, 1993.

B. de la Iglesia, M.S. Philpott, A.J. Bagnall, V.J. Rayward-Smith, Data Mining Using Multi-Objective Evolutionary Algorithms, Proceedings of the 2003 Congress on Evolutionary Computation (CEC'2003), Vol. 3, pp. 1552-1559, IEEE Press, Canberra, Australia, 2003.

D. A. Keim, J. Schneidewind, and M. Sips. Circleview : a new approach for visualizing time-related multidimensional data sets. In *Proc. of Advanced Visual Interfaces*, pages 179–182. ACM Press, 2004.

M. Laumanns, L. Thiele, K. Deb, and E. Zitsler. Combining convergence and diversity in evolutionary multi-objective optimization. *Evolutionary Computation*, 10(3) :263–282, 2002.

J.-P. Li, M. E. Balazs, G. T. Parks, and P. J. Clarkson. A species conserving genetic algorithm for multimodal function optimization. *Evolutionary Computation*, 10(3) :207–234, 2002.

D. Pantazis, T. E. Nichols, S. Baillet, and R.M. Leahy. A comparison of random field theory and permutation methods for the statistical analysis of meg data. *Neuroimage*, 2005.

J.F. Roddick and M. Spiliopoulou. A survey of temporal knowledge discovery paradigms and methods. *IEEE Trans. on Knowledge and Data Engineering*, 14(4) :750–767, 2002.

C. Saunders, D. R. Hardoon, J. Shawe-Taylor, and G. Widmer. Using string kernels to identify famous performers from their playing style. In *ECML04*, pages 384–395. Springer Verlag, 2004.

S. Shekhar, P. Zhang, Y. Huang, and R. R. Vatsavai. Spatial data mining. In H. Kargupta and A. Joshi, eds, *Data Mining : Next Generation Challenges and Future Directions*. AAAI/MIT Press, 2003.

K.L. Wu, S.K. Chen, and P.S. Yu. Interval query indexing for efficient stream processing. In *13^{th} ACM Conf. on Information and Knowledge Management*, pages 88–97, 2004.

S. Zilberstein. Resource-bounded reasoning in intelligent systems. *Computing Surveys*, 28(4), 1996.

Clustering gene expression series with prior knowledge

Laurent Bréhélin

[1] Unité de Biochimie & Physiologie Moléculaire des Plantes, 2, Place Viala, 34060 Montpellier Cedex 1, France

[2] Laboratoire d'Informatique, Robotique et Microélectronique de Montpellier, 161, rue Ada, 34392 Montpellier Cedex 5, France
brehelin@lirmm.fr

Résumé : Les biopuces permettent de mesurer le niveau d'expression de milliers de gènes au cours du temps. Ces séries d'expression de gènes constituent un matériel unique pour la compréhension des mécanismes de régulation cellulaire. Récemment, des algorithmes de classification de gènes prenant en compte les dépendances entre les temps ont été proposées. Dans cet article, nous étudions comment étendre ce genre d'approche en intégrant des connaissances *a priori* approximatives de certains profils temporels afin d'améliorer la détection des classes de gènes les plus intéressantes.

Nous proposons une approche bayésienne de ce problème. Un modèle de mélange est utilisé pour décrire et classer les données. Les paramètres de ce modèle sont contraints par une distribution *a priori* définie grâce à un nouveau type de modèles —proche des modèles de Markov cachés— qui exprime les connaissances *a priori* des profils les plus intéressants. Lorsque ces connaissances ne sont pas disponibles, la méthode permet simplement de traiter les dépendances temporelles d'une manière très naturelle. Un algorithme EM estime les paramètres du modèle de mélange en maximisant sa probabilité *a posteriori*. On observe expérimentalement que cette approche est d'utilisation aisée, et que l'incorporation de connaissances *a priori* permet de mettre en évidence les principales classes intéressantes, même lorsque celles-ci sont très réduites au regard de tous les autres gènes.

Informations supplémentaires :
http://www.lirmm.fr/~brehelin/CAp05.pdf

1 Introduction

Technological advances such as microarrays allow us to simultaneously measure the level of expression of thousands of genes in a given tissue at a given moment. These measurements can be repeated on different tissues, different biological organisms, or at different times during the life of the same organism to constitute a collection of gene

expression measurements. These collections are a unique material for understanding various cellular regulation mechanisms. These collections are either ordered or non-ordered. A non-ordered collection may, for example, be a set of measurements on different patients with a given form of cancer (Alon *et al.*, 1999), or on plants growing on different substrates. Ordered collections generally consist of series of gene expressions measured over a time course —for example along the cell cycle (Spellman *et al.*, 1998). The order is generally defined by time, but it may also be induced by other numerical features. In (Hertzberg *et al.*, 2001) for example, the expression levels are measured at different depths of the stem of poplar trees. In other studies, measurements are obtained on cells exposed to increasing concentrations of a given factor (light, chemical product, etc). In the following, such a series of gene expression measurements is called an *expression series*, and we speak about the different *time points* of the series, even if the order is not temporal.

One common problem of gene expression data analysis is the identification of co-regulated genes. This problem naturally turns into a gene clustering problem. Until recently, expression series have been analyzed with methods that do not take the time dependences into account. Such methods include hierarchical clustering with Euclidean distance (Eisen *et al.*, 1998), k-means approaches (Lloyd, 1982; Herwig *et al.*, 1999) and the Self Organizing Maps (Kohonen, 1997; Tamayo *et al.*, 1999). Since these methods are unable to explicitly deal with the data order, permuting two or more time points in all series does not change the clustering result.

A few methods specially adapted to expression series have recently been proposed. These methods involve probabilistic modeling of the data. For example, (Ramoni *et al.*, 2002) use autoregressive models of order p. (Bar-Joseph *et al.*, 2003) use cubic splines with a probabilistic component to model the classes, while (Schliep *et al.*, 2003) model each class of gene with Hidden Markov Models (HMMs) (Rabiner, 1989).

Our aim here is to investigate how to explicitly use *rough* prior knowledge about the general shape of interesting classes. By *general shape*, we mean elementary and potentially incomplete information about the evolution of the mean expression level of the classes over time. This can, for example, be knowledge like: *"Classes with increasing expression level", "Classes with bell curve shapes", "Classes with high expression level in the beginning of the series"*, etc. Of course we do not know the profile of *all* the gene classes, but sometimes we are more concerned with one or more classes. For example, in the study of (Spellman *et al.*, 1998) on the Yeast cell cycle, the authors are interested in finding the cycle-regulated genes, and thus look for sinusoidal shape classes. In a similar way, we sometimes search for genes which tend to be quickly over- (or under-) expressed at the beginning of the series —in response to a given treatment, for example. Our idea is that incorporating such (even rough) knowledge can improve the clustering result, especially when the classes of interest are very sparse with regard to all the other genes.

The approach we propose here tackles this problem. When information about one or several class shapes are available, these are directly integrated into the model, thus favoring classes with the desired profiles, and putting the other genes in separate classes. On the other hand, when no a priori information is available, the method allows a classical clustering of the series. This is done by explicitly dealing with the temporal nature

of the data, in a very intuitive way and without any assumption about a predetermined analytical form which can be difficult to estimate when the number of time points is low.

We use a Bayesian approach for this purpose. The approach involves two types of models. The first one is a probabilistic mixture model used to describe and classify the expression series. Parameters of this model are unknown and have to be estimated for the clustering. A second model, close to the HMMs and called *HPM* —for *Hidden Phase Model*—, is used to express our a priori knowledge (or simply the temporal feature of the data). We define two types of HPMs which can be used according to the situation: probabilistic and non-probabilistic HPMs. These models are completely specified by the user, and their parameters do not have to be estimated. They are used to define a prior probability distribution over the parameters of the mixture model. These parameters are estimated by maximizing the posterior probability of the model through an EM algorithm (Dempster *et al.*, 1977).

The next section presents our method, the mixture model, the two types of HPMs and the learning algorithm. In Section 3 we evaluate and experiment our method on two datasets. We conclude and propose future work directions in Section 4.

2 Method

2.1 Principle

Let $\mathcal{X}$ be a set of N expression series of length T. We assume that the data arise from a mixture model (McLachlan & Krishnan, 2000) with C components. We denote π_c as the prior probability of component c, and we have $\sum_{c=1}^{C} \pi_c = 1$. We assume that conditionally to component c, expression values at each time $t \in [1, T]$ are independent and follow a Gaussian distribution of mean μ_{ct} and variance σ_{ct}^2. The shape of component c is defined by the sequence of means $\mu_{c1} \dots \mu_{cT}$. We then have a probabilistic model of parameters $\Theta = (\pi_1, \dots, \pi_C, \theta_1, \dots, \theta_C)$ with $\theta_c = (\mu_{c1}, \dots, \mu_{cT}, \sigma_{c1}^2, \dots, \sigma_{cT}^2)$. The probability of an expression series $X = x_1 \dots x_T$ in this model is

$$P(X|\Theta) = \sum_{c=1}^{C} \pi_c \prod_{t=1}^{T} P(x_t|\mu_{ct}, \sigma_{ct}^2),$$

with $P(x_t|\mu_{ct}, \sigma_{ct}^2) = \mathcal{N}(x_t; \mu_{ct}, \sigma_{ct}^2)$. Under the assumption that series of $\mathcal{X}$ are independent, the likelihood of Θ is given by

$$L(\Theta|\mathcal{X}) = P(\mathcal{X}|\Theta) = \prod_{X \in \mathcal{X}} P(X|\Theta). \tag{1}$$

In a clustering task, the standard approach to classify a set of expression series $\mathcal{X}$ involves estimating parameters Θ that maximize Formula (1) (Maximum Likelihood Principle), and then assigning the most probable component c_{MAP} (*MAP* stands for *maximum a posteriori*) to each series $X \in \mathcal{X}$:

$$c_{\text{MAP}} = \underset{c=1\dots C}{\operatorname{argmax}} P(c|X, \Theta) = \underset{c=1\dots C}{\operatorname{argmax}} \pi_c P(X|c, \Theta) \tag{2}$$

Note that finding parameters Θ that maximize (1) is a difficult task. However, approximate solutions can be inferred with EM algorithms (Dempster *et al.*, 1977).

The above mixture model does not explicitly take into account the potential dependences between times, nor any prior knowledge about the profile of the most interesting classes. Our aim is to constraint one or some components to follow a given profile, while leaving the other components free of constraints so that they can "collect" the expression series that do not have the desired profile. For example, if we are looking for classes with bell curves, we would build a 10 component model, with 5 bell-constrained and 5 unconstrained components. We thus propose to use a Bayesian approach, which introduces knowledge by way of a prior distribution of Θ —see for example (Duda *et al.*, 2001) for a general introduction to Bayesian theory. Simply speaking, our idea is to define a prior distribution $P(\Theta)$ which is merely the product of the prior probability of the sequences of means $\mu_{c1} \dots \mu_{cT}$ associated with each component. Moreover, we want the prior probability of a given mean sequence for component c as follows: (i) the more the sequence agrees with the constraints associated with c, the higher its prior probability; (ii) sequences that disagree with the constraints have probability zero.

With a prior, we can write the posterior probability of Θ as

$$P(\Theta|\mathcal{X}) = \frac{P(\mathcal{X}|\Theta)P(\Theta)}{P(X)} \propto P(\mathcal{X}|\Theta)P(\Theta). \tag{3}$$

In this Bayesian framework, parameters Θ are estimated by maximizing the posterior probability —Equation (3)— instead of the likelihood —Expression (1). However, maximizing the posterior probability is generally more difficult than maximizing the likelihood. For example, the classical re-estimation formulae of the EM algorithm do not directly apply and, depending on the form of the chosen prior distribution, it may be hard to perform the task in reasonable time.

In our case, we first discretize the space of the means μ_{ct} in order to be able to introduce various bits of knowledge and constraints about the profiles, as well as to efficiently estimate the parameters of the model. Since we know the maximal and minimal expression values taken by the series in $\mathcal{X}$ (say x_{max} and x_{min}), we already know an upper and lower bound of the space of the means. Now we discretize this space in M equidistant steps, so that the lower and higher steps are equal to x_{min} and x_{max}, respectively. Of course M is chosen to be sufficiently large (e.g. $M = 30$) to allow accurate representation of the data. Steps are named by their number, so M is the highest step. In this discretized mean space, our probabilistic model is redefined as $\Theta = (\pi_1, \dots, \pi_C, \theta_1, \dots, \theta_C)$ with $\theta_c = (l_{c1}, \dots, l_{cT}, \sigma^2_{c1}, \dots, \sigma^2_{cT})$, with $l_{ct} \in \{1, \dots, M\}$. We denote $m : \{1, \dots, M\} \rightarrow [x_{\text{min}}, x_{\text{max}}]$ as the map function that associates step l with its expression level. The probability of an expression series $X \in \mathcal{X}$ is rewritten as

$$P(X|\Theta) = \sum_{c=1}^{C} \pi_c \prod_{t=1}^{T} P(x_t|l_{ct}, \sigma^2_{ct}),$$

with $P(x_t|l_{ct}, \sigma^2_{ct}) = \mathcal{N}(x_t; m(l_{ct}), \sigma^2_{ct})$ that follows a Gaussian distribution of mean equal to the level of expression associated with step l_{ct}, and variance σ^2_{ct}. In the following, the step sequence $l_{c1} \dots l_{cT}$ associated with class c —and which defines its

shape— is denoted as L_c. Note finally that the discretization only involves the means of the model, and not the space of the expression levels of the data. These, as well as the model variances σ_{ct}^2, remain in a continuous space.

In the next section, we show how to define the prior distribution of parameters Θ. Section 2.3 details the EM algorithm used to estimate these parameters in maximizing Expression (3).

2.2 Defining the prior distribution

Fist we define a new type of model called *Hidden Phase Models* (or *HPMs*), close to models like HMMs and finite automata. These HPMs are used to express the desired profiles of the components, and each component c is then associated with a given HPM H_c. We define two types of HPMs: probabilistic and non-probabilistic HPMs. We next show how to derive the prior distribution of Θ from the HPMs.

2.2.1 Hidden Phase Models

The general assumption behind HPMs is that the genes of a given component pass through *phases* or *biological states* over the time. This means that, for a given component, we assume that some ranges of consecutive times actually correspond to the same biological state. These phases are hidden, but they affect the mean expression level evolution of the component. For example, some phases induce an increase in the mean level expression level while others tend to decrease or stabilize the level. In the same manner, the increase (or decrease) can be high for some phases and low for others, etc.

A (non-probabilistic) HPM is defined by a quadruplet $(\mathcal{S}, \delta, \epsilon, \tau)$, where

- $\mathcal{S}$ is a set of states representing the different phases; $\mathcal{S}$ contains two special states, $start$ and end, which are used to initiate and conclude a sequence, respectively.
- $\delta : \mathcal{S} \times \mathcal{S} \mapsto \{0, 1\}$ is a function describing the authorized transitions between states. We denote $Out(s)$ as the set of states that can be reached from s. Note that if $s \in Out(s)$ then there is a loop on state s.
- ϵ is a function that associates each state $s \in \mathcal{S}$ with an interval of integers defining the minimal and maximal differences of steps that can be observed between times t and $t-1$ when genes are in state s at time t. For example, if $\epsilon(s) = [1, 3]$, this means that if the genes of the component are in phase s at time t then the step difference $(l_t - l_{t-1})$ is between 1 and 3 (so phase s increases the expression level).
- τ is a function that associates each state $s \in \mathcal{S}$ with the interval of time the state can be reached. For example, if $\tau(s) = [3, 5]$ then the genes can be in state s between times 3 and 5 included.

An HPM example is depicted in Figure 1.

Now we can see how to express our prior knowledge with an HPM. Actually an HPM defines a set of *compatible* step sequences. We say that a step sequence $L = l_1 \dots l_T$ is

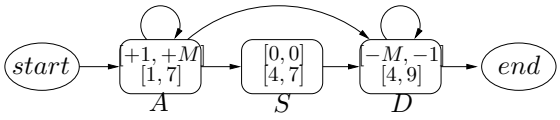


Figure 1: An HPM for clustering 9-time expression series. In each state, upper and lower intervals represent the step-difference and time intervals associated with the state, respectively. This HPM induces bell curve shapes.

compatible with an HPM H if there is a state sequence $s_0 \dots s_{T+1}$ —with $s_0 = start$ and $s_{T+1} = end$— in H, which is compatible with L. And we say that a state sequence $s_0 \dots s_{T+1}$ is compatible with L *iff* for each time $1 \leq t \leq T$ we have:

1. t included in the time interval $\tau(s_t)$;
2. $\forall t \geq 2$, $(l_t - l_{t-1})$ included in $\epsilon(s_t)$; for $t = 1$, as we do not know l_0, the genes can be in any phase so s_1 can be any state.

Considering the step sequence on the top of Figure 2, a compatible phase sequence in the HPM of Figure 1 is, for example, $start - A - A - A - A - S - D - D - D - D - end$. For the step sequence on the right, there is no compatible phase sequence in this HPM. In brief, building an HPM involves designing an HPM such that the compatible sequences have the desired profile. For example, the HPM of Figure 1 is well suited for the discovery of bell curve classes.

2.2.2 Probabilistic HPMs

Non probabilistic HPMs can be used to express strong constraints. They are generally sufficient to express knowledge about simple or well defined profiles. For more complex knowledge, or when we do not have any information about profiles and just want to express the fact that we are dealing with series data, these models can be unsuitable. Then probabilistic HPMs can be more suitable.

A probabilistic HPM is defined by a quintuplet $(\mathcal{S}, \delta, \epsilon, \tau, w)$, where $\mathcal{S}$, δ, ϵ, and τ are the same as for non-probabilistic HPMs, and $w : \mathcal{S} \times \mathcal{S} \mapsto \mathbb{R}^+$ is a function associating a weight with each authorized transition. These weights are used to compute the transition probabilities from state to state. Due to the time constraints associated with the states by way of the τ function, transition probabilities are time dependent, so we cannot simply label transitions with a probability as is done for classical HMMs. In contrast, the probability, denoted as $P(s|s', t)$, to reach state s from state s' at time t is computed as follows:

$$P(s|s', t) = \begin{cases} 0 \text{ if } t \notin \tau(s); \\ w(s) / \left(\sum_{s'' \in Out(s') \mid t \in \tau(s'')} w(s'') \right) \text{ else.} \end{cases} \quad (4)$$

One example of probabilistic HPM is depicted in Figure 2.

Probabilistic HPMs also define compatible step sequences. Moreover, all compatible sequences do not have the same probability. Let H be a probabilistic HPM and $S =$

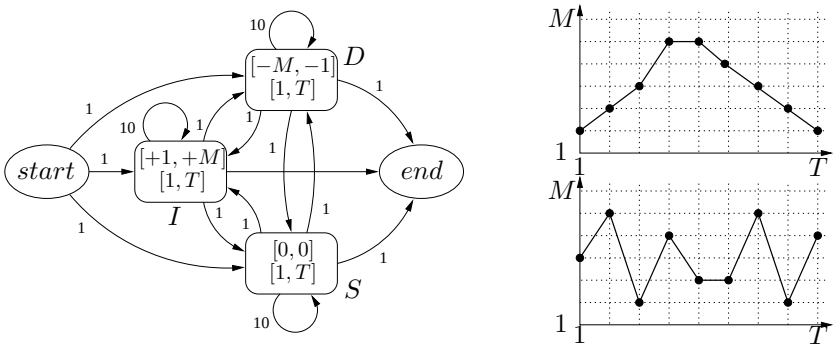


Figure 2: Left, a probabilistic HPM for clustering expression series without prior knowledge about the form of the profiles. Right, two examples of step sequences.

$s_0, s_1 \dots s_T, s_{T+1}$ a state sequence in this HPM. The probability of this sequence given H is defined by

$$P(S|H) = \prod_{t=1}^{T+1} P(s_t|s_{t-1}, t). \tag{5}$$

This model enables us to introduce more knowledge about the desired classes. For example, when we do not have any information about interesting profiles, the only thing we know is that we have to classify expression series. This means that we are seeking relatively "regular" profiles, in contrast to chaotic spiky profiles as that depicted on the bottom of Figure 2. This knowledge can be easily expressed with the probabilistic three-states HPM of Figure 2: one state (I) induces increasing steps, one (D) induces a decrease, and the last (S) induces stability. Moreover, it is assumed that, at each time, the probability of staying in the same state is higher than the probability of departure from it (weights on loops are higher than on other transitions). This HPM is compatible with any step sequence of length 9. However all sequences do not have the same probability, and spiky sequences involving many state changes are not favored.

Note that given a step sequence L, there are potentially many state sequences compatible with L. In reference to the HMM literature, the sequence of phases compatible with L which has the highest probability is called the *Viterbi sequence* of L (Rabiner, 1989), and is denoted as $V^L = v_0^L \dots v_{T+1}^L$. For example, the Viterbi sequences of the two step sequences of Figure 2 in the HPM of Figure 2, are $start - I - I - I - I - S - D - D - D - D - end$ and $start - I - I - D - I - D - S - I - D - I - end$, respectively.

2.2.3 Defining prior with HPMs

First we assume that prior probabilities of parameters π_c, L_c and σ_{ct}^2 are independent, as well as the C sets of parameters L_c and $(\sigma_{c1}^2, \dots, \sigma_{cT}^2)$, i.e., the probability distribution

can be written as:

$$P(\Theta) = P(\pi_1, \ldots, \pi_C) \prod_{c=1}^{C} P(L_c) \prod_{c=1}^{C} P(\sigma_{c1}^2, \ldots, \sigma_{cT}^2).$$

Next we assume that distributions $P(\pi_1, \ldots, \pi_C)$ and $P(\sigma_{c1}^2, \ldots, \sigma_{cT}^2)$ are uninformative and that probabilities $P(L_c)$ are the only ones that express our knowledge.

Let c be a component and H_c a non probabilistic HPM associated with this class. A prior distribution of parameters L_c can be defined with H_c by assuming that the step sequences incompatible with H_c have probability zero while compatible sequences have all the same probability, i.e.,

$$P(L|H_c) = \begin{cases} 0 \text{ if } L \text{ is incompatible with } H_c; \\ K_c \text{ else,} \end{cases} \quad (6)$$

with K_c such that $\sum_{L \in \mathcal{L}_T} P(L) = 1$, with $\mathcal{L}_T$ being the set of length T sequences.

For probabilistic HPM, we want the prior probability of a step sequence L to be proportional to the Viterbi sequence of L in H_c. Then, we set

$$P(L|H_c) = \begin{cases} 0 \text{ if } L \text{ is incompatible with } H_c; \\ K'_c \cdot P(V^L|H_c) \text{ else,} \end{cases} \quad (7)$$

with K'_c such that $\sum_{L \in \mathcal{L}_T} P(L) = 1$. For example, for the HPM of Figure 2, the prior probabilities of the two step sequences are proportional to $1/3 \cdot .7 \cdot .7 \cdot .7 \cdot .1 \cdot .1 \cdot .7 \cdot .7 \cdot .7 \cdot .1 \sim 3.9 \cdot 10^{-5}$ and $1/3 \cdot .7 \cdot .1 \cdot .1 \cdot .1 \cdot .1 \cdot .1 \cdot .1 \cdot .1 \cdot .1 \sim 2.3 \cdot 10^{-10}$, respectively. The spiky sequence is then less likely than the other one, which agrees with our prior intuition.

A prior distribution of the step sequences of length T can then be defined with a probabilistic or a non-probabilistic HPM. In practice, one or more components can be associated with a given HPM (e.g. that of Figure 1), and the other ones with a less informative HPM like that of Figure 2. We then have

$$P(\Theta) \propto \prod_{c=1}^{C} P(L_c|H_c). \quad (8)$$

2.3 Learning

Here we briefly describe the learning algorithm used to estimate parameters Θ of the mixture model. A more detailed version can be found in the supplementary information material[1]. It is an EM algorithm that searches for parameters that maximize Expression (3). We only give the algorithm used for probabilistic HPMs, since that for non-probabilistic ones can be easily adapted.

Let us first define the *complete-data* likelihood. Likelihood of Expression (1) is actually the incomplete-data likelihood, since the real components of series $X \in \mathcal{X}$ are unknown. Under the assumption that this set of components $\mathcal{C} = \{c_X \in \{1, \ldots, C\}, \forall X \in$

[1] `http://www.lirmm.fr/~brehelin/CAp05.pdf`

$\mathcal{X}\}$ is known, the complete-data likelihood can be written as

$$L(\Theta|\mathcal{X},\mathcal{C}) = P(\mathcal{X},\mathcal{C}|\Theta) = \prod_{X\in\mathcal{X}} \pi_{c_X} \prod_{t=1}^{T} P(x_t; l_{c_X t}, \sigma^2_{c_X t}).$$

The EM algorithm is an iterative algorithm that starts from an initial set of parameters $\Theta^{(0)}$, and iteratively reestimates the parameters at each step of the process. Let $Q(\Theta,\Theta^{(i)})$ denote the expectation, on the space of the hidden variables $\mathcal{C}$, of the logarithm of the complete-data likelihood, given the observed data $\mathcal{X}$ and parameters $\Theta^{(i)}$ at step i:

$$Q(\Theta,\Theta^{(i)}) \quad = \quad \sum_{\mathcal{C}\in\mathbf{C}} \log P(\mathcal{X},\mathcal{C}|\Theta) P(\mathcal{C}|\mathcal{X},\Theta^{(i)}),$$

with **C** being the space of values $\mathcal{C}$ can take. From (Dempster *et al.*, 1977), one can maximize Expression (3) by searching at each step of the algorithm for parameters π^*_c, L^*_c and ${\sigma^2_{ct}}^*$ that maximize the quantity

$$Q(\Theta,\Theta^{(i)}) + \log P(\Theta). \tag{9}$$

Since $P(\Theta)$ is not related to the parameters π_c, after some calculus, an expression can be derived for π^*_c that maximizes Expression (9):

$$\pi^*_c = \frac{1}{|\mathcal{X}|} \sum_{X\in\mathcal{X}} P(c|X,\Theta^{(i)}). \tag{10}$$

Now, due to our independence assumptions, one can estimate the L_c and σ^2_{ct} that maximize Expression (9) for each component c independently. As for parameters π_c, σ^2_{ct} are not involved in the expression of $P(\Theta)$. Moreover, since the σ^2_{ct} associated with time t is independent of all the other times, the expression of ${\sigma^2_{ct}}^*$ that maximizes (9) depends solely on the step l^*_{ct} in L^*_c:

$${\sigma^2_{ct}}^* = \frac{\sum_{X\in\mathcal{X}} (x_t - m(l^*_{ct}))^2 P(c|X,\Theta^{(i)})}{\sum_{X\in\mathcal{X}} P(c|X,\Theta^{(i)})}. \tag{11}$$

For L_c the situation is quite different since it is involved in the expression of $P(\Theta)$. The L_c that maximizes Expression (9) depends both on the data and on its Viterbi path in H_c and hence the different steps l^*_{ct} of L^*_c cannot be estimated independently. However, the step space is of finite size, so the space of the step sequences of length T is also finite. One way to compute the new L_c would be to enumerate all possible step sequences and then select the one that maximizes Expression (9). However, as the total number of length T sequences is equal to M^T, enumerating them all is clearly not suitable. Instead, we use a dynamic programming approach that iteratively computes the best sequence without enumerating all the solutions. Briefly, for each step l and each time t, we compute iteratively, from $t = 1$ to T, the best sequence —with regard to Expression(9)— that ends on step l at time t. At each iteration and for each step l,

this best sequence is computed using the results of the previous iteration, and at the end of the process the best sequence L_c^* has thus been computed in polynomial time.

The learning algorithm is depicted in Algorithm 1. When no better solution is available, the initial parameter values can be set randomly. Thanks to the EM properties, the posterior probability $P(\Theta|\mathcal{X})$ —and hence $P(\mathcal{X}|\Theta)P(\Theta)$— increases at each loop of the algorithm, until a local optimum is reach. Then it continues to increase but to a much lesser extent. A practical way to detect the convergence is to check the increase at each loop and to stop the algorithm when this value goes under a given boundary.

Algorithm 1: Learning algorithm

Set parameters to initial values
repeat
  for $c = 1$ to C **do**
    compute π_c^* with Formula (10)
    Find the optimal step sequence $L_c^* = l_{c1}^* \dots l_{cT}^*$ with the dynamic programming algorithm
    foreach time t **do**
      compute ${\sigma_{ct}^2}^*$ from l_{ct}^* with Formula (11)
  Compute $P(\mathcal{X}|\Theta)P(\Theta)$
until convergence

The total time complexity of the learning algorithm is $O(BCTM^2R^2N)$ —see supplementary information for details—, with B, C, T, M, R and N the maximal number of loops of the EM algorithm, the number of components of the mixture model, the number of time points of the data, the size of the step space, the maximal number of states of the HPMs, and the number of expression series to classify, respectively. In practice, N is potentially high (some thousands), T and R are relatively low (ten or less), M is around thirty, and less than one hundred loops are generally sufficient to ensure convergence. For the experiments in the next section for example, computing times on a 2 GHz Pentium 4, range from 20 seconds to 3 minutes according to the dataset, the type of HPMs and the number of components.

3 Evaluation and experiments

When applied to a given dataset, our method provides a mixture model, i.e. a set of profiles (the step sequences) with the variances σ_{ct}^2 associated with each time and the prior probabilities π_c. Moreover, it provides the probability membership of each gene for each class, and groups the genes according to their most probable components into clusters. Features of the mixture model are useful to access the pertinence of the clusters. Indeed, sometimes one constrained component c may fail to collect "good" genes. This occurs when no gene agrees with H_c, or when the desired genes are collected by another component with similar constraints. Then, two different situations can arise.

First, the component does not collect any gene and its probability $\pi_c = 0$. Second, the component collects some series, but these do not have the desired profile: the measures x_t are far from $m(l_{ct})$ at one or several time points (there is a gap between the series and L_c) and the variance is high at these points. This situation can be merely detected by visual inspection, or by checking if the value of σ^2_{ct} is not higher than a given threshold.

3.1 Recovering a known class of genes

In order to quantify the advantages of using prior knowledge to recover a particular class of genes, we first conducted some experiments on a dataset made up of the original Fibroblast dataset (see Section 3.3 for more details), along with some additional synthetic series that form a new artificial class. Briefly, we use a probabilistic model involving two Gaussian distributions to generate the expression levels of the artificial expression series: one Gaussian distribution is used to independently generate the gene expression levels of the first three times, while the other is used for the last nine times of the series. The mean of the first one is higher than the second, so the shape of the artificial class looks like a descending step. Figure 3 shows an example of synthetic series generated with this model. We conducted several experiments to recover the synthetic class among all other series, with the proportion of synthetic data ranging from 2% to 16% of the total data.

We use two quantities to measure the ability to recover the artificial class in the final clustering: *Recall* is the highest proportion of this class that can be found in a single cluster —so a recall of 100% is achieved when all the artificial series are in the same cluster—, and *precision* represents the proportion of artificial series in this cluster —so a precision of 100% indicates that all the series in the cluster containing most artificial series are actually artificial. For each proportion of synthetic data, we run a clustering of 11 components with two different methods. The first one does not use any prior knowledge about the class of interest, i.e., its components are completely unconstrained —this method can be viewed as a kind of k-means clustering. The second method makes use of the HPM of Figure 3 to constrain the first class, leaving the 10 others unconstrained. The experiments were repeated 100 times for each proportion of synthetic data and the results are reported in Figure 4.

Both methods achieve quite good recall, even when the proportion of the class of interest is low. Using prior knowledge gives only slightly better results. Concerning the precision, however, there is a clear difference between the two methods, and we can see that the lower the proportion of interesting class, the higher the benefit of our method. When the proportion is 2%, for example, the precision achieved with no prior knowledge is only about 21% —vs. 65% when using prior knowledge—, so the interesting series are lost among many other series, leading to a class that does not show the desired profile.

3.2 Number of components

Next we investigated the sensitivity of the method to the number of components. Determining the number of clusters is a difficult task for all clustering methods. However,

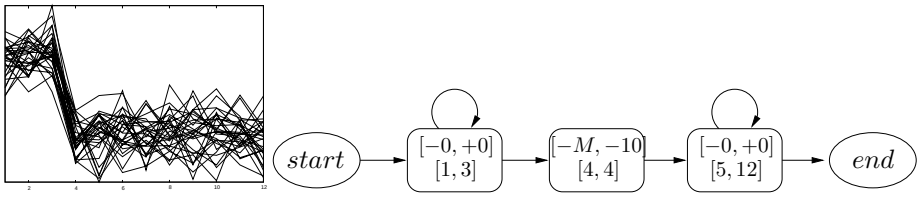


Figure 3: Left, examples of synthetic expression series added to the fibroblast dataset. Right, the HPM designed to find the synthetic class among the ”real” biological classes in the fibroblast dataset.

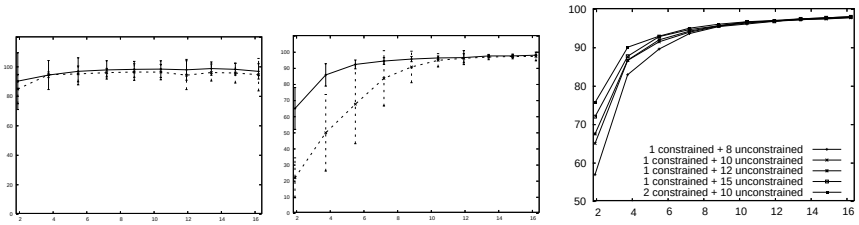


Figure 4: Recall (left) and precision (middle) achieved with (solid lines) and without (dashed lines) prior knowledge about the class of interest. The x-axes denote the proportion (in percent) of this class among all the expression series. Right, precision achieved using different number of components.

when the aim is to recover a particular class of genes rather than to infer a global clustering of the data, the problem is less acute. To illustrate this, we computed, in 100 runs, the precision and recall achieved with various numbers of constrained and unconstrained components, with the proportion of synthetic data ranging from 2% to 16% of the total data. We tried 1 constrained with 8, 10, 12 and 15 unconstrained components, and 2 constrained with 10 unconstrained components. All trials gave recall of up to 80% for all proportions of synthetic data (data not shown), and quite good precision —see right of Figure 4. Actually the best results are achieved with the highest numbers of components, so giving a sufficiently high number of components seems to be a good strategy to efficiently recover the clusters of interest.

3.3 Fibroblast dataset

Next, some experiments to find ”real” classes in the Fibroblast dataset have been carried out. This is the dataset of (Iyer *et al.*, 1999). Authors study the response of human fibroblasts to serum. The expression level of 8613 genes have been measured at 12 times, ranging from 15 min to 24 hours after serum stimulation. The authors selected a subset of 517 genes whose expression changed substantially in response to serum. The same subset, centered and reduced on genes is used here. First we clusterized the

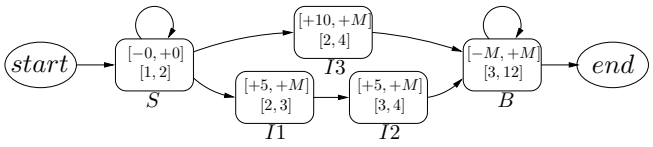


Figure 5: An HPM to uncover quick over-expression classes.

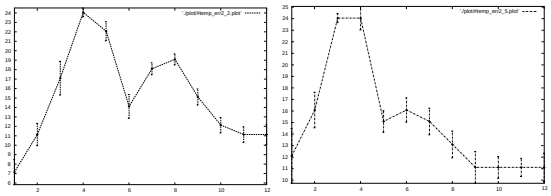


Figure 6: Fibroblast dataset. Two profiles obtained with the help of the HPM of Figure 5.

original data in 10 classes, using only knowledge about their temporal feature, i.e., by constraining all the components with an HPM like that of Figure 2. This clustering leads to 10 profiles relatively similar to those that (Iyer *et al.*, 1999) defined by hand after a hierarchical clustering. While most of these classes are well-defined, within them it is hard to identify genes that show a quick response to the serum. Only one jumbled class seems to present this feature. We designed an HPM specially adapted to such class (see Figure 5). The state S of this HPM models a potential and short —until time 2, at maximum— delay phase before over-expression. Next, 3 states are used to model the increasing phase: this can be quite moderate (at least 5 steps) during 2 times at least (states $I1$ and $I2$), or heavy (at least 10 steps) during 1 time (state $I3$). In both cases, the aim is to observe marked over-expression before time 5. The last state models the remainder of the class and is not constrained —all increases and decreases are allowed. We use a 10 components mixture model, with 3 components constrained with this special HPM, and 7 components constrained with an HPM like that of Figure 2.

Classes with the desired profile have been uncovered by this method. Figure 6 shows the mean profile of two classes. The third class has a very high variance at time 2, and a visual inspection shows that the collected series actually diverge from the profile at this point, so the class is not interesting. The two classes of Figure 6 differ by the time when genes reach their maximal over-expression —times 3-4 and times 4-5. Note that these classes show a second increase step which is not specified in the HPM we used. This illustrates the ability of the method to uncover the desired classes even when their profiles are not completely specified.

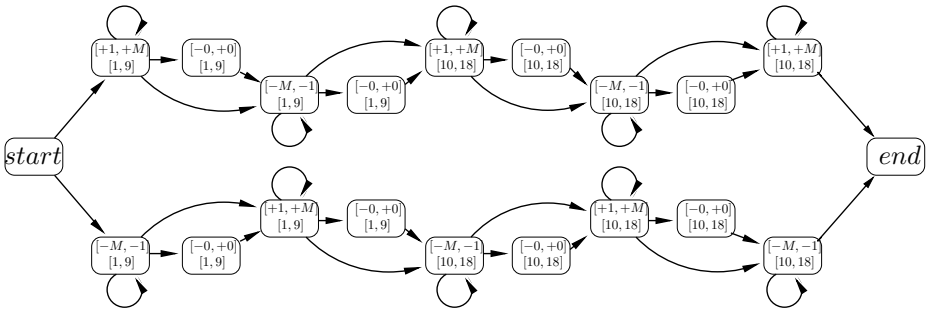


Figure 7: An HPM to find out sinusoidal profiles.

3.4 Yeast dataset

This is the dataset published in (Spellman *et al.*, 1998). Authors measure the expression level of 6178 genes 18 times during slightly more than two full cell cycles. We use the same normalization method as in (Spellman *et al.*, 1998): the logarithms of the data are centered and reduced on the genes, and genes that do not show any time points higher than 2 or lower than -2 are removed. This leads to a dataset of 1044 expression series. The main aim of the study was to find out cycle-regulated genes. So we look for classes showing a two-time repeat of the same pattern (since series span two cell cycles), i.e., classes with sinusoidal shape. The HPM of Figure 7 is designed for this purpose. It detects profiles that show whether (upper part of the HPM) 2 concave patterns —a concave pattern being an increase followed by a decreasing phase— eventually with a third increasing phase, or (lower part of the HPM) 2 convex patterns eventually followed by a third decreasing phase. Each increase or decrease can be followed by a short (one time) stability phase, and the time constraints of the τ functions require the convex or concave patterns to be equally distributed between first nine and last nine times.

A 20 component mixture model has been used for the clustering. The 10 first components have been constrained with HPM of Figure 7, while the 10 other components were not constrained to sinusoidal profiles but by the probabilistic HPM of Figure 2. Many classes that seem to be regulated by the cell cycle have been uncovered in this way. Figure 8 shows four of theses classes. These four differ by the times genes are over- or under-expressed. When superimposing the mean profiles of these classes on the same graph (see Figure 9), shifts between the different minima and maxima achieved can be seen.

4 Conclusions

We proposed a Bayesian approach for the clustering of gene expression series. This approach allows the user to easily integrate prior knowledge about the general profile of the classes of interest. This knowledge can be expressed by way of a probabilistic or non-probabilistic new type of model close to HMMs that we call a Hidden Phase

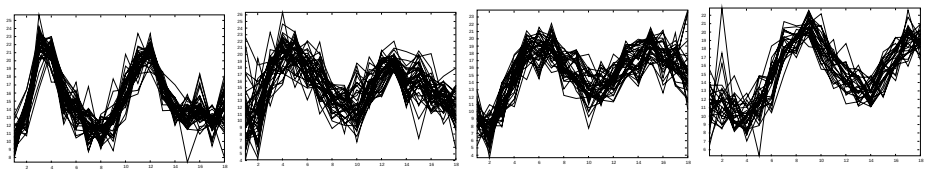


Figure 8: Yeast dataset. Four classes uncovered with the help of the HPM of Figure 7.

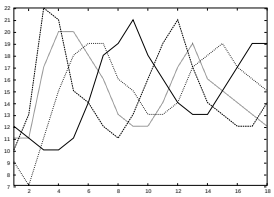


Figure 9: Yeast dataset. Superimposition of the four mean profiles of the classes of Figure 8.

Model. An HPM describes the profile of a particular class of genes. A mixture model is used to model the series, and each component of the mixture is associated with a given HPM. This defines the prior probability distribution of the parameters of the class. Then an EM algorithm is used to estimate the parameters of the mixture in maximizing its posterior probability. When no prior knowledge is available, our approach can naturally deal with the temporal nature of the series by using a specific and simple HPM.

Applied to two different datasets —(Spellman *et al.*, 1998) et (Iyer *et al.*, 1999)—, our method shows good performance and ability to efficiently uncover classes of genes with the desired profiles. In practice, appropriate HPMs can be designed easily and naturally. We experimentally observed on a mixture of natural and synthetic data that the benefit of the method increases when the number of expression series composing the classes of interest decreases with regard to the total number of series, and that it can be really interesting when this number is very low.

Many improvements seem possible on this basis. Indeed, other knowledge can be integrated in the HPMs. For example, knowledge about the desired mean expression level —and not about the *evolution* of the expression has it is done— could be easily added. Another improvement would be to introduce long-range dependences, i.e., to constrain differences of expression not only between consecutive times but also between separate times. For example, this would allow us to stipulate that the profiles should achieve their maximum at a specific time t.

Acknowledgements

I thank Olivier Martin, Gilles Caraux and Olivier Gascuel for their help and comments on this work.

References

ALON U., BARKAI N., NOTTERMAN D. A., GISH K., YBARRA S., MACK D. & LEVINE A. J. (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc Natl Acad Sci USA*, **96**(12), 6745–6750.

BAR-JOSEPH Z., GERBER G. K., GIFFORD D. K., JAAKKOLA T. S. & SIMON I. (2003). Continuous representations of time-series gene expression data. *J Comput Biol*, **10**(3-4), 341–356.

DEMPSTER A. P., LAIRD N. M. & RUBIN D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *J. Royal Stat. Soc. B*, **39**, 1–38.

DUDA R., HART P. & STORK D. (2001). *Pattern Classification*. John Wiley.

EISEN M. B., SPELLMAN P. T., BROWN P. O. & BOTSTEIN D. (1998). Cluster analysis and display of genome-wide expression patterns. *Proc Natl Acad Sci U S A*, **95**(25), 14863–14868.

HERTZBERG M., ASPEBORG H., SCHRADER J., ANDERSSON A., ERLANDSSON R., BLOMQVIST K., BHALERAO R., UHLEN M., TEERI T., LUNDEBERG J., SUNDBERG B., NILSSON P. & SANDBERG G. (2001). A transcriptional roadmap to wood formation. *Proc Natl Acad Sci USA*, **98**(25), 14732–14737.

HERWIG R., POUSTKA A. J., MULLER C., BULL C., LEHRACH H. & O'BRIEN J. (1999). Large-scale clustering of cDNA-fingerprinting data. *Genome Res*, **9**(11), 1093–105.

IYER V. R., EISEN M. B., ROSS D. T., SCHULER G., MOORE T., LEE J. C., TRENT J. M., STAUDT L. M., HUDSON J. J., BOGUSKI M. S., LASHKARI D., SHALON D., BOTSTEIN D. & BROWN P. O. (1999). The transcriptional program in the response of human fibroblasts to serum. *Science*, **283**(5398), 83–87.

KOHONEN T. (1997). *Self-Organizing Maps*. Springer.

LLOYD S. (1982). Least squares quantization in PCM. *IEEE Trans. Info. Theory*, **IT-2**, 129–137.

MCLACHLAN G. & KRISHNAN T. (2000). *Finite mixture models*. John Wiley.

RABINER L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, **77**(2), 257–285.

RAMONI M. F., SEBASTIANI P. & KOHANE I. S. (2002). Cluster analysis of gene expression dynamics. *Proc Natl Acad Sci USA*, **99**(14), 9121–9126.

SCHLIEP A., SCHONHUTH A. & STEINHOFF C. (2003). Using hidden markov models to analyze gene expression time course data. *Bioinformatics*, **19 Suppl 1**(14), 255–263.

SPELLMAN P. T., SHERLOCK G., ZHANG M. Q., IYER V. R., ANDERS K., EISEN M. B., BROWN P. O., BOTSTEIN D. & FUTCHER B. (1998). Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. *Mol Biol Cell*, **9**(12), 3273–3297.

TAMAYO P., SLONIM D., MESIROV J., ZHU Q., KITAREEWAN S., DMITROVSKY E., LANDER E. S. & GOLUB T. R. (1999). Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation. *Proc Natl Acad Sci U S A*, **96**(6), 2907–2912.

Exploiter l'information mutuelle inter-gènes pour réduire la dimension des données biopuces : une approche basée sur la construction automatique d'attributs

Blaise Hanczar, Jean-Daniel Zucker

EPML-CNRS IAPuces
Lim&Bio - Université Paris13
74, rue Marcel Cachin 93017 Bobigny Cedex
hanczar_blaise@yahoo.fr

Résumé : Cet article décrit et analyse expérimentalement une méthode originale de réduction de dimension pour les données biopuces. Les biopuces qui permettent de mesurer simultanément le niveau d'expression de milliers de gènes dans une condition donnée (tissu, cellule ou temps) produisent des données qui posent des problèmes spécifiques d'apprentissage automatique. La disproportion entre le nombre d'attributs (de l'ordre de la dizaine de milliers) et celui des exemples (de l'ordre de la centaine) requiert une réduction de dimension. Si l'information mutuelle gène/classe est souvent utilisée pour filtrer les gènes nous proposons une approche qui prend en compte celle de couple de gènes/classe. Plusieurs heuristiques de sélection de gènes basées sur ce principe sont proposées ainsi qu'une procédure de construction automatique d'attributs forçant les algorithmes d'apprentissage à tirer partie de ces couples de gènes. Les premiers résultats de réduction de dimension, puis de construction d'attributs et d'apprentissage sur plusieurs bases de données biopuces publiques montrent expérimentalement l'intérêt des approches.

1 Introduction

La transcriptomique est la description et l'analyse des données liées à l'étude des profils et de l'expression des gènes. Ce domaine a fait de gros progrès ces dernières années grâce en particulier aux puces à ADN (ou biopuces). Si un nombre croissant de projets bioscientifiques incluent désormais des études basées sur cette technologie c'est quelle permet de mesurer simultanément l'expression de plusieurs dizaines de milliers de gènes. Parmi les applications prometteuses de ces puces, il y a d'une part l'amélioration des diagnostics de certaines maladies comme le cancer mais aussi une meilleure compréhension de leur étiologie(Clement, 2000). Dans ces applications, le rôle de la classification est souvent crucial. Différents classeurs ont été utilisés pour

le diagnostic de cancers à partir de telles données : des réseaux Bayésiens, des arbres neuronaux, réseaux de neurones à base de fonctions radiales(Hwang *et al.*, 2002) ou des machines à vecteurs de support.

La tâche des bioinformaticiens du transcriptome est donc souvent de construire des classeurs à partir de matrice d'expressions où chaque condition ou patient est décrit par des valeurs réelles correspondant aux niveaux d'expression des gènes représentés sur la puce. Ces modèles doivent prédire aussi précisément que possible un paramètre clinique (comme le type de tumeur) représentant la classe. L'un des problèmes clefs lié à l'utilisation de biopuces est leur coût d'une part et la très grande variabilité inhérente à cette technologie d'autre part. Les jeux de données les plus larges disponibles dans la littérature comportent peu de patients (de 50 à quelques centaines) et un nombre important de gènes (de quelques centaines à quarante mille). Comme dans les applications de fouille de textes où les textes sont représentés par des sac-de-mots, ce déséquilibre entre le nombre d'exemples et le nombre d'attributs nuit gravement à la précision des algorithmes de prédictions. De fait, il a été démontré qu'un nombre trop important de dimensions favorisait le sur-apprentissage. Les modèles obtenus généralisent souvent mal le concept que l'on essaie d'apprendre, ce problème connu sous le nom de "la malédiction de la dimension"(Bellman, 1961) . Pour pallier ce problème on utilise classiquement en apprentissage des méthodes de réduction de dimensions. Le but de cette étape est d'identifier un sous-ensemble réduit d'attributs qui maximise les performances de prédiction. Ces méthodes sont largement utilisées dans le problème spécifique de l'analyse de données issues de puces à ADN.

La majorité des méthodes de réduction de données biopuces s'intéressent peu ou pas du tout aux interactions entre les gènes. La détection de ces interactions étant implicitement laissée à la charge des algorithmes d'apprentissage en aval de ces méthodes de sélection. Dans cet article nous postulons l'intérêt a priori de la prise en compte des interactions entre gènes dans la phase même de la réduction de dimension. Le tableau 1 ci-dessous qui représente le rang des 10 couples de gènes possédant la meilleure information mutuelle avec la classe illustre l'intérêt d'une telle prise en compte. Les couples de gènes en gras sont ceux qu'une procédure de sélection retenant les 100 meilleurs gènes, basés sur leur information mutuelle propre avec la classe, permettrait de reformer. Les autres couples sont ceux qui ne pourraient être reformés. Ainsi le premier couple de gènes ne pourrait être reformé après une telle réduction car le gène Hsa.22167 ne possède qu'une information mutuelle le reléguant en 592ième position et il ne serait pas conservé parmi les cent gènes les plus informatifs. Ce couple est un exemple d'interaction positive entre deux gènes. Or, et c'est là un point clef de notre approche, certains des couples à très forte interaction sont constitués de gènes qu'une sélection classique n'aurait pas retenue car trop peu informatif pris individuellement. Dans l'exemple donné, seul 4 des 10 meilleurs couples auraient été reformés si seul les 100 meilleurs gènes avaient été sélectionnés. Nous proposons dans cet article des heuristiques afin d'évaluer et d'identifier ces interactions pour réduire le nombre de dimensions des données biopuces tout en garantissant que des groupes de gènes ayant une très forte information mutuelle soient préservés par la procédure de réduction de dimension. Nous présentons ensuite un algorithme permettant par construction d'attributs de forcer les algorithmes d'apprentissage à prendre en compte ces gènes peu informatifs

mais à fortes interactions.

TAB. 1 – Ce tableau présente les couple de gènes présentant la meilleur information mutuelle avec la classe. Les couples en gras sont ceux qu'une procédure de sélection retenant les 100 meilleurs gènes basés sur l'information mutuelle permettrait de reformer, en non gras les couples qui ne pourraient être reformés.

	gene1	rang(Gene1)	gene2	rang(gene2)	Reformé
pair 1	Hsa.37937	1	Hsa.22167	592	Non
pair 2	Hsa.8147	2	Hsa.3933	355	Non
pair 3	Hsa.934	146	Hsa.1131	4	Non
pair 4	**Hsa.25322**	**5**	**Hsa.36696**	**33**	**Oui**
pair 5	**Hsa.22762**	**40**	**Hsa.7**	**9**	**Oui**
pair 6	Hsa.579	23	Hsa.5392	135	Non
pair 7	**Hsa.878**	**11**	**Hsa.442**	**95**	**Oui**
pair 8	**Hsa.6376**	**750**	**Hsa.1832**	**3**	**Oui**
pair 9	Hsa.6814	63	Hsa.2939	61	Non
pair 10	Hsa.1517	1583	Hsa.127	109	Non

La partie deux présente un état de l'art des méthodes de réduction de dimensions dans le domaine des données biopuces. Dans la troisième partie, nous proposons plusieurs heuristiques de calcul de couples de gènes à forte information mutuelle afin de baser la réduction sur ce calcul. Puis, dans la partie quatre, nous proposons un moyen d'exploiter cette information dans les algorithmes d'apprentissage automatique. Nous présentons dans la partie cinq les résultats expérimentaux des différents algorithmes sur plusieurs jeux de données internationaux et montrons l'intérêt de notre approche par un gain significatif de précision.

2 Etat de l'art

2.1 Vue d'ensemble des méthodes de réduction

La littérature sur les méthodes de sélection en apprentissage automatique est vaste (Liu & Motoda, 1998). La plupart de ces méthodes de réduction a été explorées sur des données biopuces et publiées dans la littérature scientifique. On peut les classer en trois familles : les méthodes de sélection de gènes, les méthodes de sélection de sous-ensemble gènes, les méthodes de reformulation.

Les méthodes de sélection de gènes utilisent un score de pertinence pour chaque gène, et fournissent une liste ordonnée de gènes. Ben-Dor compare plusieurs de ces méthodes et décrit une procédure pour estimer leurs p-value associées (Ben-Dor *et al.*, 2000). La méthode mise en oeuvre dans le logiciel SAM (Significant Analysis of Microarray) (Tusher *et al.*, 2001), aujourd'hui très populaire parmi les biologistes, est un représentant de cette famille. Cette procédure identifie des gènes différentiellement exprimés. Un autre représentant de cette famille est l'approche basée sur l'algorithme RELIEF proposée par Mary et al. (Mary *et al.*, 2003). La complexité algorithmique de ces méthodes

est faible et permet des mises en oeuvre efficaces (SAM est distribué comme un plug-in excel). Un autre avantage de ces méthodes est aussi la facilité d'interprétation des résultats qu'elles fournissent aux biologistes. Seuls les gènes ayant un score supérieur à un seuil sont conservés. En revanche ces méthodes conduisent souvent à des représentations pour lesquelles les performances des algorithmes d'apprentissage sont inférieures à celle des deux autres familles.

Les méthodes de sélection de sous-ensemble ne considèrent pas les gènes un à un par groupe de gènes. Une mesure de pertinence est définie pour un sous-ensemble de gènes. La sélection ou non d'un gène dépend donc des autres gènes et le problème d'optimisation qui consiste à trouver le meilleur sous-ensemble de gène est NP-complet. Il n'est donc par surprenant que cette famille compte de nombreuses techniques issues de l'apprentissage artificiel, comme les algorithmes génétiques (Li *et al.*, 2001), les méthodes Wrapper (Inza *et al.*, 2002), la méthode SVM-RFE (Reverse Feature Elimination) (Guyon *et al.*, 2002), etc. Si ces dernières méthodes s'appliquent dans le cas général, d'autres méthodes, plus originales ont été développées dans le cadre spécifique de l'analyse des biopuces. La méthode proposée par Xing qui combine "unconditionnal mixture modeling", information mutuelle et "markov blanket filtering" donne des résultats remarquables(Xing *et al.*, 2001). À l'opposé de la famille précédente, ces méthodes ont généralement de meilleures performances mais une complexité plus grande. L'interprétation biologique est là aussi naturelle car chaque méthode renvoie un sous ensemble de gènes.

Les méthodes de réduction par reformulation projettent les données dans un nouvel espace plus petit. Ce nouvel espace est défini par des attributs qui sont une combinaison des gènes. L'analyse par composantes principales est la plus connue des méthodes entrant dans cette dernière famille. Il existe d'autres méthodes de changement de représentation développées spécifiquement pour les données biopuces. C'est le cas de la méthode proposée par Qi(Qi, 2002) basée sur l'amplitude et la forme statistique des gènes pour construire de nouveaux attributs. L'algorithme PROGENE(Hanczar *et al.*, 2003) proposé par les auteurs est une autre méthode de cette famille qui crée des prototypes de gènes pour compresser l'information contenue dans des groupes de gènes dont l'expression est similaire. Cette famille donne généralement des représentations offrant de bonnes performances aux algorithmes d'apprentissage. En revanche, les algorithmes pour engendrer ces représentations sont de complexités importantes. A l'opposée des deux familles précédentes, l'interprétation biologique de ces représentations est difficile, car elles produisent un ensemble d'attributs qui sont des combinaisons de gènes qui n'ont pas de signification biologique immédiate.

Dans notre problématique, l'objectif est de sélectionner le ou les sous-ensemble(s) de gènes dont l'information mutuelle, avec la classe à prédire, est maximale. Le problème d'optimisation de l'information mutuelle exact d'un ensemble de gènes par rapport à une classe nécessite de calculer la probabilité jointe de tous les gènes sélectionnés et n'est donc de ce fait pas soluble en pratique. Pour limiter la complexité des algorithmes, il est généralement fait l'hypothèse que les gènes sont indépendants les uns des autres, dans ce cas il suffit de calculer la somme des informations mutuelles entre chaque gène et la classe. Des algorithmes récemment développés prennent en compte indirectement l'interaction entre les gènes en recherchant un ensemble qui maximise l'information

mutuelle entre chaque gène et la classe et qui minimise l'information mutuelle entre chaque gène sélectionné (Wu *et al.*, 2003; Xing *et al.*, 2001). L'approche que nous proposons consiste à utiliser les interactions entre les gènes en vue de réduire le nombre dimensions par un critère basé sur l'information mutuelle d'un groupe de gènes et la classe.

3 Réduire les dimensions en exploitant l'information mutuelle

L'entropie et l'information mutuelle sont ci-dessous brièvement redéfinies et explicitées dans le contexte des données biopuces.

3.1 Définition de l'information mutuelle

Considérons une variable aléatoire C pouvant prendre $n_C$valeurs. Après plusieurs mesures de C, on peut estimer empiriquement les probabilités$\{p(c_1), ..., p(c_{n_C})\}$ de chaque états $\{c_1, ..., c_{n_C}\}$de la variable C. L'entropie de Shannon (Shannon, 1948) de la variable est définie par :

$$H(C) = -\sum_{i=1}^{n_C} p(c_i) log_2(p(c_i))$$

L'information mutuelle mesure la dépendance entre deux variables. Plus cette valeur est élevée plus les variables sont liées, quand elle est nulle les variables sont indépendantes. Dans le contexte des puces à ADN, on utilise cette mesure pour identifier les gènes qui sont liés au paramètre bioclinique que l'on cherche à prédire, que nous appelons la classe. Soit C ce paramètre, l'information mutuelle entre C et un gène G se calcule par la formule suivante :

$$I(G, C) = H(G) + H(C) - H(G, C)$$

L'information mutuelle qu'apporte deux gènes peut se décomposer en une somme d'information mutuelle de chaque gène et d'interaction entre les gènes (Jakulin & Bratko, 2003).

$$I(G_1, G_2, C) = I(G_1, C) + I(G_2, C) + Inter(G_1, G_2, C)$$

avec

$$\begin{gathered} Inter(G_1, G_2) = -H(G_1, G_2, C) \\ +H(G_1, C) + H(G_2, C) + H(G_1, G_2) \\ -H(G_1) - H(G_2) - H(C) \end{gathered}$$

Notons que l'information d'une paire de gène n'est pas seulement la somme des informations de chaque gènes, mais qu'une interaction entre les gènes intervient également. Lorsque cette interaction est positive, on parle de *synergie* entre les gènes, lorsqu'elle

est négative on dit qu'il y a *redondance* entre les gènes. On peut généraliser ce principe pour des ensembles de gènes plus important $X = \{G_1, ..., G_p\}$en exploitant les interactions d'ordre supérieur :

$$I(X, C) = H(C) - H(C|X)$$

$$I(X, C) = \sum_{i=1}^{p} I(G_i, C) + \sum_{\Delta \subset X}^{n_A} Inter(\Delta, C)$$

$$Inter(X) = -\sum_{\Delta \subset X}^{p} (-1)^{|X|-|\Delta|} H(\Delta)$$

avec Δ un sous ensemble de gènes de X.

L'information mutuelle est strictement croissante avec le nombre de gènes, c'est à dire :

$$\forall Y, I(G_1, .., G_n, C) \leq I(G_1, ..., G_n, Y, C)$$

Notre objectif peut se reformuler ainsi : trouver le plus petit ensemble de gènes possible qui maximise l'information mutuelle avec la classe. Or du fait de la monotonie de l'information mutuelle en fonction du nombre N de gènes, on en déduit que pour trouver le sous-groupe de K gènes qui maximise l'information mutuelle, il suffit de sélectionner tous les groupes de K gènes disponibles. Cette solution pose un problème combinatoire évident. Comme nous l'avons vu plus haut il est donc indispensable de diminuer le nombre de gènes pour pallier ce problème de "malédiction de la dimension".

3.2 Quantifier l'information mutuelle

Du fait qu'il n'est pas possible en pratique de mesurer l'information mutuelle totale d'un ensemble important de gènes, on recourt à des approximations de cette valeur. La méthode la plus simple et la plus utilisée est de négliger les interactions entre les gènes. On ne calcule alors que la somme des informations mutuelles entre chaque gène et la classe. Soit $X = \{G_1, ..., G_p\}$un ensemble de gènes,

$$I(X, C) \approx \sum_{i=1}^{p} I(G_i, C)$$

Nous proposons ici d'utiliser une approximation plus fine, en prenant compte des interactions d'ordre supérieur, c'est-à-dire entre des ensembles de deux ou plus de gènes et la classe. En pratique nous nous limiterons dans cet article aux interactions entre des paires de gènes. Le calcul de l'information mutuelle dans ce cas est la somme des informations mutuelles entre chaque paire de gènes et la classe. Soit $X = \{P_1, ..., P_p\}$un ensemble de paires de gènes avec $P_i = \{G_{i1}, G_{i2}\}$,

$$I(X, C) \approx \sum_{i=1}^{p} I(G_{i1}, G_{i2}, C)$$

$$I(X,C) \approx \sum_{i=1}^{2p} I(G_i,C) + \sum_{i=1}^{p} Inter(G_{i1},G_{i2},C)$$

Etant donné le faible nombre d'exemples dont nous disposons, nous pouvons nous interroger sur la validité des calculs d'entropie. Il est bien connu que l'estimation de l'entropie à partir d'un ensemble fini d'exemples est biaisée. Roulston (Roulston, 1999) a montré que $H^{vrai}(X) = H^{observé}(X) + \frac{M_X - 1}{2N}$ avec M le nombre d'états de X et N le nombre d'exemples disponibles. Cette correction de l'entropie insérée dans le calcul de l'information mutuelle donne : $I^{vrai}(G,C) = I^{observé}(G,C) - \frac{M_{GC} - M_G - M_c + 1}{2N}$ et $I^{vrai}(G_1,G_2,C) = I^{observé}(G_1,G_2,C) - \frac{M_{G_1G_2C} - M_{G_1G_2} - M_C + 1}{2N}$. Notre objectif est d'identifier les gènes et paires de gènes les plus informatives. Ce n'est donc pas la valeur de l'information mutuelle qui nous intéresse, mais le classement des gènes et paires de gènes que l'on peut en extraire. Or, si le nombre d'états possible M est le même pour tous les gènes alors le classement des gènes et paires de gènes reste inchangé. On peut donc négliger, dans le problème qui nous intéresse, le biais de l'entropie dù au faible nombre d'exemples disponibles.

3.3 Une procédure de réduction basée sur l'information mutuelle de paires de gènes avec la classe

Notre objectif est de trouver un sous-ensemble de P gènes (avec P < N le nombre initial de gènes) tel que l'information mutuelle des couples de gènes par rapport à la classe soit maximale. Dans le cas où l'on néglige les interactions entre les gènes, le problème est simple, il suffit de calculer l'information mutuelle entre chaque gène et la classe, puis de sélectionner ceux dont les valeurs sont les plus élevées. Dans notre cas où l'on considère les interactions d'ordre trois (une paire de gènes et la classe), il faut explorer l'espace des paires de gènes de taille $O(N^2)$. Pour éviter d'explorer tout cet espace, une heuristique naturelle d'exploration des paires consistent à calculer d'abord les informations mutuelles de chaque gène et ensuite de former les N-1 premières paires à partir du meilleur gène et d'un autre gène. Pour découvrir P gènes différents il faut dans le pire des cas explorer toutes les paires soit N(N-1)/2. Etant donné le nombre important de gènes disponibles, nous recourons à une heuristique plus puissante permettant de limiter les paires à celles qui forment une sous-partition de l'ensemble des N gènes. Au delà de diviser l'espace des paires par deux (Pour N=2k cet espace est de taille $(1 + (N-1)) * (N/2)/2 = k^2$) c'est surtout par son aptitude à générer des paires formées de gènes distincts qu'elle tire son efficacité.

Le gène G1 ayant la plus grande information mutuelle $I(G_1,C)$ avec la classe est d'abord sélectionné, puis on recherche le gène G2 maximisant l'information mutuelle $I(G_1,G_2,C)$. La paire (G1, G2) est mise de côté et le processus est réitéré jusqu'à ce que P paires soient obtenues. Chaque gène ne peut donc faire partie que d'une seul paire, cela permet d'obtenir plus de variété dans les paires. Cette heuristique est décrite dans l'algorithme 1 qui correspond à l'extraction de paires sans remises.

Algorithme 1 Recherche de paires de gènes

1. $Paire \leftarrow \oslash$
2. $Gene \leftarrow all.gene$
3. pour i de 1 à nb.paire.max
   (a) $G1 \leftarrow argmax_g\,(I(g, C))$
   (b) $G2 \leftarrow argmax_g\,(I(G1, g, C))$
   (c) $Paire \leftarrow Paire \cup (G1, G2)$
   (d) $Gene \leftarrow Gene\text{-}\{G1, G2\}$
4. retourne (Paire)

4 Construction d'attributs pour représenter les paires les plus informatives

4.1 Le problème de l'apprentissage

Les algorithmes de classification usuels ne sont pas conçus pour traiter des paires de gènes. Et lorsqu'un algorithme de classification prend en entrée une liste de gènes ce qui a motivé leur sélection (synergie de certain des couples) est ignoré. Si les algorithmes utilisent en priorité les gènes les plus informatifs il y a peu de chances que les couples qui ont présidés à la réduction soient reformés. La phase de sélection de paires perd alors de son intérêt. Pour pallier ce problème, nous avons développé une approche par construction d'attribut. Un nouvel attribut est construit pour chaque paire, il synthétise l'information contenu dans chacun des deux gènes de la paire et dans leur synergie (interaction positive).

4.2 Construction d'attributs pour représenter les paires de gènes

Nous présentons ci-dessous deux méthodes, PCAMINFO et FEATKNN que nous avons envisagé pour construire de nouveaux attributs à partir de chacune des paires que nous venons d'identifier.

La méthode PCAMINFO que nous proposons s'inspire des travaux de Bollacker(K. & J., 1996). Nous utilisons la *matrice d'information* pour construire les nouveaux attributs, c'est une matrice carrée contenant autant de lignes et de colonnes qu'il y a de gènes, à chaque case (i,j) correspond l'information mutuelle de la paire de gène correspondante $I(G_i, G_j, C)$. Le principe de cette méthode repose sur l'analyse en composantes principales (ACP). Dans une ACP on calcule la matrice de covariance pour construire de nouveaux attributs qui maximisent la variance. Nous remplaçons dans cette méthode la matrice de covariance par la *matrice d'information*. Nous obtenons ainsi de nouveaux attributs qui sont des combinaisons linéaires de gènes, et qui maximisent l'information mutuelle.

La méthode FEATKNN, que nous avons développée, est limitée ici aux problèmes à

deux classes. Son adaptation aux problèmes multiclasses sort du cadre de cet article. Construire un nouvel attribut A requière de définir ses valeurs à partir des deux gènes $G_{i,1}$ $G_{i,2}$ de la paire P_i . Notre idée est que les valeurs de A_i doivent être déterminées par la densité des exemples de classes positives et négatives projetées dans l'espace à deux dimension $G_{i,1} \times G_{i,2}$. Soit un exemple de test t (dont la classe n'est pas connue), la valeur de l'attribut A_i pour cet exemple t se calcul de la sorte : $A_i(t) = -1+2\frac{n_{i,+}}{k}$ où $n_{i,+}$ est le nombre d'exemples de classe positive contenu dans les K plus proche voisin de t, en utilisant la distance euclidienne dans l'espace $G_{i,1} \times G_{i,2}$. Le nouvel attribut a des valeurs comprises entre -1 et 1, lorsque un exemple t est proche d'exemples de classe positive (resp. negative) l'attribut tend vers +1 (resp. -1). Un nouvel attribut est construit de la sorte pour chaque paire de gènes sélectionnées.

5 Expérimentation

Nous avons utilisé trois jeux de données biopuces pour évaluer expérimentalement la combinaison de la méthode de réduction et celles de reformulation que nous venons de décrire. Au delà de l'amélioration de performance, il s'agissait de tester notre hypothèse selon laquelle l'interaction entre le gènes permet d'améliorer l'apprentissage.

5.1 Données et prétraitement

Le premier des jeux de données porte sur le cancer du colon, le second sur la leucémie et le dernier sur un ensemble de tumeurs dit "small round blue-cell tumors" (SRBCT). Les deux premiers sont couramment utilisés dans la littérature pour tester les performances des algorithmes(Ben-Dor *et al.*, 2000). Les données du cancer du colon comportent 62 sujets dont 22 sont atteints d'un cancer du colon et les 40 autres sont sains. On dispose de l'expression de 2000 gènes pour chaque sujet, le but est de prédire si un patient est atteint d'un cancer ou non. Les données leucémie regroupent l'expression de 7129 gènes de 72 patients atteints de leucémie, 47 ont une "acute lymphoblastic leukemia" (ALL) et 25 ont une "acute myeloid leukemia" (AML). Le but est de prédire le type de leucémie de chaque patient. Les données SRBCT contient les données d'expression de 2308 gènes de 63 patients ayant une tumeur, dont 23 sont des tumeurs de type "sarcome d'Ewing" ("ewing sarcomas"), 20 sont de type "rhabdomyosarcomas", 12 sont de type "neuroblastomas" et 8 de type "burkitt lymphomas". Le but de cette tâche d'apprentissage est de prédire le type de tumeur dont est atteint le patient.

Pour des raisons d'efficacité, les données d'expression sont discrétisées. Nous utilisons une méthode par histogramme pour discrétiser l'expression de chaque gène. L'amplitude du gène est tout d'abord calculée, puis divisée en trois sous-intervalles de taille égale. Un gène peut donc prendre trois états : sur-exprimé, non modulé, sous-exprimé.

5.2 Analyse des paires les plus informatives

Afin de mesurer l'importance des interactions entre gènes, nous examinons empiriquement l'information mutuelle des meilleurs gènes et paires des données du cancer du colon. Ces données comportant relativement peu de gènes (2000) pour des données

TAB. 2 – Les 20 meilleurs paires de gènes des données biopuces du cancer du colon. Pour chaque paire de gènes (G1,G2), on a calculé l'information mutuelle (I) et le rang (rg) de chacun des deux gènes, ainsi que l'interaction (inter) et l'information mutuelle totale de la paire.

g1	I(G1)	rg(G1)	G2	I(G2)	rg(2)	inter	I(G1,G2)
Hsa.37937	0.47	1	Hsa.22167	0.06	592	0.26	0.79
Hsa.8147	0.38	2	Hsa.3933	0.08	355	0.16	0.62
Hsa.934	0.13	146	Hsa.1131	0.3	4	0.19	0.62
Hsa.25322	0.28	5	Hsa.36696	0.2	33	0.13	0.61
Hsa.22762	0.2	40	Hsa.7	0.26	9	0.14	0.6
Hsa.579	0.22	23	Hsa.5392	0.13	135	0.22	0.57
Hsa.878	0.25	11	Hsa.442	0.15	95	0.17	0.57
Hsa.6376	0.05	750	Hsa.1832	0.34	3	0.02	0.41
Hsa.6814	0.17	63	Hsa.2939	0.17	61	0.22	0.56
Hsa.1517	0.01	1583	Hsa.127	0.14	109	0.4	0.55
Hsa.812	0.14	123	Hsa.2451	0.24	13	0.17	0.55
Hsa.3305	0.24	15	Hsa.466	0.2	34	0.1	0.54
Hsa.42949	0.09	315	Hsa.2928	0.18	51	0.27	0.54
Hsa.821	0.23	22	Hsa.43431	0.06	542	0.25	0.54
Hsa.8068	0.18	59	Hsa.1317	0.18	54	0.18	0.54
Hsa.2386	0.07	474	Hsa.692	0.27	8	0.2	0.54
Hsa.36694	0.19	49	Hsa.1276	0.11	218	0.23	0.53
Hsa.1682	0.13	136	Hsa.21868	0.07	434	0.33	0.53
Hsa.692	0.27	8	Hsa.31801	0.13	138	0.12	0.52
Hsa.41280	0.23	18	Hsa.18787	0.02	1248	0.2	0.45

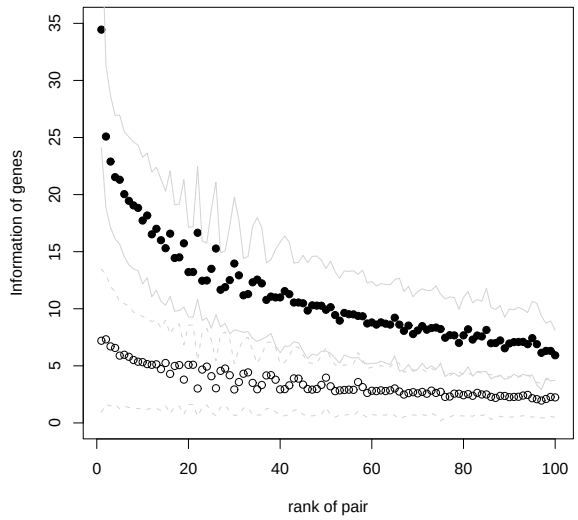


FIG. 1 – Information mutuelle par rapport à la classe des gènes formant les 1000000 meilleurs paires, chaque point représente un ensemble de 10000 paires. Les points noirs (resp. blanc) correspondent aux gènes ayant la plus grande (resp. petite) information mutuelle dans la paire.

biopuces, l'information mutuelle et l'interaction de toutes les paires de gènes a donc pu être calculé, soit 1998000 paires. L'information mutuelle de chaque gène par rapport à la classe a également été calculée, ce qui a permis de définir un classement avec un rang pour chaque gène. Ces résultats sont regroupés dans un tableau dans lequel chaque paire est caractérisée par le nom, l'information mutuelle et le rang de chacun des deux gènes qui la constitue ainsi que l'interaction et l'information mutuelle totale de la paire. Le tableau 2 montre les 20 meilleures paires des données du cancer du colon. Dans cet exemple la meilleure paire est formée par le meilleur gène (Hsa.37937) et par le 592ème meilleur gène (Hsa.22167), les informations mutuelles par rapport à la classe de ces gènes sont respectivement de 0.47 et 0.06. L'interaction entre ces deux gènes est de 0.26, l'information mutuelle totale de cette paire est de 0.47+0.06+0.26=0.79. Le tableau 2 montre que les valeurs d'interaction sont aussi importantes que les informations mutuelles de chaque gène. Ceci abonde dans le sens de notre hypothèse selon laquelle les interactions entre les gènes ne sont pas négligeables.

La figure 1 montre l'information mutuelle des gènes formant les 1000000 meilleurs paires, chaque point représente un ensemble de 10000 paires. Les points noirs (resp. blanc) correspondent aux gènes ayant la plus grande (resp. petite) information mutuelle dans la paire. Nous pouvons déduire deux choses de ce graphique, premièrement les gènes ayant une faible information mutuelle peuvent très bien faire partis des

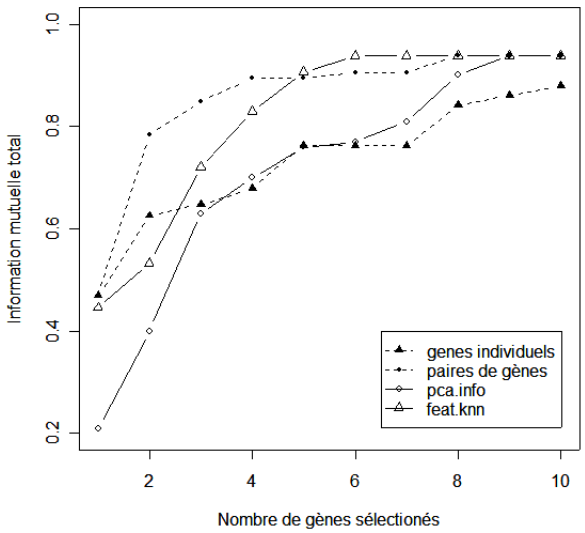


FIG. 2 – Comparaison de l'information mutuelle totale des gènes sélectionnées par les différentes méthodes. noir :sélection de gènes, rouge :sélection de paires, bleu :pca.info

meilleures paires. Deuxièmement, on constate que toutes les meilleures paires de gènes sont constituées en moyenne d'un gène très informatif et d'un second moyennement ou peu informatif. Cette constatation montre l'intérêt du choix de notre heuristique de sélection de paires que l'on forme à partir de gènes ayant le meilleur rang.

5.3 Information mutuelle obtenue par les différentes heuristiques

L'objectif que nous avons défini est la sélection de l'ensemble de gènes le plus petit possible maximisant l'information mutuelle, nous testons comment les heuristiques présentées FEATKNN ET PCAMINFO répondent à cet objectif. Chaque heuristique a été utilisée sur les trois jeux de données pour construire un ensemble de 10 attributs. Nous avons donc un ensemble comportant les 10 gènes de plus haut rang, un comportant les 5 meilleures paires de gènes et 2 ensembles comportant 10 attributs construits par FEATKNN et PCAMINFO. L'information mutuelle de ces ensembles a été mesurée et comparée. Nous nous sommes limités à des ensembles de 10 attributs maximum, au-delà le temps de calcul pour obtenir l'information mutuelle devient trop important. La figure 2 montre les résultats sur les données du cancer du colon en fonction de la taille du sous-ensemble de gènes. On constate que seul l'ensemble des gènes de plus haut rang ne parvient pas à atteindre l'information mutuelle maximum, sa courbe croit lentement, les gènes de cet ensemble ont probablement une interaction faible. Ces résultats expérimentaux indique que, sur les données étudiées, prendre en compte les interactions entre

les gènes permet de maximiser l'information mutuelle en utilisant moins de gènes ou d'attributs.

5.4 Apprentissage automatique

Il a été montré ci-dessus que les méthodes qui prennent en compte l'interaction entre les gènes, permettent d'obtenir des ensembles de gènes ou d'attributs plus informatifs et de plus petite taille que les méthodes qui les négligent. On peut donc supposer que ces méthodes vont améliorer les performances en classification. Nous évaluons dans cette partie l'impact de ces méthodes de réduction sur la classification.

5.4.1 Algorithmes de classification

Des recherches récentes ont mis en évidence les bonnes performances de certains algorithmes de classification sur les données biopuces. Une étude présentée par Lee, compare notamment 21 algorithmes de classification sur 7 jeux de données biopuces, et conclut que les meilleures performances sont obtenues par les méthodes les plus sophistiquées, en particulier les machines à vecteurs de support(Lee *et al.*, In press). À l'opposé, Dudoit met en évidence les remarquables performances de méthodes simples (Dudoit *et al.*, 2002). Le choix de la méthode de classification à utiliser sur les données biopuces est donc une question qui reste ouverte et qui dépend des particularités des données utilisées. Pour évaluer les techniques de réduction, nous avons réalisés nos expérimentations avec les trois méthodes considérées comme les meilleures dans les deux papiers cités, c'est à dire : les machines à vecteur de support (SVM), les k plus proches voisins (KNN) et l'analyse discriminante diagonale linéaire (DLD). Elles sont toutes trois de fait très utilisées dans la communauté de l'analyse de données issues de puces à ADN.

5.4.2 Evaluation des performances

Une fois les données discrétisées, la méthode de réduction de dimension puis de construction d'attributs sont appliquées. Les trois algorithmes de classification mentionnés plus haut sont ensuite utilisés et l'erreur en généralisation du modèle obtenu est évaluée. La validation croisée est couramment utilisée pour calculer l'erreur en généralisation. Toutefois Braga-Neto a montré que cet estimateur n'est pas le plus approprié avec des données comportant peu d'exemples, comme c'est le cas avec les puces à ADN. La validation croisée à une grande variance et les estimateurs par bootstrap sont préférés dans ce cas et en particulier l'estimateur .632(Braga-Neto & Dougherty, 2004). Pour éviter le biais du à la sélection de gènes mis en évidence par Ambroise et McLahan (Ambroise & McLachlan, 2002), l'évaluation de l'erreur en généralisation du modèle s'effectue dans une boucle externe au processus de sélection de gènes. C'est à dire qu'à chaque itération de l'estimateur .632, un échantillon bootstrap d'exemples est sélectionné, cet ensemble est utilisé pour la sélection de gènes puis pour la construction du model. Les exemples non sélectionnés dans l'échantillon bootstrap sont utilisés pour calculer l'erreur du modèle et n'interviennent à aucun moment dans le processus de sélection de gènes.

TAB. 3 – Résultats en classification

	Réduction	Leucémie	cancer du colon	SRCBT
SVM	tous les gènes	12.3	17.5	10.7
	genes individuel	4.3	12.5	2.1
	paires de gènes	4.8	11.8	1.9
	pca.info	4.3	12.1	**1.4**
	feat.knn	**2.8**	**10.7**	N/A
KNN	tous les gènes	8.4	20	15.9
	genes individuel	6.1	13.9	5
	paires de gènes	6.2	14.4	3.7
	pca.info	**5.2**	13.6	**3**
	feat.knn	5.3	**13.5**	N/A
DLD	tous les gènes	11.5	19.5	6.2
	genes individuel	4.8	14.7	1.2
	paires de gènes	4.8	15.4	**1**
	pca.info	5.2	12.9	1.1
	feat.knn	**3.8**	**12.5**	N/A

5.4.3 Résultats en classification

Le tableau 3 récapitule les différents résultats en classification, rapelons que la méthode FEAT.KNN ne traite pas les problèmes multiclasses, c'est pourquoi elle a pas de résultats sur les données SRCBT. On constate sans surprise que les méthodes de réduction de dimension améliorent considérablement les performances en classification. Les méthodes de sélections des meilleurs gènes et des meilleures paires donnent des résultats similaires. Nous venons de voir plus haut que les paires de gènes apportaient plus d'information, comment expliquer que ces paires n'améliorent pas les performances en classification ? Il est vraisemblable que l'information contenue dans l'interaction entre les gènes d'une paire ne soit pas totalement exploitée par les algorithmes de classification. Une grande partie de l'information calculée durant la phase de sélection de paires est alors perdue. Les deux méthodes de construction d'attributs ont de meilleures performances, en particulier FEATKNN. Les nouveaux attributs construits par ces méthodes synthétisent l'information contenue dans les gènes et leurs interactions. Dans ce cas la classification exploite l'interaction entre les gènes par l'intermédiaire de ces attributs, ce qui explique de meilleurs résultats. Il n'en reste pas moins que l'interprétation biologique devient alors différente des approches classiques. Il n y a plus de gènes maximalement discriminant mais des listes de couples. Les couples de gènes peuvent éventuellement être utilisés dans l'étude des réseaux de régulation.

6 Conclusion

Nous avons présenté dans cet article plusieurs procédures de réduction de dimensions de données biopuces afin d'améliorer les performances en classification. Ces méthodes

sont basées sur l'hypothèse que l'information apportée par l'interaction entre les gènes n'est pas négligeable. Nous nous sommes limités dans cette étude aux interactions entre les paires de gènes. Si quantifier l'information des gènes et des interactions à partir du calcul d'information mutuelle est naturel, cette simple réduction n'améliore pas nécessairement les performances. Nous avons développé deux méthodes de construction d'attributs, PCAMINFO et FEATKNN qui permettent de forcer les algorithmes d'apprentissage à prendre en compte des paires présentant une forte information mutuelle. L'intérêt expérimental de ces interactions à été évaluée sur trois jeux de données où les performances sont améliorées. Nos travaux actuels s'orientent vers la prise en compte de synergies entre des groupes plus larges de gênes, ainsi qu'une analyse théorique des gains obtenus par ces approches de réduction de dimensions.

7 Remerciements

Nous remercions les relecteurs pour leurs critiques constructives et interessantes. Nous remercions aussi tout le sevice de nutrition de l'hopital Hotel-Dieu pour l'aide sur l'interprétation des données d'expression.

Références

AMBROISE C. & MCLACHLAN G. M. J. (2002). Selection bias in gene extraction on the basis of microarray gene expression data. *Proc. Natl. Acad. Sci.*, **99**(10), 6562–6566.

BELLMAN R. (1961). *Adaptive Control Processes : A Guided Tour*. Princeton University Press.

BEN-DOR A., FRIEDMAN N. & YAKHINI Z. (2000). *Scoring genes for relevance*. Rapport interne AGL-2000-13, Agilent Technologies.

BRAGA-NETO U. & DOUGHERTY E. (2004). Is cross-validation valid for small-sample microarray classification ? *Bioinformatics*, **20**(3), 374–380.

CLEMENT (2000). Monogenic forms of obesity : From mice to human. *Ann Endocrinol*.

DUDOIT S., FRIDLYAND J. & SPEED P. (2002). Comparison of discrimination methods for classification of tumors using gene expression data. *Journal of American Statististial Association*, **97**, 77–87.

GUYON I., WESTON J., BARNHILL S. & VAPNIK V. (2002). Gene selection for cancer classification using support vector machines. *Machine Learning*, **46**, 389–422.

HANCZAR B., COURTINE M., BENIS A., HENEGAR C., CLÉMENT K. & ZUCKER J. (2003). Improving classification of microarray data using prototype-based feature selection. *SIGKDD Explorations*, **5**, 23–30.

HWANG K., CHO D., PARK S., KIM S. & ZHANG B. (2002). Applying machine learning techniques to analysis of gene expression data : Cancer diagnosis. In *Methods of Microarray Data Analysis (Proceedings of CAMDA'00)*, p. 167–182 : Kluwer Academic Publichers.

INZA I., SIERRA B., BLANCO R. & LARRAÑAGA P. (2002). Gene selection by sequential wrapper approaches in microarray cancer class prediction. *Journal of Intelligent and Fuzzy Systems*, p. 25–34.

JAKULIN A. & BRATKO I. (2003). Analyzing attribute dependencies. *Proceedings A of the 7th European Conference on Principles and Practice of Knolegde Discovery in Databases (PKDD)*, p. 229–240.

K. B. & J. G. (1996). Linear feature extractors based on mutual information. In *Int. Conf. On Pattern Recognition (ICPR96)*.

LEE J. W., LEE J. B., PARK M. & SONG S. H. (In press). An extensive comparison of recent classification tools applied to microarray data. *Computational Statistics and Data Analysis*.

LI L., DARDEN T., WEINBERG C., LEVINE A. & PEDERSEN L. (2001). Gene assessment and sample classification for gene expression data using a genetic algorithm/k-nearest neighbor method. *Combinatorial Chemistry and High Throughput Screening*, p. 727–739.

LIU H. & MOTODA H. (1998). *Feature Selection for Knowledge Discovery and Data Mining*. Kluwer Adcademic Publishers.

MARY J., MERCIER G., COMET J., CORNUÉJOLS A., FROIDEVAUX C. & DUTREIX M. (2003). An attribute estimation technique for the analysis of microarray data. In *Proc. of the Dieppe Spring school on Modelling and simulation of biological processes in the context of genomics*.

QI H. (2002). Feature selection and knn fusion in molecular classification of multiple tumor types. *International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences (METMBS'02)*.

ROULSTON M. (1999). Estimating the errors on measured entropy and mutual information. *Physica D*, **125**, 285–294.

SHANNON E. (1948). A mathemitical theory of communication. *The Bell System Technical Journal*, **27**, 623–656.

TUSHER V., TIBSHIRANI R. & CHU G. (2001). Significance analysis of microarrays applied to the ionizing radiation response. *PNAS*, **98**, 5116–5121.

WU X., YE Y. & ZHANG L. (2003). Graphical modeling based gene interaction analysis for microarray data. *SIGKDD Exploration*, **5**, 91–100.

XING E., JORDAN M. & KARP R. (2001). Feature selection for high-dimensional genomic microarray data. In *Proceedings of the Eighteenth International Conference in Machine Learning, ICML2001*.

Classification of Domains with Boosted Blast*

Cécile Capponi[1], Gwennaele Fichant[2], and Yves Quentin[2]

[1] LIF - CNRS, Université de Provence, 39, avenue Joliot Curie
13453 Marseille Cedex 13, France
(e-mail: capponi@cmi.univ-mrs.fr)

[2] LMGM - IBCG - CNRS, Université Paul Sabatier, 118, route de Narbonne,
31062 Toulouse Cedex, France
(e-mail: fichant@bibcg.biotoul.fr, quentin@ibcg.biotoul.fr)

1 Introduction

One way of predicting the function of a protein is to identify a known domain (subsequence of amino-acids within the whole sequence of the protein) in the protein, despite sequence modifications which may occur during evolution. In many cases, comparing a new sequence of protein p with few sequences of the family F under study is enough for predicting whether $p \in F$. Such a similarity search may be achieved for example by using an alignment program such as BLAST (Altschul *et al.*, 1990). However, when the domain of a family is not well conserved, there is no satisfying method for retrieving this domain onto a new sequence. This is the case of Membrane Spanning Domains (MSDs), which plays the role of a pore through which a substrate goes in and/or out of the cell : its sequences are not conserved. The IRIS strategy (Quentin *et al.*, 2002) gives good results on MSDs, but requires their previous subdivision into 18 subfamilies. We thus propose here to use the boosting technique in order to learn the whole family from BLAST alignements of protein sequences, without any pre-clustering.

2 Boosting BLAST

Let $S = \{(x_1, y_1), ..., (x_n, y_n)\}$ a set of annotated protein sequences, where a protein $x_i \in X$ is labelled $y_i = +1$ if x_i carries a MSD, and -1 otherwise. For aligning two sequences, the heuristic wide-used algorithm BLAST assigns an *e-value* to the optimal alignment it computes : the smaller the e-value, the most significant the alignment. Let $\mathcal{A}(x, D, \tau)$ be a formatted result of BLAST : it is a set that contains all the proteins of the database D aligned with x with an e-value less than τ. In order to classify proteins according to the unconserved domain under study, we propose to use AdaBoost (Freund & Schapire, 1997), where BLAST instantiates the method that produces a weak decision

*This work is granted by the national ACI IMPBIO program.

rule, here represented by a stump (algorithm 1). We expect boosting to extend the local predictivity of BLAST to a global predictivity.

Algorithm 1 BlastBoost(τ,T) where T is the number of boosting steps, and τ is the e-value threshold

Given : $(x_1, y_1), \cdots, (x_n, y_n)$ where $x_i \in X$ and $y_i \in Y = \{-1, +1\}$
Initialize $D_1(i) \leftarrow 1/n$
for all $t = 1, \cdots, T$ **do**
Select $x_{i,t}$ according to D_t and Compute $\mathcal{A}_t = \mathcal{A}(x_{i,t}, X, \tau)$ with BLAST
Get $h_t : X \rightarrow \{-1, +1\}$ such that $\forall x \in X$:

$$\text{if } x \in \mathcal{A}_t \text{ then } h_t(x) = y_{i,t} \text{ else } h_t(x) = \text{argmax}_{k \in \{-1,+1\}} \sum_{j, y_j = k, x_j \notin \mathcal{A}_t} D_t(j)$$

Compute $\epsilon_t = \sum_{i=1..n, h_t(x_i) \neq y_i} D_t(i)$ and $\alpha_t = \frac{1}{2} \ln \left(\frac{1-\epsilon_t}{\epsilon_t} \right)$
Update : $D_{t+1}(i) \leftarrow \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$
end for
Output the final classifier : $H(x) = \text{sign} \left(\sum_{t=1}^{T} \alpha_t h_t(x) \right)$

We actually tuned and tested a variant of BlastBoost, where each decision tree involves three queries selected according to D_t, and where two e-value thresholds are considered, depending on the label of the query x_t. We used eight genomes to learn the model, and eight other genomes to test it. Both learning and test sets are made up of all the examples of considered genomes (proteins that carry a MSD), and all the counter-examples (proteins aligned to one example with BLAST while not carrying a MSD). The obtained results are similar when swapping the learning and test genomes.

The test error gets stable after around 400 rounds. The selectivity of BlastBoost is almost as good as this of the IRIS method, while its sensitivity is better (up to 0.999, while 0.946 with IRIS). Yet, IRIS previously subdivides the functional families into 18 subfamilies, so its learning and testing steps are independent from one subfamily to another, therefore more accurate. Hence the results of BlastBoost are good : with comparable results, it is efficient on the whole functional family even if the proteins sequences are not conserved.

Références

ALTSCHUL S., GISH W., MILLER W., MYERS E. & LIPMAN D. (1990). Basic local alignment search tool. *Journal of Molecular Biology*, **215**, 403–410.

FREUND Y. & SCHAPIRE R. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, **55**(1), 119–139.

QUENTIN Y., CHABALIER J. & FICHANT G. (2002). Strategies for the identification, the assembly and the classification of integrated biological systems in completely sequenced genomes. *Computers and Chemistry*, **26**, 447–457.

Extraction de concepts sous contraintes dans des données d'expression de gènes*

Baptiste Jeudy[1], François Rioult[2]

[1] Équipe Universitaire de Recherche en Informatique de St-Etienne (EURISE),
Université de St-Etienne.
baptiste.jeudy@univ-st-etienne.fr

[2] GREYC - CNRS UMR 6072,
Université de Caen Basse-Normandie
francois.rioult@info.unicaen.fr

Résumé : L'une des activités les plus importantes en biologie est l'analyse des données d'expression de gènes. Les biologistes espèrent ainsi mieux comprendre les fonctions des gènes et leurs interactions. Nous étudions dans cet article une technique permettant d'aider à l'analyse de ces données d'expression : l'extraction de concepts sous contraintes. Pour cela, nous proposons d'extraire des fermés sous contraintes dans les données "transposées" en utilisant des algorithmes classiques. Ceci nous amène a étudier la "transposition" des contraintes dans les données transposées de manière à pouvoir les utiliser dans ces algorithmes.

Mots-clés : Extraction de connaissances, Data-mining, Concepts Formels, Itemsets Fermés, Contraintes.

1 Motivations

Maintenant que le décodage du génome est terminé pour de nombreuses espèces animales et végétales, il reste encore un formidable défi pour la biologie moderne : comprendre la fonction de tous ces gènes et la manière dont ils interagissent entre-eux. Pour cela, les biologistes mènent des expériences de mesure de l'expression de gènes. Celles-ci ont pour but de leur fournir des données leur permettant de faire des hypothèses sur ces fonctions et ces interactions.

Les données d'expression de gènes se présentent typiquement sous la forme d'une matrice binaire. Chaque colonne représente un gène et chaque ligne donne les résultats d'une expérience de mesure du niveau d'expression des gènes. Chacune de ces expériences consiste à déterminer, pour une cellule donnée issue d'une situation biologique donnée (par exemple un organe spécifique, une culture cellulaire), quels sont les gènes qui sont sur-exprimés, c'est-à-dire ceux qui ont une

*Ce travail a été partiellement financé par l'ACI Bingo

activité biologique importante au moment de la mesure. Dans la matrice, les gènes qui sont sur-exprimés dans une situation biologique sont codés par un 1. Ceux qui ne le sont pas sont codés par un 0. La table 1 donne un exemple d'une telle matrice.

	Gène 1	Gène 2	Gène 3	Gène 4
cellule 1	1	1	1	0
cellule 2	1	1	1	0
cellule 3	0	1	1	1

TAB. 1 – Exemple de matrice d'expression de gènes

Dans cet article, nous étudions une technique de fouille de données permettant d'aider le biologiste à faire des hypothèses sur les fonctions des gènes et la manière dont ils interagissent. Pour cela, les techniques d'extraction de motifs semblent particulièrement adaptées. Il existe cependant de nombreux types de motifs : les itemsets, les itemsets fermés ou libres, les règles d'association ou encore les concepts formels. Nous avons choisi ici d'étudier l'extraction des concepts.

Dans ce cadre, un concept formel est une paire (G, E) où G est un ensemble de gènes (i.e., un ensemble de colonnes de la matrice) appelé intension du concept et E un ensemble d'expériences (i.e., un ensemble de lignes) appelé extension du concept. Ces ensembles sont tels que si $g \in G$ et $e \in E$, alors le gène g est sur-exprimé dans l'expérience e (il y a un 1 dans la ligne e colonne g). De plus, les deux ensembles G et E sont maximaux, i.e., ils ne peuvent pas grossir sans perdre la propriété précédente (une définition plus formelle des concepts est donnée dans la section 2). Autrement dit, un concept est une sous-matrice maximale ne contenant que des 1. Dans notre matrice exemple, ({Gène 1, Gène 2, Gène 3}, {cel 1, cel 2 }) est un concept.

Du point de vue du biologiste, les concepts sont très intéressants. En effet, un concept (G, E) regroupe des gènes qui sont sur-exprimés dans les mêmes expériences. Si la fonction de certains de ces gènes est connue, cela peut permettre de faire des hypothèses sur la fonction de ceux qui sont inconnus. De plus, si les expériences apparaissant dans l'extension E partagent des propriétés communes (par exemple, elles concernent toutes des cellules du foie ou des cellules cancéreuses), cela permet encore une fois de faire des hypothèses sur les gènes. Le fait que les concepts associent à la fois des gènes et des expériences est donc un avantage par rapport à d'autres motifs comme les itemsets ou les règles d'association qui ne portent que sur les gènes. De plus, un gène (ou une expérience) peut apparaître dans plusieurs concepts (par opposition à ce qui se passe dans le cas du clustering). Si le biologiste s'intéresse à un gène particulier, il peut donc étudier quels sont les gènes liés à celui-ci (i.e., apparaissant dans les mêmes concepts) suivant les situations biologiques. Cela est très important car il s'avère en effet qu'un gène peut intervenir dans plusieurs fonctions biologiques différentes. Enfin, les concepts sont beaucoup moins nombreux que les itemsets tout en représentant la même information : ils sont donc plus simples à exploiter.

Pour simplifier encore l'exploitation de ces concepts par le biologiste, l'utilisa-

tion de contraintes semble pertinente : le biologiste peut indiquer une contrainte qui doit être satisfaite par tous les concepts extraits. Par exemple, il peut imposer qu'un gène particulier (ou ensemble de gène) apparaisse (ou pas) dans les concepts extraits. Il peut aussi se restreindre aux concepts impliquant des expériences sur des cellules cancéreuses ou contenant au moins 5 gènes. L'utilisation des contraintes permet finalement au biologiste de mieux cibler sa recherche.

1.1 Notre contribution

Nous proposons dans cet article d'étudier l'extraction de concepts sous contraintes dans des données d'expression de gènes. Cette extraction pose deux problèmes principaux :

1. utilisation des contraintes : nous laissons la possibilité à l'utilisateur de spécifier une contrainte portant à la fois sur l'intension et l'extension du concept. Ces contraintes sont utiles pour l'utilisateur pour préciser sa recherche mais elles sont aussi parfois indispensables pour rendre l'extraction faisable. En effet, il est généralement impossible d'extraire tous les concepts. Il faut donc dans ce cas utiliser les contraintes *pendant* l'extraction (et non pas seulement dans une phase de filtrage des concepts *après* l'extraction) pour diminuer la complexité celle-ci.
2. taille des données : la complexité des algorithmes d'extraction est généralement linéaire par rapport au nombre de lignes et exponentielle par rapport au nombre de colonnes. Or dans le cas des données d'expression de gènes, le nombre de colonnes est souvent très important : l'utilisation de techniques comme les puces à ADN permet d'obtenir l'expression de milliers de gènes en une seule expérience. D'un autre coté, le nombre d'expériences est souvent réduit du fait du temps nécessaire à leur mise en place et de leur coût. Ceci amène à des matrices comportant beaucoup de colonnes (jusqu'à plusieurs milliers) et relativement peu de lignes (quelques dizaines ou centaines) ce qui est plutôt atypique dans le domaine du data-mining. Les algorithmes classiques ne sont donc pas bien adaptés à ce type de données.

L'extraction de motifs sous contrainte est un thème de recherche qui a été très étudié ces dernières années (Srikant *et al.*, 1997; Ng *et al.*, 1998; Garofalakis *et al.*, 1999; Boulicaut & Jeudy, 2000; Pei & Han, 2000; Zaki, 2000; Boulicaut & Jeudy, 2001; Bucila *et al.*, 2003; Albert-Lorincz & Boulicaut, 2003; Bonchi *et al.*, 2003; Bonchi & Lucchese, 2004)... De nombreux algorithmes ont été proposés et tentent d'utiliser efficacement les contraintes pour diminuer les temps d'extraction en élaguant le plus tôt possible l'espace de recherche. L'extraction de concept est fortement liée à l'extraction d'itemsets libres ou fermés dont l'étude a également donné lieu à de nombreux travaux (Pasquier *et al.*, 1999; Boulicaut *et al.*, 2000; Pei *et al.*, 2000; Zaki & Hsiao, 2002; Boulicaut *et al.*, 2003)...

Cependant, ces travaux ne font pas d'extraction de concepts sous contrainte et ne sont pas adaptés à des données ayant plus de colonnes que de lignes. En ce qui concerne l'extraction de concepts sous contraintes, une proposition

récente à été faite dans (Besson *et al.*, 2004). Cependant, l'algorithme proposé, D-Miner, ne permet que de traiter un type particulier de contraintes, les contraintes monotones. Nous verrons dans la section 4 comment l'étude que nous proposons ici va nous permettre de traiter aussi les contraintes anti-monotones avec cet algorithme.

En ce qui concerne le second problème, plusieurs propositions ont été faites récemment pour le résoudre : L'algorithme CARPENTER (Pan *et al.*, 2003) est conçu pour extraire les fermés fréquents dans une base de données avec plus de colonnes que de lignes. Dans (Rioult *et al.*, 2003; Rioult & Crémilleux, 2004), les auteurs utilisent des algorithmes classiques mais au lieu de faire l'extraction dans les données originales, ils travaillent sur la matrice transposée. Dans ce cas, la matrice transposée comporte beaucoup de lignes et peu de colonnes, ce qui permet d'utiliser les techniques habituelles efficacement. Cependant, ces travaux ne traitent que du cas de la contrainte de fréquence ou de contraintes simples sur les itemsets. Le cas général où la contrainte est une formule booléenne construite à partir de contraintes simples, portant à la fois sur l'intension et l'extension, n'est pas abordé.

Notre proposition est donc d'utiliser des algorithmes classiques (éventuellement légèrement modifiés) dans la matrice transposée, afin de travailler sur des données au format plus classique (peu de colonnes, beaucoup de lignes). Pour pouvoir traiter des contraintes complexes portant sur les concepts, nous allons présenter ici une étude théorique sur les contraintes et sur la manière de les "transposer" (en fait, il s'agira plutôt d'une projection) de façon à pouvoir les utiliser dans la matrice transposée.

Cet article est organisé de la manière suivante : dans la section 2, nous rappelons quelques définitions à propos de l'extraction d'itemsets et de la correspondance de Galois. Nous présentons ensuite formellement le problème que nous cherchons à résoudre. Dans la section 3, nous présentons la projection des contraintes simples et composées. Ensuite, la section 4 montre comment utiliser la projection de contraintes et l'extraction dans la matrice transposée pour résoudre notre problème. Finalement, nous concluons dans la section 5.

2 Définitions

Pour éviter les confusions entre les lignes (ou colonnes) de la base de données originale et les lignes (ou colonnes) de base de données "transposée", nous définissons une base de données comme une relation entre deux ensembles : un ensemble d'attributs et un ensemble d'objets. L'ensemble des **attributs** (ou items) est noté $\mathcal{A}$ et correspond, dans notre application biologique, à l'ensemble des gènes. L'ensemble des **objets** est noté $\mathcal{O}$ et représente les situations biologiques. L'**espace des attributs**, $2^{\mathcal{A}}$, est la collection des sous-ensembles de $\mathcal{A}$, appelés **itemsets** et l'**espace des objets**, $2^{\mathcal{O}}$, est la collection des sous-ensembles de $\mathcal{O}$. Lorsqu'on considère l'ordre défini par l'inclusion ensembliste, chacun des espaces $2^{\mathcal{A}}$ et $2^{\mathcal{O}}$ est naturellement muni d'une structure de treillis.

Une base de données est une relation binaire de $\mathcal{A} \times \mathcal{O}$ et peut être représentée

	a_1	a_2	a_3	a_4
o_1	1	1	1	0
o_2	1	1	1	0
o_3	0	1	1	1

	o_1	o_2	o_3
a_1	1	1	0
a_2	1	1	1
a_3	1	1	1
a_4	0	0	1

TAB. 2 – Représentation originale et transposée de la base de données présentée table 1. Les attributs sont $\mathcal{A} = \{a_1, a_2, a_3, a_4\}$ et les objets sont $\mathcal{O} = \{o_1, o_2, o_3\}$. Nous utilisons une notation sous forme de chaîne pour les ensembles, par exemple $a_1a_3a_4$ désigne l'ensemble d'attributs $\{a_1, a_3, a_4\}$ et o_2o_3 désigne l'ensemble d'objets $\{o_2, o_3\}$. Cette base de données sera utilisée dans tous les exemples.

par une matrice booléenne dont les colonnes sont les attributs et les lignes sont les objets. Cette matrice constitue la représentation originale de la base. Au cours de cet article, nous considérerons que la base de données a plus d'attributs que d'objets et nous utiliserons également la représentation transposée des données, où les attributs de la base sont portés sur les lignes et les objets sur les colonnes (cf. Table 2).

2.1 Correspondance de Galois

L'idée principale qui fonde notre travail est d'utiliser la correspondance forte entre les treillis des $2^{\mathcal{A}}$ et $2^{\mathcal{O}}$, appelée **correspondance de Galois**. Cette correspondance a été utilisée la première fois en fouille de données quand des algorithmes d'extraction des itemsets fermés fréquents ont été proposés (Pasquier *et al.*, 1999) et elle est aussi utilisée dans de nombreux travaux en apprentissage conceptuel (Wille, 1992; Nguifo & Njiwoua, 2000).
Étant donnée une base de données *bd*, les opérateurs f et g de Galois sont définis par :

– f, appelé *intension*, est une fonction de $2^{\mathcal{O}}$ vers $2^{\mathcal{A}}$ définie par

$$f(O) = \{a \in \mathcal{A} \mid \forall o \in O,\ (a, o) \in bd\}\,,$$

– g, appelé *extension*, est une fonction de $2^{\mathcal{A}}$ vers $2^{\mathcal{O}}$ définie par

$$g(A) = \{o \in \mathcal{O} \mid \forall a \in A,\ (a, o) \in bd\}\,.$$

Pour un ensemble A, $g(A)$ est aussi appelé **l'ensemble support** de A dans *bd*. C'est l'ensemble des objets qui sont en relation avec tous les attributs de A. La **fréquence** de A dans *bd*, notée $\mathsf{Freq}(A)$, est définie par $\mathsf{Freq}(A) = |g(A)|$.

Ces deux fonctions créent un lien entre l'espace des attributs et l'espace des objets. Pourtant, comme les deux espaces n'ont a priori pas le même cardinal, aucune bijection n'est possible entre eux. Cela signifie que plusieurs ensembles d'attributs ont la même image par g dans l'espace des objets et vice-versa. On peut donc définir deux relations d'équivalence r_a et r_o sur $2^{\mathcal{O}}$ et $2^{\mathcal{A}}$:

– si A et B sont deux ensembles d'attributs, $A\, r_a\, B$ si $g(A) = g(B)$,

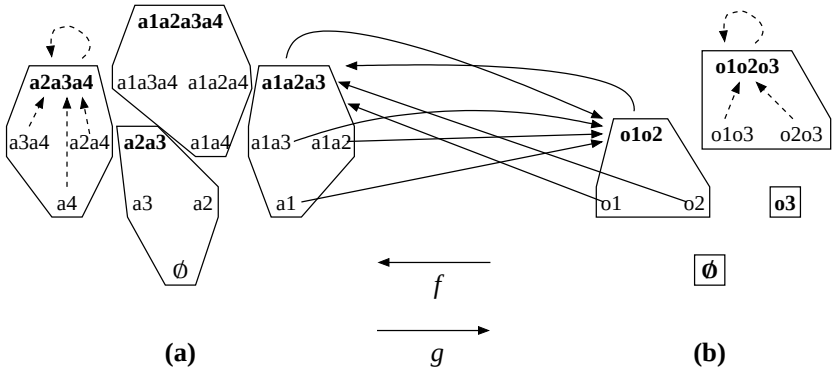


FIG. 1 – Les classes d'équivalence pour r_a dans le treillis des attributs (a) et pour r_o dans celui des objets (b). Les ensembles fermés sont en gras. Les flèches représentent les opérateurs f et g entre les classes de $a_1a_2a_3$ et o_1o_2. Les flèches en pointillés représentent les opérateurs de clôture h et h'.

– si O et P sont deux ensembles d'objets, $O\, r_o\, P$ si $f(O) = f(P)$.

Dans chaque classe d'équivalence, il y a un élément particulier : le plus grand élément d'une classe, au sens de l'inclusion, est unique et appelé **ensemble d'attributs fermé** pour r_a ou **ensemble d'objets fermé** pour r_o. Les opérateurs f et g de Galois fournissent, par composition, deux opérateurs de **fermeture** notés $h = f \circ g$ et $h' = g \circ f$. Les ensembles fermés sont les points fixes des opérateurs de fermeture et la fermeture d'un ensemble est l'ensemble fermé de sa classe d'équivalence. Dans la suite, nous évoquerons indifféremment h ou h' avec la notation cl.

Une paire (A, O) constituée d'un ensemble d'attributs fermé A et de l'ensemble d'objets fermé correspondant O est appelée un **concept formel**. L'ensemble des concepts de la base de données bd est noté :

$$\mathsf{Concepts}(bd) = \{(A, O) \mid f(O) = A \wedge g(A) = O\}\,.$$

Exemple 1
Dans la figure 1, les ensembles d'objets fermés sont $\emptyset$, o_3, o_1o_2, *et* $o_1o_2o_3$. *Les ensembles d'attributs fermés sont* a_2a_3, $a_2a_3a_4$, $a_1a_2a_3$ *et* $a_1a_2a_3a_4$. *Comme* $g(o_1o_2) = a_1a_2a_3$ *et* $f(a_1a_2a_3) = o_1o_2$, $(a_1a_2a_3, o_1o_2)$ *est un concept. Les autres concepts sont* $(a_2a_3, o_1o_2o_3)$, $(a_2a_3a_4, o_3)$, $(a_1a_2a_3a_4, \emptyset)$.

Propriété 1
A et B sont des ensembles d'attributs, O et P des ensembles d'objets et E un ensemble d'attributs ou d'objets.

– *f sont g sont décroissantes par rapport à l'inclusion : si $A \subseteq B$ alors $g(B) \subseteq g(A)$ et si $O \subseteq P$, $f(P) \subseteq f(O)$;*
– *$f \circ g \circ f = f$;*
– *E est fermé si et seulement si $\mathsf{cl}(E) = E$ et sinon $E \subseteq \mathsf{cl}(E)$;*

– (A, O) *est un concept si et seulement si* O *est fermé et* $A = f(O)$

2.2 Contraintes

Afin de permettre au biologiste de focaliser son étude sur les concepts qui l'intéressent réellement, nous lui laissons la possibilité de définir une contrainte qui devra être satisfaite par tous les concepts extraits.

Si on note $\mathcal{B}$ l'ensemble des bases de données booléennes (i.e., des matrices booléennes), on appelle **contrainte sur les concepts** une fonction booléenne $\mathcal{C}$ de $2^{\mathcal{A}} \times 2^{\mathcal{O}} \times \mathcal{B}$.

Outre le fait qu'une contrainte permet de mieux cibler les ensembles extraits, leur utilisation, lorsqu'elles sont efficacement intégrées à l'algorithme d'extraction, permet également de réduire considérablement le temps de calcul. C'est ce qui explique l'intérêt croissant ces dernières années pour l'étude des algorithmes d'extraction sous contraintes. Cependant, les contraintes utilisées dans ces algorithmes ne portent généralement que sur les itemsets (et pas simultanément sur les itemsets et les ensembles d'objets). Mais, dans la section suivante, nous verrons comment projeter une contrainte sur les concepts pour obtenir une contrainte ne portant plus que sur les objets, et ainsi pouvoir utiliser des techniques classiques d'extraction sous contraintes (sauf que nous les utiliserons dans les données transposées).

Parmi les contraintes portant sur les itemsets, la plus utilisée est sans doute la contrainte de fréquence minimale $\mathcal{C}_{\gamma\text{-}\mathsf{freq}}$. Cette contrainte est satisfaite par les itemsets dont la fréquence est supérieure à un seuil *gamma* fixé par l'utilisateur : $\mathcal{C}_{\gamma\text{-}\mathsf{freq}}(X) = (\mathsf{Freq}(X) > \gamma)$. On peut également être intéressé sa négation : c'est-à-dire chercher des itemsets suffisamment rares et donc utiliser une contrainte de fréquence maximale. Il existe également de nombreuses contraintes syntaxiques. Une contrainte est syntaxique lorsqu'elle ne dépend pas de la matrice des données *bd*. Par exemple, la contrainte $\mathcal{C}(A) = a_1 \in A$ est syntaxique, alors que la contrainte de fréquence ne l'est pas (en effet, la fréquence d'un itemset dépend des données).

Parmi les contraintes syntaxiques, les contraintes de "sur-ensemble" et de "sous-ensemble" permettent par combinaison (conjonction, disjonction, négation) de construire les autres contraintes syntaxiques (cf. table 3). Étant donné un ensemble constant E, la contrainte de sous-ensemble $\mathcal{C}_{\subseteq E}$ est définie par : $\mathcal{C}_{\subseteq E}(X) = (X \subseteq E)$. La contrainte de sur-ensemble $\mathcal{C}_{\supseteq E}$ est définie par : $\mathcal{C}_{\supseteq E}(X) = (X \supseteq E)$. Remarquons que comme nous allons ensuite utiliser des contraintes sur les itemsets et les ensembles d'objets, les ensembles X et E peuvent soit être tous les deux des itemsets soit tous les deux des ensembles d'objets.

Lorsqu'une valeur numérique $a.v$ est associée à chaque attribut a (par exemple un coût), on peut définir d'autres contraintes syntaxiques du type (Ng *et al.*, 1998) $\mathrm{MAX}(X)\,\theta\,\alpha$ (où $\theta \in \{<, >, \leq, \geq\}$) pour différents opérateurs d'agrégation tels que MAX, MIN, SOM (la somme), MOY (la moyenne). Parmi ces contraintes, celles qui utilisent les opérateurs MIN et MAX peuvent être récrites simplement en utilisant les contraintes $\mathcal{C}_{\supseteq E}$ et $\mathcal{C}_{\subseteq E}$ en utilisant l'ensemble

$\mathsf{sup}_\alpha = \{a \in \mathcal{A} \mid a.v > \alpha\}$ comme indiqué dans la table 3.

$X \not\subseteq E \equiv \neg\mathcal{C}_{\subseteq E}(X)$	$X \cap E = \emptyset \equiv X \subseteq \overline{E}$
$X \not\supseteq E \equiv \neg\mathcal{C}_{\supseteq E}(X)$	$X \cap E \neq \emptyset \equiv \neg(X \subseteq \overline{E})$
$\text{MIN}(X) > \alpha \equiv X \subseteq sup_\alpha$	$\text{MAX}(X) > \alpha \equiv X \cap sup_\alpha \neq \emptyset$
$\text{MIN}(X) \leq \alpha \equiv X \not\subseteq sup_\alpha$	$\text{MAX}(X) \leq \alpha \equiv X \cap sup_\alpha = \emptyset$

$$|X \cap E| \geq 2 \equiv \bigvee_{1 \leq i < j \leq n} e_i e_j \subseteq X$$

TAB. 3 – Exemples de contraintes obtenues par combinaison des contraintes de sur-ensemble et de sous-ensemble. $E = \{e_1, e_2, ..., e_n\}$ est un ensemble constant et X un ensemble variable. Le complémentaire de E dans $\mathcal{A}$ ou dans $\mathcal{O}$ est noté $\overline{E}$.

Le fait de récrire toutes ces contraintes syntaxiques en utilisant uniquement les contraintes $\mathcal{C}_{\subseteq E}$ et $\mathcal{C}_{\supseteq E}$ nous permettra de limiter le nombre de contraintes à étudier dans la section 3 sur la projection des contraintes.

Finalement, toutes ces contraintes peuvent être combinées pour construire une contrainte sur les concepts, par exemple $\mathcal{C}(A, O) = (a_1 a_2 \subseteq A \wedge (O \cap o_4 o_5 = \emptyset))$.

Pour pouvoir utiliser efficacement les contraintes dans les algorithmes d'extraction, il est nécessaire d'étudier leurs propriétés. Ainsi, deux types de contraintes importantes ont été mises en évidence : les contraintes monotones et les contraintes anti-monotones. Une contrainte $\mathcal{C}$ est **anti-monotone** si $\forall A, B \ (A \subseteq B \wedge \mathcal{C}(B)) \Longrightarrow \mathcal{C}(A)$. $\mathcal{C}$ est **monotone** si $\forall A, B \ (A \subseteq B \wedge \mathcal{C}(A)) \Longrightarrow \mathcal{C}(B)$. Dans les deux définitions, A et B peuvent être des ensembles d'attributs ou d'objets. La contrainte de fréquence est anti-monotone. L'anti-monotonicité est une propriété importante, parce que les algorithmes d'extraction par niveaux l'utilisent la plupart du temps pour élaguer l'espace de recherche. En effet, quand un ensemble ne satisfait pas la contrainte, ses spécialisations non plus et elles peuvent donc être élaguées (Agrawal *et al.*, 1996).

Les compositions élémentaires de telles contraintes ont les mêmes propriétés : la conjonction ou la disjonction de deux contraintes anti-monotones (resp. monotones) est anti-monotone (resp. monotone). La négation d'une contrainte anti-monotone est monotone, et vice-versa.

2.3 Définition du problème

Nous définissons la tâche d'extraction de concepts sous contraintes de la manière suivante : étant donnés une base de données bd, une contrainte $\mathcal{C}$ sur les concepts, nous voulons extraire l'ensemble des concepts qui satisfont $\mathcal{C}$, c'est-à-dire la collection :

$$\{(A, O) \in \mathsf{Concepts}(bd) \mid \mathcal{C}(A, O, bd)\}\,.$$

3 Projections de contraintes

Pour extraire les concepts sous contraintes, nous proposons d'utiliser des techniques classiques d'extraction de fermés sous contraintes dans la matrice transposée. Cependant, dans ces algorithmes, les contraintes possibles portent uniquement sur les itemsets. Donc, s'ils sont utilisés dans la matrice transposée, les contraintes porteront sur les ensembles d'objets.

Pour permettre leur utilisation, nous allons donc étudier dans cette section un mécanisme de "projection" de contrainte : Étant donné une contrainte $\mathcal{C}$ portant sur les concepts (c'est-à-dire à la fois sur l'intension et l'extension), nous voulons calculer une contrainte $p(\mathcal{C})$ portant uniquement sur les ensembles d'objets et telle que la collection des ensembles fermés d'objets satisfaisant cette contrainte soit exactement la collection des extensions des concepts satisfaisant $\mathcal{C}$. De cette manière, La collection des concepts satisfaisant $\mathcal{C}$ est exactement la collection des $(f(O), O)$ tels que O est fermé et satisfait la contrainte projetée $p(\mathcal{C})$.

$$\{(A,O) \in \mathsf{Concepts}(bd) \mid \mathcal{C}(A,O,bd)\} = \{(f(O),O) \in \mathcal{A} \times \mathcal{O} \mid p(\mathcal{C})(O,bd) \wedge O \in \mathsf{Fermés}(bd)\}\,.$$

Ainsi, pour résoudre notre problème, il suffira d'extraire les ensembles fermés d'objets O satisfaisant $p(\mathcal{C})$ et de générer tous les concepts de la forme $(f(O), O)$.

3.1 Définitions et propriétés

Cela signifie que nous voulons extraire des ensembles fermés d'objets O tels que le concept $(f(O), O)$ satisfasse la contrainte $\mathcal{C}$. Par conséquent, une définition naturelle de la projection de la contrainte $\mathcal{C}$ est :

Définition 1 (Contrainte projetée)
Étant donnée une contrainte $\mathcal{C}$ sur les concepts, nous définissons la contrainte projetée de $\mathcal{C}$ de la façon suivante :

$$p(\mathcal{C})(O, bd) = \mathcal{C}(f(O), O, bd),$$

Le proposition suivante assure que l'on obtient bien le résultat voulu :

Proposition 1
Soit $\mathcal{C}$ une contrainte sur les concepts, bd une base de donnée et $p(\mathcal{C})$ la projection de la contrainte $\mathcal{C}$. Alors :

$$\{(A,O) \in \mathsf{Concepts}(bd) \mid \mathcal{C}(A,O,bd)\} = \{(f(O),O) \in \mathcal{A} \times \mathcal{O} \mid p(\mathcal{C})(O,bd) \wedge O \in \mathsf{Fermés}(bd)\}\,.$$

Preuve : $(A,O) \in \mathsf{Concepts}(bd) \wedge \mathcal{C}(A,O,bd) \Leftrightarrow O \in \mathsf{Fermés}(bd) \wedge A = f(O) \wedge \mathcal{C}(A,O,bd) \Leftrightarrow O \in \mathsf{Fermés}(bd) \wedge \mathcal{C}(f(O),O,bd) \Leftrightarrow O \in \mathsf{Fermés}(bd) \wedge p(\mathcal{C})(O,bd)$. □

Par conséquent, pour extraire la collection des concepts qui satisfont $\mathcal{C}$, nous pouvons utiliser des algorithmes classiques d'extraction de fermés dans la matrice transposée avec la contrainte $p(\mathcal{C})$. Cependant, il faut vérifier que cette contrainte $p(\mathcal{C})$ est effectivement utilisable dans ces algorithmes.

Nous allons commencer par étudier les contraintes complexes, c'est-à-dire des contraintes construites à partir de contraintes plus simples en utilisant des opérateurs booléens comme la conjonction, la disjonction ou la négation.

Proposition 2
Si $\mathcal{C}$ et $\mathcal{C}'$ sont deux contraintes sur les concepts, alors :

$$p(\mathcal{C} \wedge \mathcal{C}') = p(\mathcal{C}) \wedge p(\mathcal{C}'),$$
$$p(\mathcal{C} \vee \mathcal{C}') = p(\mathcal{C}) \vee p(\mathcal{C}'),$$
$$p(\neg\mathcal{C}) = \neg p(\mathcal{C}).$$

Preuve : Pour la conjonction : $p(\mathcal{C} \wedge \mathcal{C}')(O) = (\mathcal{C} \wedge \mathcal{C}')(f(O), O) = \mathcal{C}(f(O), O) \wedge \mathcal{C}'(f(O), O) = (p(\mathcal{C}) \wedge p(\mathcal{C}'))(O)$. La preuve est similaire pour la disjonction et la négation. □

Cette proposition permet de "pousser" la projection dans les contraintes complexe. L'étape suivante est donc d'étudier ce qui se passe au niveau des contraintes élémentaires.

Ces contraintes élémentaires peuvent porter sur l'intension du concept (ex : $\mathcal{C}(A, O) = (a_1 \in A)$) ou sur son extension (ex : $\mathcal{C}(A, O) = (|O \cap o_1 o_3 o_5| \geq 2)$. Les contraintes élémentaires qui ne portent que sur l'extension des concepts ne sont pas modifiées par la projection, nous allons donc nous focaliser sur les contraintes portant sur les itemsets.

Les contraintes les plus efficacement prises en compte par les algorithmes d'extraction sous contrainte sont les contraintes monotones et anti-monotones. Il est donc important d'étudier comment se comporte la projection de contraintes par rapport à ces propriétés :

Proposition 3
Soit $\mathcal{C}$ une contrainte sur les itemsets :
- *si $\mathcal{C}$ est anti-monotone alors $p(\mathcal{C})$ est monotone ;*
- *si $\mathcal{C}$ est monotone alors $p(\mathcal{C})$ est anti-monotone.*

Preuve : Si O est un ensemble d'objet, $p(\mathcal{C})(O) = \mathcal{C}(f(O))$ par définition de la projection. Or f est décroissante par rapport à l'inclusion (cf. prop. 1) d'où les propriétés. □

3.2 Projection de contraintes classiques

Dans la section précédente, nous avons donné la définition de la projection de contrainte. Cette définition fait intervenir $f(O)$. Cela signifie que pour tester la contrainte projetée, il est nécessaire, pour chaque ensemble d'objets O, de calculer son intension $f(O)$. Certains algorithmes, tels que CHARM (Zaki &

Hsiao, 2002), utilisent une structure de données particulière –la représentation verticale des données– et par conséquent calculent pour chaque ensemble O l'ensemble $f(O)$. Cependant, beaucoup d'autres algorithmes n'utilisent pas cette structure et ne peuvent donc directement utiliser les contraintes projetées. C'est pour cette raison que dans cette section nous étudions les contraintes projetées de contraintes classiques et nous calculons une expression de ces contraintes ne faisant plus intervenir $f(O)$.

Nous allons d'abord étudier la contrainte de fréquence minimale (qui est la contrainte la plus courante) : $\mathcal{C}_{\gamma\text{-freq}}(A) = (\mathsf{Freq}(A) > \gamma)$. Par définition, sa contrainte projetée est : $p(\mathcal{C}_{\gamma\text{-freq}})(O) = (\mathsf{Freq}(f(O)) > \gamma)$. Par définition de la fréquence, $\mathsf{Freq}(f(O)) = |g(f(O))| = |\mathsf{cl}(O)|$ et si O est un ensemble fermé d'objets, $\mathsf{cl}(O) = O$ et par conséquent $p(\mathcal{C}_{\gamma\text{-freq}})(O) = (|O| > \gamma)$. Finalement, la projection de la contrainte de fréquence minimale est une contrainte de taille minimale. Si on avait considéré la contrainte de fréquence maximale, on aurait évidement trouvé comme projection une contrainte de taille maximale.

De par la symétrie du problème, il découle que la projection de la contrainte de taille maximale (resp. minimale) est la contrainte de fréquence : si $\mathcal{C}(A) = (|A|\,\theta\,\alpha)$ alors $p(\mathcal{C})(O) = (|f(O)|\,\theta\,\alpha)$. Or $|f(O)|$ est exactement la fréquence de O si on se place dans la matrice transposée.

Les deux propositions suivantes donnent l'expression de la projection des contraintes de sur-ensemble et de sous-ensemble :

Proposition 4
Soit E un itemset, alors :

$$p(\mathcal{C}_{\supseteq E})(O) \equiv g(E) \supseteq \mathsf{cl}(O).$$

Preuve : $p(\mathcal{C}_{\supseteq E})(O) \Leftrightarrow (E \subseteq f(O)) \Rightarrow (g(E) \supseteq g \circ f(O)) \Leftrightarrow (g(E) \supseteq \mathsf{cl}(O))$. Réciproquement, $(g \circ f(O) \subseteq g(E)) \Rightarrow (f \circ g \circ f(O) \supseteq f \circ g(E)) \Rightarrow (f(O) \supseteq \mathsf{cl}(E)) \Rightarrow f(O) \supseteq E$. □

Proposition 5
Soit E un itemset, alors, si E est fermé :

$$p(\mathcal{C}_{\subseteq E})(O) \equiv g(E) \subseteq \mathsf{cl}(O),$$

si E n'est pas fermé, on pose $\overline{E} = \mathcal{A} \setminus E = \{f_1, ..., f_m\}$ et :

$$p(\mathcal{C}_{\subseteq E})(O) \equiv (\mathsf{cl}(O) \not\supseteq g(f_1) \wedge \mathsf{cl}(O) \not\supseteq g(f_2) \wedge ... \wedge \mathsf{cl}(O) \not\supseteq g(f_m).$$

Preuve : $p(\mathcal{C}_{\subseteq E})(O) \Leftrightarrow \mathcal{C}_{\subseteq E}(f(O)) \Leftrightarrow (f(O) \subseteq E) \Rightarrow (g \circ f(O) \supseteq g(E)) \Leftrightarrow (\mathsf{cl}(O) \supseteq g(E))$. Réciproquement, (si E est fermé) : $(g(E) \subseteq g \circ f(O)) \Rightarrow (f \circ g(E) \supseteq f \circ g \circ f(O)) \Rightarrow (\mathsf{cl}(E) \supseteq f(O)) \Rightarrow (E \supseteq f(O))$. Si E n'est pas fermé, on récrit la contrainte : $(A \subseteq E) = f_1 \notin A \wedge ... \wedge f_m \notin A$ et on utilise les propositions 2 et 4. □

La table 3.2 récapitule les contraintes projetées de contraintes classiques. Les contraintes de fréquence et de taille ont été traitées plus haut. Les deux propriétés

Contrainte $\mathcal{C}(A)$	Contrainte projetée $p(\mathcal{C})(O)$
$\mathsf{Freq}(A)\,\theta\,\alpha$	$\lvert O\rvert\,\theta\,\alpha$
$\lvert(\rvert\,A)\,\theta\,\alpha$	$\mathsf{Freq}(O)\,\theta\,\alpha$
$A \subseteq E$	si E est fermé : $g(E) \subseteq O$
	sinon : $O \not\subseteq g(f_1) \wedge ... \wedge O \not\subseteq g(f_m)$
$E \subseteq A$	$O \subseteq g(E)$
$A \not\subseteq E$	si E est fermé : $g(E) \not\subseteq O$
	sinon : $O \subseteq g(f_1) \vee ... \vee O \subseteq g(f_m)$
$E \not\subseteq A$	$O \not\subseteq g(E)$
$A \cap E = \emptyset$	si $\overline{E}$ est fermé : $g(\overline{E}) \subseteq O$
	sinon : $O \not\subseteq g(e_1) \wedge ... \wedge O \not\subseteq g(e_n)$
$A \cap E \neq \emptyset$	si $\overline{E}$ est fermé : $g(\overline{E}) \not\subseteq O$
	sinon : $O \subseteq g(e_1) \vee ... \vee O \subseteq g(e_n)$
$\mathrm{SOM}(A)\,\theta\,\alpha$	$\mathsf{Freq}_p(O)\,\theta\,\alpha$
$\mathrm{MOY}(A)\,\theta\,\alpha$	$\mathsf{Freq}_p(O)/\mathsf{Freq}(O)\,\theta\,\alpha$
$\mathrm{MIN}(A) > \alpha$	$p(A \subseteq sup_\alpha)$
$\mathrm{MIN}(A) \leq \alpha$	$p(A \not\subseteq sup_\alpha)$
$\mathrm{MAX}(A) > \alpha$	$p(A \cap sup_\alpha \neq \emptyset)$
$\mathrm{MAX}(A) \leq \alpha$	$p(A \cap sup_\alpha = \emptyset)$
$\theta \in \{<, >, \leq, \geq\}$	

TAB. 4 – Contraintes projetées. A est un ensemble variable d'attributs, $E = \{e_1, e_2, ..., e_n\}$ un ensemble fixé d'attributs, $\overline{E} = \mathcal{A} \setminus E = \{f_1, f_2, ..., f_m\}$ son complémentaire et O un ensemble d'objets fermé.

précédentes, avec l'aide de la table 3 et de la proposition 2 nous permettent de calculer la projection des contraintes syntaxiques, exceptées les contraintes utilisant les opérateurs d'agrégation MOY et SOM.

Dans cette table, on suppose que l'ensemble d'objets O est fermé. Cela n'est pas une restriction importante dans la mesure où nous ne nous intéressons qu'à des algorithmes d'extraction de fermés (ces fermés serviront à générer les concepts).

Examinons maintenant les contraintes utilisant les opérateurs d'agrégation MOY et SOM. Par définition, les contraintes projetées sont :$\text{MOY}(f(O))\,\theta\,\alpha$ et $\text{SOM}(f(O))\,\theta\,\alpha$. Il faut donc trouver une expression de $\text{MOY}(f(O))$ et $\text{SOM}(f(O))$ ne faisant plus intervenir f. En fait, il suffit d'étudier l'opérateur SOM car $\text{MOY}(f(O)) = \text{SOM}(f(O))/\,|f(O)| = \text{SOM}(f(O))/\mathsf{Freq}(O)$ donc si nous trouvons une expression de $\text{SOM}(f(O))$ dans la base projetée, nous obtiendrons aussi une expression pour $\text{MOY}(f(O))$.

L'ensemble $f(O)$ est un ensemble d'attribut, donc dans la matrice transposée, c'est un ensemble de lignes. Les valeurs $a.v$ sur lesquelles la somme est calculée sont attachées aux attributs a et donc aux lignes de la matrice transposée. La valeur $\text{SOM}(f(O))$ est donc la somme de ces valeurs v sur toutes les lignes de $f(O)$, c'est-à-dire les lignes contenant O. Autrement dit, $\text{SOM}(f(O))$ est une fréquence pondérée par les valeurs v (nous notons cette fréquence pondérée Freq_p). Celle-ci peut être facilement calculée par les algorithmes en plus de la fréquence "classique" Freq. Il suffit pour cela, lors de la passe sur les données, d'incrémenter cette fréquence pondérée de $a.v$ pour chaque ligne a contenant O.

Ces expressions de la contrainte projetée sont intéressantes car elles n'impliquent plus le calcul de $f(O)$ pour chaque ensemble devant être testé. Les ensembles $g(\overline{E})$ or $g(e_i)$ qui apparaissent dans ces contraintes peuvent quant à eux être calculés une fois pour toute lors de la première passe sur les données (en effet, l'ensemble E est constant).

Exemple 2
Dans cet exemple, nous montrons comment calculer la projection d'une contrainte complexe dans la base de données de la table 2. La contrainte est : $\mathcal{C}(A, O, bd) = (A \cap a_1a_4 \neq \emptyset)$. *Dans la table 2, l'itemset* $\overline{a_1a_4} = a_2a_3$ *est fermé. Par conséquent, la contrainte projetée est* $p(\mathcal{C})(O) = (g(a_2a_3) \not\subseteq O)$. *Comme* $g(a_2a_3) = o_1o_2o_3$, $p(\mathcal{C})(O) = (o_1o_2o_3 \not\subseteq O)$. *Les ensembles fermés d'objets qui satisfont cette contrainte sont* $T = \{\emptyset, o_1o_2, o_3\}$. *Nous pouvons ensuite calculer les concepts correspondants qui sont :* $(a_1a_2a_3a_4, \emptyset)$, $(a_1a_2a_3, o_1o_2)$ *et* $(a_2a_3a_4, o_3)$.

4 Utilisation de la projection de contraintes

Dans cette section, nous présentons deux stratégies pour extraire les concepts satisfaisant une contrainte $\mathcal{C}$ et ainsi résoudre le problème posé dans la section 2.3.

La première stratégie utilise les algorithmes classiques d'extraction de fermés :

1. Calculer la contrainte projetée $p(\mathcal{C})$ de $\mathcal{C}$ en utilisant la table 3.2 et la propriété 2 ;

2. Utiliser un algorithme pour l'extraction de fermés sous contraintes dans la matrice transposée (comme par exemple, ceux proposés dans (Bonchi & Lucchese, 2004) ou (Boulicaut & Jeudy, 2001)) avec la contrainte $p(\mathcal{C})$. Il est aussi possible d'utiliser des algorithmes d'extraction de fermés fréquent tels que CHARM (Zaki & Hsiao, 2002), CARPENTER (Pan *et al.*, 2003) ou CLOSET (Pei *et al.*, 2000) en leur rajoutant une étape d'élagage supplémentaire pour traiter la contrainte (à la manière de ce qui est fait dans (Pei & Han, 2000).
3. Ces algorithmes extraient des ensembles fermés. Cela signifie qu'ils vont retourner les ensembles d'objets fermés (car nous travaillons dans la matrice transposée) qui satisfont la contrainte $p(\mathcal{C})$. Il faut alors pour chacun de ces fermés calculer son intension $f(O)$, d'après la proposition 1, les paires $(f(O), O)$ ainsi formées seront exactement les concepts qui satisfont la contrainte $\mathcal{C}$. Le calcul de $f(O)$ peut être fait lors d'une dernière passe sur les données ou alors intégré dans les algorithmes. En fait, ces algorithmes calculent ces intensions lors du calcul de la fréquence des ensembles (la fréquence de O est $|f(O)|$). Il suffit donc de les modifier pour qu'ils stockent ces intensions.

La seconde stratégie est basée sur le nouvel algorithme D-Miner (Besson *et al.*, 2004). Cet algorithme extrait des concepts sous une contrainte $\mathcal{C}$ qui est la conjonction d'une contrainte monotone sur les attributs et d'une contrainte monotone sur les objets. Il ne peut cependant pas traiter le cas où des contraintes anti-monotones sont utilisées.

Notre stratégie consiste alors à projeter les contraintes anti-monotones définies dans l'espace des attributs sur l'espace des objets et à projeter les contraintes anti-monotones définies dans l'espace des objets sur l'espace des attributs. En effet, d'après la proposition 3, la projection transforme une contrainte anti-monotone en une contrainte monotone. Cela permet donc d'utiliser D-Miner avec des contraintes monotones et anti-monotones. Nous n'avons présenté que la projection des contraintes de l'espace des attributs sur l'espace des objets. Cependant, la projection dans l'autre sens est similaire. En fait, il suffit de remplacer la fonction f par la fonction g.

5 Conclusion

L'analyse des données d'expression de gènes pose un problème spécifique pour l'extraction de motifs : les données contiennent beaucoup plus de colonnes que de lignes ce qui rend les algorithmes d'extraction classiques inopérants. Dans ce cas, extraire les motifs dans la matrice transposée permet de s'affranchir de ce problème.

La transposition a déjà été étudié dans le cas de la contrainte de fréquence, mais l'étude générale de ce qui se passe dans le cas d'une contrainte complexe restait à faire. Cette étude nous a permis de proposer des stratégies pour extraire des concepts sous contraintes. Ces stratégies, plutôt que de proposer un nouvel

algorithme, se fondent sur l'utilisation d'algorithmes classiques et éprouvés d'extraction de fermés ou de concepts. Afin de rendre leur utilisation possible, nous avons défini une opération de projection des contraintes et nous avons étudié ses propriétés ainsi que les projections de contraintes classiques.

Références

Agrawal R., Mannila H., Srikant R., Toivonen H. & Verkamo A. I. (1996). Fast discovery of association rules. In U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth & R. Uthurusamy, Eds., *Advances in Knowledge Discovery and Data Mining*, p. 307–328. Menlo Park : AAAI Press.

Albert-Lorincz H. & Boulicaut J.-F. (2003). Mining frequent sequential patterns under regular expressions : a highly adaptative strategy for pushing constraints. In *Third SIAM International Conference on Data Mining (SIAM DM'03)*, p. 316–320.

Besson J., Robardet C. & Boulicaut J.-F. (2004). Constraint-based mining of formal concepts in transactional data. In H. Dai, R. Srikant & C. Zhang, Eds., *Proceedings of the 8th Pacif-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'04)*, volume 3056 of *Lecture Notes in Computer Science*, p. 615–624, Sydney, Australia.

Bonchi F., Giannotti F., Mazzanti A. & Pedreschi D. (2003). Exante : Anticipated data reduction in constrained pattern mining. In *Proceedings of the 7th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'03)*, volume 2838 of *Lecture Notes in Artificial Intelligence*, Cavtat-Dubrovnik, Croatia.

Bonchi F. & Lucchese C. (2004). On closed constrained frequent pattern mining. In *Proceedings of the Fourth IEEE International Conference on Data Mining (ICDM'04)*, Brighton, UK.

Boulicaut J.-F., Bykowski A. & Rigotti C. (2000). Approximation of frequency queries by mean of free-sets. In D. Zighed, J. Komorowski & J. M. Zytkow, Eds., *Proceedings of the 4th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'00)*, volume 1910 of *Lecture Notes in Artificial Intelligence*, p. 75–85, Lyon, France : Berlin : Springer-Verlag.

Boulicaut J.-F., Bykowski A. & Rigotti C. (2003). Free-sets : a condensed representation of boolean data for the approximation of frequency queries. *Data Mining and Knowledge Discovery*, **7**(1), 5–22.

Boulicaut J.-F. & Jeudy B. (2000). Using constraint for itemset mining : should we prune or not ? In A. Doucet, Ed., *Actes des 16e Journées Bases de Données Avancées (BDA'00)*, p. 221–237, Blois, France : Université de Tours.

Boulicaut J.-F. & Jeudy B. (2001). Mining free-sets under constraints. In M. E. Adiba, C. Collet & B. C. Desai, Eds., *Proceedings of the International Database Engineering & Applications Symposium (IDEAS'01)*, p. 322–329, Grenoble, France : IEEE Computer Society.

Bucila C., Gehrke J. E., Kifer D. & White W. (2003). Dualminer : A dual-pruning algorithm for itemsets with constraints. *Data Mining and Knowledge Discovery*, **7**(4), 241–272.

Garofalakis M. M., Rastogi R. & Shim K. (1999). SPIRIT : Sequential pattern mining with regular expression constraints. In M. P. Atkinson & others, Eds., *Proceedings of the 25nd International Conference on Very Large Data Bases (VLDB'99)*, p. 223–234, Edinburgh, UK : San Francisco : Morgan Kaufmann.

Ng R., Lakshmanan L. V., Han J. & Pang A. (1998). Exploratory mining and pruning optimizations of constrained associations rules. In L. M. Haas & A. Tiwary, Eds., *Proceedings of ACM SIGMOD Conference on Management of Data (SIGMOD'98)*, volume 27(2) of *SIGMOD Record*, p. 13–24, Seattle, Washington, USA : New York : ACM Press.

Nguifo E. M. & Njiwoua P. (2000). GLUE : a lattice-based constructive induction system. *Intelligent Data Analysis*, **4**(4), 1–49.

Pan F., Cong G., Tung A. K. H., Yang J. & Zaki M. J. (2003). CARPENTER : Finding closed patterns in long biological datasets. In *Proceedings of the 9th International Conference on Knowledge Discovery and Data Mining (KDD'03)*, Washington DC : New York : ACM Press.

Pasquier N., Bastide Y., Taouil R. & Lakhal L. (1999). Efficient mining of association rules using closed itemset lattices. *Information Systems*, **24**(1), 25–46.

Pei J. & Han J. (2000). Can we push more constraints into frequent pattern mining ? In *Proceedings of the 6th International Conference on Knowledge Discovery and Data Mining (KDD'00)*, p. 350–354, Boston, USA : New York : ACM Press.

Pei J., Han J. & Mao R. (2000). CLOSET an efficient algorithm for mining frequent closed itemsets. In D. Gunopulos & R. Rastogi, Eds., *Proceedings of the ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD'00)*, Dallas, Texas, USA. Disp. en ligne (sept. 2002) `http://www.cs.ucr.edu/~dg/DMKD.html`, 10 pages.

Rioult F., Boulicaut J.-F., Crémilleux B. & Besson J. (2003). Using transposition for pattern discovery from microarray data. In *8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, p. 73–79, San Diego, USA.

Rioult F. & Crémilleux B. (2004). Optimisation of pattern mining : a new method founded on database transposition. In *EIS'04*.

Srikant R., Vu Q. & Agrawal R. (1997). Mining association rules with item constraints. In D. Heckerman, H. Mannila, D. Pregibon & R. Uthurusamy, Eds., *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining (KDD'97)*, p. 67–73, Newport Beach, California, USA : Menlo Park : AAAI Press.

Wille R. (1992). Concept lattices and conceptual knowledge systems. *Computer mathematic applied*, **23**((6-9)), 493–515.

Zaki M. J. (2000). Sequence mining in categorical domains : incorporating constraints. In *Proceedings of the 9th ACM International Conference on Information and Knowledge Management (CIKM'00)*, p. 422–429, Washington DC, USA : New York : ACM Press.

Zaki M. J. & Hsiao C.-J. (2002). CHARM : An efficient algorithm for closed itemset mining. In R. Grossman, J. Han, V. Kumar, H. Mannila & R. Motwani, Eds., *2nd International SIAM Conference on Data Mining (SDM02)*, Arlington, USA. Disp. en ligne (sept. 2002) `http://www.siam.org/meetings/sdm02/`, 17 pages.

Semi-supervised Learning by Entropy Minimization *

Yves Grandvalet[1], Yoshua Bengio[2]

[1] Heudiasyc, UMR 6599 CNRS/UTC
60205 Compiègne cedex, France
grandval@utc.fr

[2] Dept. IRO, Université de Montréal
Montreal, Qc, H3C 3J7, Canada
bengioy@iro.umontreal.ca

Abstract : We consider the semi-supervised learning problem, where a decision rule is to be learned from labeled and unlabeled data. In this framework, we motivate minimum entropy regularization, which enables to incorporate unlabeled data in the standard supervised learning. This regularizer can be applied to any model of posterior probabilities. Our approach provides a new motivation for some existing semi-supervised learning algorithms which are particular or limiting instances of minimum entropy regularization. A series of experiments illustrates that the proposed solution benefits from unlabeled data. The method challenges mixture models when the data are sampled from the distribution class spanned by the generative model. The performances are definitely in favor of minimum entropy regularization when generative models are misspecified, and the weighting of unlabeled data provides robustness to the violation of the "cluster assumption". Finally, we also illustrate that the method can be far superior to manifold learning in high dimension spaces, and also when the manifolds are generated by moving examples along the discriminating directions.

Résumé : Nous considérons le problème de l'apprentissage semi-supervisé, où une règle de décision est induite sur la base d'exemples étiquetés et non-étiquetés. Dans ce cadre, nous motivons l'utilisation de la régularisation par minimum d'entropie, qui permet d'utiliser les données non-étiquetées dans l'apprentissage de modèles discriminants. Cette technique peut être appliquée à tout modèle discriminant estimant des probabilités *a posteriori*. Notre approche fournit un nouveau point de vue sur certains algorithmes d'apprentissage semi-supervisé existants, qui peuvent être interprétés comme cas particulier ou limite de la régularisation par minimum d'entropie. Une série d'expérience illustre que la solution proposée permet de bénéficier de données non-étiquetées. Les résultats

*This work was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence IST-2002-506778. This publication only reflects the authors' views.

concurrencent ceux des modèles génératifs quand ces derniers utilisent le modèle correct, correspondant à celui de la distribution des données. Les performances sont clairement en faveur de la régularisation par minimum d'entropie quand le modèle devient incorrect, et la pondération des exemples non-étiquetés permet d'atteindre des solutions robustes aux violations de l'hypothèse de départ, qui postule que les classes sont bien séparées. Finalement, nous illustrons que la méthode peut également être de loin supérieure aux techniques récentes d'apprentissage de variété, que ce soit dans les espaces de grande dimension, ou quand les variétés sont produites par des exemples transformés sur les directions discriminantes.

Keywords : semi-supervised learning, minimum entropy, transduction, self-training, EM algorithm, spectral methods, logistic regression.

1 Introduction

In the classical supervised learning classification framework, a decision rule is to be learned from a learning set $\mathcal{L}_n = \{\mathbf{x}_i, y_i\}_{i=1}^n$, where each example is described by a pattern $\mathbf{x}_i \in \mathcal{X}$ and by the supervisor's response $y_i \in \Omega = \{\omega_1, \ldots, \omega_K\}$. Here, we consider semi-supervised learning, where the supervisor's responses are limited to a subset of $\mathcal{L}_n$.

In the terminology used here, semi-supervised learning refers to learning a decision rule on $\mathcal{X}$ from labeled and unlabeled data. However, the related problem of transductive learning, that is, of predicting labels on a set of predefined patterns, is addressed as a side issue. Semi-supervised problems occur in many applications where labeling is performed by human experts. They have been receiving much attention during the last few years, but some important issues are unresolved (Seeger, 2002).

In the probabilistic framework, semi-supervised learning can be modeled as a missing data problem, which can be addressed by generative models such as mixture models thanks to the EM algorithm and extensions thereof (McLachlan, 1992). Generative models apply to the joint density of patterns and class (X, Y). They have appealing features, but they also have major drawbacks. Their estimation is much more demanding than discriminative models, since the model of $P(X, Y)$ is exhaustive, hence necessarily more complex than the model of $P(Y|X)$. More parameters are to be estimated, resulting in more uncertainty in the estimation process. The generative model being more precise, it is also more likely to be misspecified. Finally, the fitness measure is not discriminative, so that better models are not necessarily better predictors of class labels. These difficulties have lead to proposals where unlabeled data are processed in supervised classification algorithms (Bennett & Demiriz, 1999; Joachims, 1999; Amini & Gallinari, 2002; Grandvalet, 2002; Szummer & Jaakkola, 2003). Here, we propose an estimation principle applicable to any probabilistic classifier, aiming at making the most of unlabeled data when they should be beneficial, while controling their contribution to provide robustness to the learning scheme.

2 Derivation of the Criterion

2.1 Likelihood

The maximum likelihood principle is one of the main estimation technique in supervised learning, which is closely related to the more recent margin maximization techniques such as boosting and support vector machines (Friedman *et al.*, 2000). We start here by looking at the contribution of unlabeled examples to the (conditional) likelihood.

The learning set is denoted $\mathcal{L}_n = \{\mathbf{x}_i, \mathbf{z}_i\}_{i=1}^n$, where $\mathbf{z}_i \in \{0,1\}^K$ denotes the dummy variable representing the actually available labels (while y represents the precise and complete class information): if $\mathbf{x}_i$ is labeled ω_k, then $z_{ik} = 1$ and $z_{i\ell} = 0$ for $\ell \neq k$; if $\mathbf{x}_i$ is unlabeled, then $z_{i\ell} = 1$ for $\ell = 1, \ldots, K$. More generally, $\mathbf{z}$ should be thought as the index of possible labels. Hence, an imprecise knowledge, such that "$\mathbf{x}_i$ belongs either to class ω_1 or ω_2" can be encoded by letting z_{i1} and z_{i2} to be one, and $\mathbf{z}_{ik} = 0$ for $k > 2$.

We assume that labels are missing at random, that is, the missingness mechanism is independent from the missing information: the label is missing because it is not observed, not because it is deliberately hidden. In the general setup where $\mathbf{z}$ can indicate any subset of Ω, some information is missing when two or more labels are possible. The missing-at-random assumption reads $P(\mathbf{z}|\mathbf{x}, Y = \omega_k) = P(\mathbf{z}|\mathbf{x}, Y = \omega_\ell)$ for all (k, ℓ) such that $z_k = z_\ell = 1$.

Assuming independent examples, and noting that the event "y_i belongs to the subset indicated by $\mathbf{z}_i$" follows a Bernoulli distribution of parameter $\sum_{k=1}^K z_{ik} P(Y = \omega_k|\mathbf{x}_i)$, the conditional log-likelihood of $(Z|X)$ on the observed sample is then

$$L(\boldsymbol{\theta}; \mathcal{L}_n) = \sum_{i=1}^n \log \left(\sum_{k=1}^K z_{ik} f_k(\mathbf{x}_i; \boldsymbol{\theta}) \right) + h(\mathbf{z}_i) \ , \tag{1}$$

where $h(\mathbf{z})$, which does not depend on $P(X, Y)$, is only affected by the missingness mechanism, and $f_k(\mathbf{x}; \boldsymbol{\theta})$ is the model of $P(Y = \omega_k|\mathbf{x})$ parameterized by $\boldsymbol{\theta}$.

This criterion is a concave function of $f_k(\mathbf{x}_i; \boldsymbol{\theta})$, and for simple models such as the ones provided by logistic regression, the semi-supervised objective function is also concave in $\boldsymbol{\theta}$, so that the global solution can be obtained by numerical optimization. Maximizing (1) corresponds to maximizing the complete likelihood if no assumption whatsoever is made on $P(X)$ (McLachlan, 1992).

Provided $f_k(\mathbf{x}_i; \boldsymbol{\theta})$ sum to one, the likelihood is not affected by unlabeled data: unlabeled data convey no information. In the maximum a posteriori (MAP) framework, Seeger (2002) remarks that unlabeled data are useless regarding discrimination when the priors on $P(X)$ and $P(Y|X)$ factorize: observing $\mathbf{x}$ does not inform about y, unless the modeler assumes so. Benefitting from unlabeled data requires assumptions of some sort on the relationship between X and Y. In the MAP framework, this will be encoded by a prior distribution. As there is no such thing like a universally relevant prior, we should look for an induction bias allowing to process unlabeled data when the latter is known to convey information.

2.2 When Are Unlabeled Examples Informative?

Theory provides little support to the numerous experimental evidences, such as (Joachims, 1999; Nigam & Ghani, 2000; Nigam *et al.*, 2000), showing that unlabeled examples can help the learning process. Learning theory is mostly developed at the two extremes of the statistical paradigm: in parametric statistics where examples are known to be generated from a known class of distribution, and in the distribution-free Structural Risk Minimization (SRM) or Probably Approximately Correct (PAC) frameworks. Semi-supervised learning, in the terminology used here, does not fit the distribution-free frameworks: no positive statement can be made without distributional assumptions, as for some distributions $P(X, Y)$ unlabeled data are non-informative while supervised learning is an easy task. In this regard, generalizing from labeled and unlabeled data may differ from transductive inference.

In parametric statistics, theory has shown the benefit of unlabeled examples, either for specific distributions (O'Neill, 1978), or for mixtures of the form $P(\mathbf{x}) = pP(\mathbf{x}|Y = \omega_1) + (1 - p)P(\mathbf{x}|Y = \omega_2)$ where the estimation problem is essentially reduced to the one of estimating the mixture parameter p (Castelli & Cover, 1996). These studies conclude that the (asymptotic) information content of unlabeled examples decreases as classes overlap.[1] Thus, the assumption that classes are well apart, separated by a low density area, is sensible if we expect to take advantage of unlabeled examples.

2.3 A Measure of Class Overlap

The conditional entropy $H(Y|X)$ is a measure of class overlap, which is invariant to the parameterization of the model. The entropy may be related to the usefulness of unlabeled data only where labeling is indeed ambiguous. Hence, we propose to measure the conditional entropy of class labels conditioned on the observed variables

$$H(Y|X, Z) = -E_{XYZ}[\log P(Y|X, Z)] \ , \tag{2}$$

where E_X denotes the expectation with respect to X.

In the MAP framework, assumptions are encoded by means of a prior on the model parameters. Stating that we expect a high conditional entropy does not uniquely define the form of the prior distribution, but the latter can be derived by resorting to the maximum entropy principle.[2] Let $(\boldsymbol{\theta}, \boldsymbol{\psi})$ denote the model parameters of $P(X, Y, Z)$; the maximum entropy prior verifying $E_{\Theta\Psi}[H(Y|X, Z)] = c$, where the constant c quantifies how small the entropy should be on average, takes the form

$$P(\boldsymbol{\theta}, \boldsymbol{\psi}) \propto \exp\left(-\lambda H(Y|X, Z)\right) \ , \tag{3}$$

where λ is the positive Lagrange multiplier corresponding to the constant c.

[1] This statement, given explicitly by O'Neill (1978), is also formalized, though not stressed, by Castelli & Cover (1996), where the Fisher information for unlabeled examples at the estimate $\hat{p}$ is clearly a measure of the overlap between class conditional densities: $I_u(\hat{p}) = \int \frac{(P(\mathbf{x}|Y=\omega_1) - P(\mathbf{x}|Y=\omega_2))^2}{\hat{p}P(\mathbf{x}|Y=\omega_1) + (1-\hat{p})P(\mathbf{x}|Y=\omega_2)} \, d\mathbf{x}$.

[2] Here, maximum entropy refers to the construction principle which enables to derive distributions from constraints, not to the content of priors regarding entropy.

Computing $H(Y|X,Z)$ requires a model of $P(X,Y,Z)$ whereas the choice of supervised classification is motivated by the possibility to limit modeling to conditional probabilities. We circumvent the need of additional modeling by applying the plug-in principle, which consists in replacing the expectation with respect to (X,Z) by the sample average. This substitution, which can be interpreted as "modeling" $P(X,Z)$ by its empirical distribution, yields

$$H_{\text{emp}}(Y|X,Z;\mathcal{L}_n) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K} P(Y=\omega_k|\mathbf{x}_i,\mathbf{z}_i)\log P(Y=\omega_k|\mathbf{x}_i,\mathbf{z}_i) \quad . \quad (4)$$

This empirical functional is plugged in (3) to define an empirical prior on parameters $\boldsymbol{\theta}$, that is, a prior whose form is partly defined from data (Berger, 1985).

2.4 Entropy Regularization

As detailed in appendix A, the missing-at-random assumption implies

$$P(Y=\omega_k|\mathbf{x},\mathbf{z}) = \frac{z_k P(Y=\omega_k|\mathbf{x})}{\sum_{\ell=1}^{K} z_\ell P(Y=\omega_\ell|\mathbf{x})} \quad . \quad (5)$$

Recalling that $f_k(\mathbf{x};\boldsymbol{\theta})$ denotes the model of $P(Y=\omega_k|\mathbf{x})$, the model of $P(Y=\omega_k|\mathbf{x},\mathbf{z})$ is defined as follows:

$$g_k(\mathbf{x},\mathbf{z};\boldsymbol{\theta}) = \frac{z_k f_k(\mathbf{x};\boldsymbol{\theta})}{\sum_{\ell=1}^{K} z_\ell f_\ell(\mathbf{x};\boldsymbol{\theta})} \quad .$$

For labeled data, $g_k(\mathbf{x},\mathbf{z};\boldsymbol{\theta}) = z_k$, and for unlabeled data, $g_k(\mathbf{x},\mathbf{z};\boldsymbol{\theta}) = f_k(\mathbf{x};\boldsymbol{\theta})$.

From now on, we drop the reference to parameter $\boldsymbol{\theta}$ in f_k and g_k to lighten notation. The MAP estimate is defined as the maximizer of the posterior distribution, that is, the maximizer of

$$\begin{aligned} C(\boldsymbol{\theta},\lambda;\mathcal{L}_n) &= L(\boldsymbol{\theta};\mathcal{L}_n) - \lambda H_{\text{emp}}(Y|X,Z;\mathcal{L}_n) \\ &= \sum_{i=1}^{n}\log\left(\sum_{k=1}^{K} z_{ik} f_k(\mathbf{x}_i)\right) + \lambda\sum_{i=1}^{n}\sum_{k=1}^{K} g_k(\mathbf{x}_i,\mathbf{z}_i)\log g_k(\mathbf{x}_i,\mathbf{z}_i) \quad , (6) \end{aligned}$$

where the constant terms in the log-likelihood (1) and log-prior (3) have been dropped. While $L(\boldsymbol{\theta};\mathcal{L}_n)$ is only sensitive to labeled data, $H_{\text{emp}}(Y|X,Z;\mathcal{L}_n)$ is only affected by the value of $f_k(\mathbf{x})$ on unlabeled data.

Note that the empirical approximation H_{emp} (4) of H (2) breaks down for wiggly functions $f_k(\cdot)$ with abrupt changes between data points (where $P(X)$ is bounded from below). As a result, it is important to constrain $f_k(\cdot)$ in order to enforce the closeness of the two functionals. In the following experimental section, we imposed such a constraint on $f_k(\cdot)$ by adding to the criterion C (6) a smoothness penalty.

3 Related Work

3.1 Minimum Entropy in Pattern Recognition

Minimum entropy regularizers have been used in other contexts to encode learnability priors (Brand, 1999). In a sense, H_{emp} can be seen as a poor's man way to generalize this approach to continuous input spaces. This empirical functional was also used by Zhu *et al.* (2003) as a criterion to learn weight function parameters in the context of transduction in manifold learning.

3.2 Input-Dependent and Information Regularization

Input-dependent regularization aims at incorporating some knowledge about the density $P(X)$ in the modeling of $P(Y|X)$. In the framework of Bayesian inference, Seeger (2002) proposes to encode this knowledge by structural dependencies in the prior distributions. Information regularization, proposed by Szummer & Jaakkola (2003) and later developped by Corduneanu & Jaakkola (2003), is another approach where the density $P(X)$ is assumed to be known, and where the mutual information between variables X and Y is supposed to be low within predefined neighborhoods.

Entropy regularization differs from input-dependent regularization in that it is expressed only in terms of $P(Y|X,Z)$ and does not involve a model of $P(X)$. However, we stress that for unlabeled data, the MAP estimation is consistent with the maximum (complete) likelihood approach when $P(X)$ is small near the decision surface. Indeed, whereas the complete likelihood maximizes $\log P(X)$ on unlabeled data, the regularizer minimizes the conditional entropy on the same points. Hence, the two criteria agree provided the class assignments are confident in high density regions, or conversely, when label switching occurs in a low density area.

3.3 Self-Training

Self-training (Nigam & Ghani, 2000) is an iterative process, where a learner imputes the labels of examples which have been classified with confidence in the previous step. Amini & Gallinari (2002) analyzed this technique and have shown that it is equivalent to a version of the classification EM algorithm, which minimizes the likelihood deprived of the entropy of the partition. In the context of conditional likelihood with labeled and unlabeled examples, the criterion is

$$\sum_{i=1}^{n} \log \left(\sum_{k=1}^{K} z_{ik} f_k(\mathbf{x}_i) \right) + \sum_{k=1}^{K} g_k(\mathbf{x}_i) \log g_k(\mathbf{x}_i) \quad ,$$

which is recognized as an instance of the criterion (6) with $\lambda = 1$.

Self-confident logistic regression (Grandvalet, 2002), also proposed by Jin & Ghahramani (2003) as "the EM model", is another algorithm optimizing the criterion for $\lambda = 1$. Using smaller λ values is expected to have two benefits. First, the influence of unlabeled examples can be controlled, in the spirit of EM-λ (Nigam *et al.*, 2000). Second, slowly increasing λ defines a scheme similar to the increase of the C^* parameter in the

transductive SVM algorithm of Joachims (1999). These schemes are somewhat similar to the deterministic annealing procedures, used for example in clustering (Rose *et al.*, 1990). They are expected to help the optimization process to avoid poor local minima of the criterion.

3.4 Maximal Margin Separators

Maximal margin separators are theoretically well founded models which have shown great success in supervised classification. For linearly separable data, they have been shown to be a limiting case of probabilistic hyperplane separators (Tong & Koller, 2000).

In the framework of transductive learning, Vapnik (1998) proposed to broaden the margin definition to unlabeled examples, by taking the smallest Euclidean distance between any (labeled and unlabeled) training point to the classification boundary. The following theorem, whose proof is given in Appendix B, generalizes Theorem 5, Corollary 6 of Tong & Koller (2000) to the margin defined in transductive learning when using the proposed minimum entropy criterion.

Theorem 1
In the two-class linear separable case, the logistic regression model with bounded weights, fitted by the minimum entropy criterion, converges towards the maximum margin separator (with maximal distance from labeled and unlabeled examples) as the bound goes to infinity.

Hence, the minimum entropy solution can closely mimic semi-supervised SVM (Bennett & Demiriz, 1999), which partially solves the enumeration problem of the original solution proposed by Vapnik (1998).

Note however that our criterion is not concave in f_k, so that the convergence toward the global maximum cannot be guaranteed. To our knowledge, this apparent fault is shared by all inductive semi-supervised algorithms (learning a decision rule) dealing with a large number of unlabeled data in reasonable time, such as mixture models or the transductive SVM of Joachims (1999): explicitly or implicitly, inductive semi-supervised algorithms impute labels which are consistent with a decision rule. The enumeration of all possible configurations is only avoided thanks to an heuristic process which may fail. Most transduction algorithms avoid this enumeration problem because their labeling process is not required to comply with a parameterized decision rule. This clear computational advantage has however its counterpart: label propagation is performed via a predefined, hence non-discriminant, similarity measure. The experimental section below demonstrates that this may be a serious shortcomming in high dimensional spaces, or when *a priori* similar patterns should be discriminated.

4 Experiments

4.1 Artificial Data

In this section, we chose a simple experimental setup in order to avoid artifacts stemming from optimization problems. This setting enables to check to what extent supervised learning can be improved by unlabeled examples, and if minimum entropy can compete with generative methods which are usually advocated in this framework.

The minimum entropy criterion is applied to the logistic regression model. It is compared to logistic regression fitted by maximum likelihood (ignoring unlabeled data) and logistic regression with all labels known. The former shows what has been gained by handling unlabeled data, and the latter provides the "crystal ball" ultimate performance obtained by guessing correctly all labels. All hyper-parameters (weight-decay for all logistic regression models plus the λ parameter (6) for minimum entropy) are tuned by ten-fold cross-validation.

Minimum entropy logistic regression is also compared to the classic EM algorithm for Gaussian mixture models (two means and one common covariance matrix estimated by maximum likelihood on labeled and unlabeled examples (McLachlan, 1992). Bad local maxima of the likelihood function are avoided by initializing EM with the parameters of the true distribution when the latter is a Gaussian mixture, or with maximum likelihood parameters on the (fully labeled) test sample when the distribution departs from the model. This initialization advantages EM, since it is guaranteed to pick, among all local maxima of the likelihood, the one which is in the basin of attraction of the optimal value. Furthermore, this initialization prevents interferences that may result from the "pseudo-labels" given to unlabeled examples at the first E-step. In particular, "label switching" (badly labeled clusters) is avoided at this stage.

4.1.1 Correct joint density model

In the first series of experiments, we consider two-class problems in a 50-dimensional input space. Each class is generated with equal probability from a normal distribution. Class ω_1 is normal with mean $(aa \dots a)$ and unit covariance matrix. Class ω_2 is normal with mean $-(aa \dots a)$ and unit covariance matrix. Parameter a tunes the Bayes error which varies from 1 % to 20 % (1 %, 2.5 %, 5 %, 10 %, 20 %). The learning sets comprise n_l labeled examples, ($n_l = 50, 100, 200$) and n_u unlabeled examples, ($n_u = n_l \times (1, 3, 10, 30, 100)$). Overall, 75 different setups are evaluated, and for each one, 10 different training samples are generated. Generalization performances are estimated on a test set of size $10\,000$.

This benchmark provides a comparison for the algorithms in a situation where unlabeled data are known to convey information. Besides the favorable initialization of the EM algorithm to the optimal parameters, EM benefits from the *correctness* of the model: data were generated according to the model, that is, two Gaussian subpopulations with identical covariances. The logistic regression model is only *compatible* with the joint distribution, which is a weaker fulfillment than correctness.

As there is no modeling bias, differences in error rates are only due to differences in estimation efficiency. The overall error rates (averaged over all settings) are in favor of

minimum entropy logistic regression (14.1 ± 0.3 %). EM (15.7 ± 0.3 %) does worse on average than logistic regression (14.9 ± 0.3 %). For reference, the average Bayes error rate is 7.7 % and logistic regression reaches 10.4 ± 0.1 % when all examples are labeled.

Figure 1 provides more informative summaries than these raw numbers. The plots represent the error rates (averaged over n_l) versus Bayes error rate and the n_u/n_l ratio. The first plot shows that, as asymptotic theory suggests (O'Neill, 1978; Castelli & Cover, 1996), unlabeled examples are more beneficial when the Bayes error is low. This observation supports the relevance of the minimum entropy assumption.

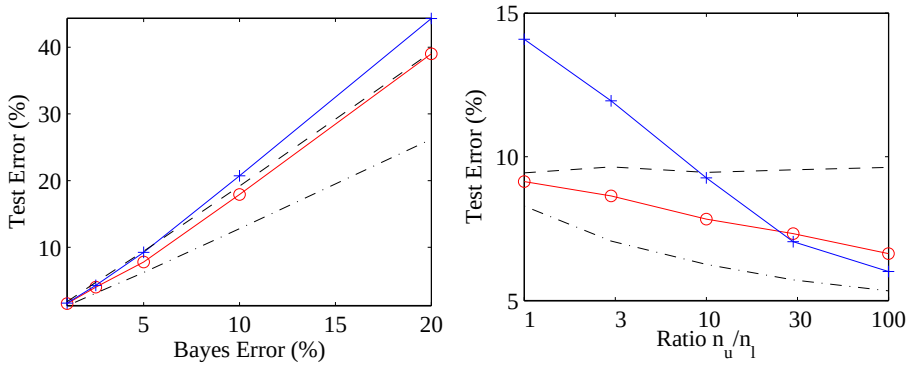


Figure 1: Left: test error *vs.* Bayes error rate for $n_u/n_l = 10$; right: test error *vs.* n_u/n_l ratio for 5 % Bayes error ($a = 0.23$). Test errors of minimum entropy logistic regression ($\circ$) and mixture models ($+$). The errors of logistic regression (dashed), and logistic regression with all labels known (dash-dotted) are shown for reference.

The second plot illustrates that the minimum entropy model takes quickly advantage of unlabeled data when classes are well separated. With $n_u = 3n_l$, the model considerably improves upon the one discarding unlabeled data. This graph also illustrates the consequence of the demanding parametrization of generative models. For very large sample sizes, with 100 times more unlabeled examples than labeled examples, the generative approach eventually becomes more accurate than the diagnosis approach. However, mixture models are outperformed by the simple logistic regression model when the sample size is low, because their number of parameters is quadratic (*vs.* linear) in the number of input features.

4.1.2 Misspecified joint density model

In a second series of experiments, the setup is slightly modified by letting the class-conditional densities be corrupted by outliers. For each class, the examples are generated from a mixture of two Gaussians centered on the same mean: a unit variance component gathers 98 % of examples, while the remaining 2 % are generated from a large variance component, where each variable has a standard deviation of 10. The

mixture model used by EM is now slightly misspecified since the whole distribution is still modeled by a simple two-components Gaussian mixture. The results, displayed in the left-hand-side of Figure 2, should be compared with the right-hand-side of Figure 1. The generative model dramatically suffers from the misspecification and behaves worse than logistic regression for all sample sizes. The unlabeled examples have first a beneficial effect on test error, then have a detrimental effect when they overwhelm the number of labeled examples. On the other hand, the diagnosis models behave smoothly as in the previous case, and the minimum entropy criterion performance improves.

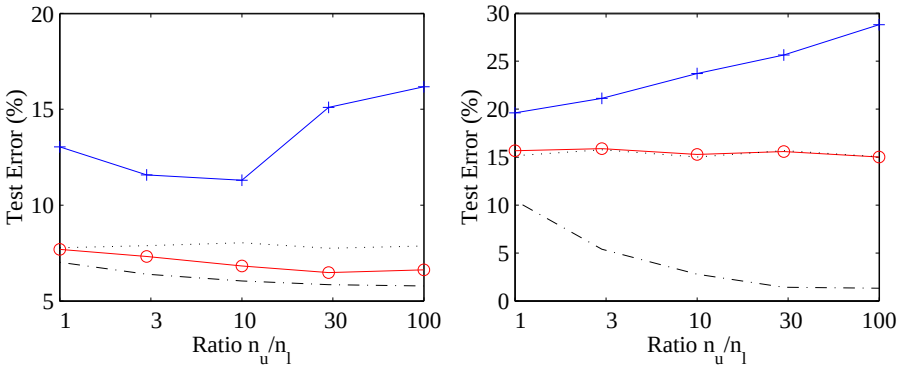


Figure 2: Test error *vs.* n_u/n_l ratio for $a = 0.23$. Average test errors for minimum entropy logistic regression ($\circ$) and mixture models ($+$). The test error rates of logistic regression (dotted), and logistic regression with all labels known (dash-dotted) are shown for reference. Left: experiment with outliers; right: experiment with uninformative unlabeled data.

The last series of experiments illustrate the robustness with respect to the cluster assumption, by testing it on distributions where unlabeled examples are not informative, and where a low density $P(X)$ does not indicate a boundary region. The data is drawn from two Gaussian clusters like in the first series of experiment, but the label is now independent of the clustering: an example $\mathbf{x}$ belongs to class ω_1 if $x_2 > x_1$ and belongs to class ω_2 otherwise: the Bayes decision boundary is now separates each cluster in its middle. The mixture model is unchanged. It is now far from the model used to generate data. The right-hand-side plot of Figure 1 shows that the favorable initialization of EM does not prevent the model to be fooled by unlabeled data: its test error steadily increases with the amount of unlabeled data. On the other hand, the diagnosis models behave well, and the minimum entropy algorithm is not distracted by the two clusters; its performance is nearly identical to the one of training with labeled data only (cross-validation provides λ values close to zero), which can be regarded as the ultimate performance in this situation.

4.1.3 Comparison with manifold transduction

Although our primary goal is to infer a decision function, we also provide comparisons with a transduction algorithm of the "manifold family". We chose the consistency method of Zhou *et al.* (2004) for its simplicity. As suggested by the authors, we set $\alpha = 0.99$ and the scale parameter σ^2 was optimized on test results (Zhou *et al.*, 2004). The results are reported in Table 1. The experiments are limited due to the memory requirements of the consistency method in our naive MATLAB implementation.

Table 1: Error rates (%) of minimum entropy (ME) *vs.* consistency method (CM), for $a = 0.23$, $n_l = 50$, and a) pure Gaussian clusters b) Gaussian clusters corrupted by outliers c) class boundary passing through the Gaussian clusters.

n_u	50	150	500	1500
a) ME	10.8 ± 1.5	9.8 ± 1.9	8.8 ± 2.0	8.3 ± 2.6
a) CM	21.4 ± 7.2	25.5 ± 8.1	29.6 ± 9.0	26.8 ± 7.2
b) ME	8.5 ± 0.9	8.3 ± 1.5	7.5 ± 1.5	6.6 ± 1.5
b) CM	22.0 ± 6.7	25.6 ± 7.4	29.8 ± 9.7	27.7 ± 6.8
c) ME	8.7 ± 0.8	8.3 ± 1.1	7.2 ± 1.0	7.2 ± 1.7
c) CM	51.6 ± 7.9	50.5 ± 4.0	49.3 ± 2.6	50.2 ± 2.2

The results are extremely poor for the consistency method, whose error is way above minimum entropy, and which does not show any sign of improvement as the sample of unlabeled data grows. Furthermore, when classes do not correspond to clusters, the consistency method performs random class assignments. In fact, our setup, which was designed for the comparison of global classifiers, is extremely defavorable to manifold methods, since the data is truly 50-dimensional. In this situation, local methods suffer from the "curse of dimensionality", and many more unlabeled examples would be required to get sensible results. Hence, these results mainly illustrate that manifold learning is not the best choice in semi-supervised learning for truly high dimensional data.

4.2 Facial Expression Recognition

We now consider an image recognition problem, consisting in recognizing seven (balanced) classes corresponding to the universal emotions (anger, fear, disgust, joy, sadness, surprise and neutral). The patterns are gray level images of frontal faces, with standardized positions, as displayed in figure 3. The data set comprises 375 such pictures made of 140×100 pixels (Abboud *et al.*, 2003; Kanade *et al.*, 2000)

We tested kernelized logistic regression (Gaussian kernel), its minimum entropy version, nearest neigbor and the consistency method. We repeatedly (10 times) sampled 1/10 of the dataset for providing the labeled part, and the remainder for testing. Although (α, σ^2) were chosen to minimize the test error, the consistency method performed poorly with 63.8 ± 1.3 % test error (compared to 86 % error for random assignments). Nearest-neighbor get similar results with 63.1 ± 1.3 % test error, and Kernelized logistic regression (ignoring unlabeled examples) improved to reach 53.6 ± 1.3 %.

Anger Fear Disgust Joy Sadness Surprise Neutral

Figure 3: Examples from the facial expression recognition database.

Minimum entropy kernelized logistic regression regression achieves 52.0 ± 1.9 % error (compared to about 20 % errors for human on this database). The scale parameter chosen for kernelized logistic regression (by ten-fold cross-validation) amount to use a global classifier. Again, the local methods fail. This may be explained by the fact that the database contains several pictures of each person, with different facial expressions. Hence, local methods are likely to pick the same identity instead of the same expression, while global methods are able to learn the discriminating directions.

5 Discussion

We propose to tackle the semi-supervised learning problem in the supervised learning framework, by using a minimum entropy regularizer. This regularizer is motivated by theory, which shows that the information content of unlabeled examples is higher for classes with little overlap. Maximum a posteriori estimation enables to incorporate minimum entropy regularization in the learning process of any probabilistic classifier. In this framework, minimum entropy is interpreted as a "usefulness prior" for unlabeled examples, whose strength can be controlled.

Minimizing entropy gradually increases the confidence of the classifier output at unlabeled examples. Our proposal encompasses self-learning as a particular case, where, at the end of the learning process, entropy minimization converges to a solution assigning hard labels to unlabeled data. The transductive large margin classifier is another limiting case: minimizing entropy on the training sample is a means to drive the decision boundary away from these examples.

Both local and global classifiers can be fitted by the minimum entropy criterion. Global classifiers allow to improve over manifold learning when data do not lie on a low-dimensional manifold, or, as illustrated in the expression recognition experiment, when the classification task aims at differentiating examples transformed along some global direction of the manifold. Our experiments also suggest that supervised learning with minimum entropy regularization may be a serious contender to generative models. It compares favorably to mixture models in three situations: for small sample sizes, where the generative model cannot completely benefit from the knowledge of the correct joint model; when the joint distribution is (even slightly) misspecified; when the unlabeled examples turn out to be non-informative regarding class probabilities.

A Detailed derivation of $P(Y|X,Z)$

Bayes' rule and total probability theorem yields:

$$P(Y=\omega_k|\mathbf{x},\mathbf{z}) = \frac{P(\mathbf{z}|\mathbf{x},Y=\omega_k)P(Y=\omega_k|\mathbf{x})}{\sum_{\ell=1}^{K} P(\mathbf{z}|\mathbf{x},Y=\omega_\ell)P(Y=\omega_\ell|\mathbf{x})} \tag{7}$$

From the definition of $\mathbf{z}$, we have $P(\mathbf{z}|Y=\omega_k)=0$ when $z_k=0$, which implies $P(\mathbf{z}|\mathbf{x},Y=\omega_k) = z_k P(\mathbf{z}|\mathbf{x},Y=\omega_k)$.

$$\begin{aligned} P(Y=\omega_k|\mathbf{x},\mathbf{z}) &= \frac{z_k P(\mathbf{z}|\mathbf{x},Y=\omega_k)P(Y=\omega_k|\mathbf{x})}{\sum_{\ell=1}^{K} z_\ell P(\mathbf{z}|\mathbf{x},Y=\omega_\ell)P(Y=\omega_\ell|\mathbf{x})} & (8)\\ &= \frac{z_k P(Y=\omega_k|\mathbf{x})}{\sum_{\ell=1}^{K} z_\ell P(Y=\omega_\ell|\mathbf{x})} \ , & (9) \end{aligned}$$

where the last line is derived from the missing-at-random assumption, $P(\mathbf{z}|\mathbf{x},Y=\omega_k) = P(\mathbf{z}|\mathbf{x},Y=\omega_\ell)$ for all (k,ℓ) such that $z_k=z_\ell=1$.

■

B Proof of theorem 1

Theorem 1

In the two-class linear separable case, the logistic regression model with bounded weights, fitted by the minimum entropy criterion, converges towards the maximum margin separator (with maximal distance from labeled and unlabeled examples) as the bound goes to infinity.

Proof.

Consider the logistic regression model parameterized by $\boldsymbol{\theta}=(\mathbf{w},b)$: $P(Y|\mathbf{x})$ is modeled by $f(\mathbf{x};\boldsymbol{\theta}) = \frac{1}{1+e^{-(\mathbf{w}^T\mathbf{x}+b)}}$ for the positive class, and by $1-f(\mathbf{x};\boldsymbol{\theta})$ for the negative class. Let $t_i \in \{-1,+1\}$ be a binary variable defined as follows: if $\mathbf{x_i}$ is a positive labeled example, $t_i=+1$; if $\mathbf{x_i}$ is a negative labeled example, $t_i=-1$; if $\mathbf{x_i}$ is an unlabeled example, $t_i = \mathrm{sign}(f(\mathbf{x}_i)-1/2)$. The *margin* for the labeled or unlabeled example i is defined as $m_i(\boldsymbol{\theta}) = t_i(\mathbf{w}^T\mathbf{x_i}+b)$.

The cost C (6) can then be written as a function of $m_i = m_i(\boldsymbol{\theta})$ as follows

$$C(\boldsymbol{\theta}) = -\sum_{i=1}^{n_l} \log(1+e^{-m_i}) - \lambda \sum_{i=n_l+1}^{n} \log(1+e^{-m_i}) + \frac{m_i e^{-m_i}}{1+e^{-m_i}} \ , \tag{10}$$

where the indices $[1,n_l]$ and $[n_l+1,n]$ correpond to labeled and unlabeled data, respectively. The bounded weight estimate $\widehat{\boldsymbol{\theta}}_B = (\widehat{\mathbf{w}}_B, \widehat{b}_B)$ is obtained by optimizing C under the constraint $\|\mathbf{w}\| \leq B$. In the sequel, $m_i(\widehat{\boldsymbol{\theta}}_B)$ will be denoted $\widehat{m}_i$

We first show that, as B goes to infinity, all margins $\widehat{m}_i$ go to infinity. Let $\boldsymbol{\theta}^* = (\mathbf{w}^*, b^*)$ be the parameters of the maximum margin separator with $\|\mathbf{w}^*\|=1$. Let

m_i^* be $m_i(\boldsymbol{\theta}^*)$. From the definition of $\boldsymbol{\theta}^*$, $m_i^* > 0$, $i = 1, \ldots, n$. Since $m_i(B\boldsymbol{\theta}^*) = Bm_i(\boldsymbol{\theta}^*)$, $\lim_{B\to\infty} m_i(B\boldsymbol{\theta}^*) = \infty$, and $C(B\boldsymbol{\theta}^*)$ goes to zero:

$$\begin{aligned}\lim_{B\to\infty} C(B\boldsymbol{\theta}^*) &= \lim_{B\to\infty} -\sum_{i=1}^{n_l} e^{-Bm_i^*} - \lambda \sum_{i=n_l+1}^{n} Bm_i^* e^{-Bm_i^*} \\ &= 0 \ .\end{aligned}$$

Suppose now that there is at least one example i, such that $\widehat{m}_i \leq M$, where M is a positive constant. Then, C (10) can trivialy be bounded from above: $C(\widehat{\boldsymbol{\theta}}_B) \leq -\log(1+\exp(-M))$ if i is labeled and $C(\widehat{\boldsymbol{\theta}}_B) \leq -\lambda\log(1+\exp(-M))$ if i is unlabeled. Since $B\boldsymbol{\theta}^*$ is an admissible solution with $\lim_{B\to\infty} C(B\boldsymbol{\theta}^*) = 0$, $\widehat{\boldsymbol{\theta}}_B$ cannot maximize C if M is finite. We thus conclude that $\lim_{B\to\infty} \widehat{m}_i = \infty$, $i = 1, \ldots, n$.

We now show that $\|\widehat{\mathbf{w}}_B\| = B$. For this, we write the gradient of $C(\alpha\boldsymbol{\theta})$ with respect to α:

$$\begin{aligned}\left.\frac{\partial C(\alpha\boldsymbol{\theta})}{\partial\alpha}\right|_{\alpha=1} &= \sum_{i=1}^{n_l} \frac{e^{-m_i}}{(1+e^{-m_i})}\frac{\partial m_i}{\partial\alpha} + \lambda \sum_{i=n_l+1}^{n} \frac{m_i e^{-m_i}}{(1+e^{-m_i})^2}\frac{\partial m_i}{\partial\alpha} \\ &= \sum_{i=1}^{n_l} \frac{m_i e^{-m_i}}{(1+e^{-m_i})} + \lambda \sum_{i=n_l+1}^{n} \frac{m_i^2 e^{-m_i}}{(1+e^{-m_i})^2} \ ,\end{aligned}$$

As $\lim_{B\to\infty} \widehat{m}_i > 0$, each term in the sum is strictly positive for $\boldsymbol{\theta} = \widehat{\boldsymbol{\theta}}_B$, and $\lim_{B\to\infty} \left.\frac{\partial C(\alpha\widehat{\boldsymbol{\theta}}_B)}{\partial\alpha}\right|_{\alpha=1} > 0$. The constraint $\|\mathbf{w}\| \leq B$ is thus active at $\widehat{\boldsymbol{\theta}}_B = (\widehat{\mathbf{w}}_B, \widehat{b}_B)$, hence $\|\widehat{\mathbf{w}}_B\| = B$.

Finally, we derive that logistic regression asymptotically achieves maximum margin separation. Let $m_0^* = \min_{i\in[1,n]} m_i^*$ and $\widehat{m}_0 = \min_{i\in[1,n]} \widehat{m}_i$ denote the minimum margin among all labeled and unlabeled examples, and the minimum margin achieved by logistic regression, respectively. We show below that $\lim_{B\to\infty} \frac{\widehat{m}_0}{\|\widehat{\mathbf{w}}_B\|} = m_0^*$.

Let I_l^* and I_u^* denote the set of indices of respectively labeled and unlabeled examples with minimum margin $I_l^* = \{i \in [1, n_l] | m_i^* = m_0^*\}$ and $I_u^* = \{i \in [n_l+1, n] | m_i^* = m_0^*\}$. Accordingly, we denote $\widehat{I}_l = \{i \in [1, n_l] | \widehat{m}_i = \widehat{m}_0\}$ and $\widehat{I}_u = \{i \in [n_l + 1, n] | \widehat{m}_i = \widehat{m}_0\}$. Finally, we define $c^* = \min(2, \arg\min_{m_i^* > m_0^*} \frac{m_i^* - m_0^*}{m_0^*})$, and $\widehat{c} = \min(2, \arg\min_{\widehat{m}_i > \widehat{m}_0} \frac{\widehat{m}i - \widehat{m}_0}{\widehat{m}_0})$.

$$\begin{aligned}C(B\boldsymbol{\theta}^*) &= -\left(|I_l^*| - \lambda \sum_{i\in I_u^*} (1 + Bm_i^*)\right) e^{-Bm_0^*} + O(Bm_0^* \, e^{-c^* B m_0^*}) \\ C(\widehat{\boldsymbol{\theta}}_B) &= -\left(|\widehat{I}_l| - \lambda \sum_{i\in \widehat{I}_u} (1 + \widehat{m}_i)\right) e^{-\widehat{m}_0} + O(\widehat{m}_0 \, e^{-\widehat{c}\widehat{m}_0})\end{aligned}$$

where $|I|$ denote the cardinal number of set I.

We now note that for any $\varepsilon < 1$, $\lim_{B\to\infty} Bm_0^* e^{-(c^*-\varepsilon)Bm_0^*} = 0$, hence

$$\begin{aligned}\lim_{B\to\infty} e^{\varepsilon Bm_0^*} C(B\boldsymbol{\theta}^*) &= \lim_{B\to\infty} -\left(|I_l^*| - \lambda \sum_{i\in I_u^*}(1+Bm_i^*)\right) e^{-(1-\varepsilon)Bm_0^*} \\ &= 0\end{aligned}$$

The optimality of $\widehat{\boldsymbol{\theta}}_B$ entails $C(\widehat{\boldsymbol{\theta}}_B) \geq C(B\boldsymbol{\theta}^*)$, which implies that for any $\varepsilon < 1$,

$$\begin{aligned}\lim_{B\to\infty} e^{\varepsilon Bm_0^*} C(\widehat{\boldsymbol{\theta}}_B) &= 0 \\ \lim_{B\to\infty} -\left(|\widehat{I}_l| - \lambda \sum_{i\in\widehat{I}_u}(1+\widehat{m}_i)\right) e^{\varepsilon Bm_0^* - \widehat{m}_0} &= 0\end{aligned}$$

Hence, for any $\varepsilon < 1$, $\lim_{B\to\infty} B(\frac{\widehat{m}_0}{B} - \varepsilon m_0^*) = \infty$. As by definition $\frac{\widehat{m}_0}{B} \leq m_0^*$, we conclude that $\lim_{B\to\infty} \frac{\widehat{m}_0}{B} = \lim_{B\to\infty} \frac{\widehat{m}_0}{\|\widehat{\mathbf{w}}_B\|} = m_0^*$.

■

Note that besides the linear separable case, this theorem can be easily extended to kernelized logistic regression using kernels ensuring linear separability (such as the Gaussian kernel).

References

ABBOUD B., DAVOINE F. & MO D. (2003). Expressive face recognition and synthesis. In *Computer Vision and Pattern Recognition Workshop*, volume 5, p.5̃4.

AMINI M. R. & GALLINARI P. (2002). Semi-supervised logistic regression. In *15th European Conference on Artificial Intelligence*, p. 390–394: IOS Press.

BENNETT K. P. & DEMIRIZ A. (1999). Semi-supervised support vector machines. In M. S. KEARNS, S. A. SOLLA & D. A. COHN, Eds., *Advances in Neural Information Processing Systems 11*, p. 368–374: MIT Press.

BERGER J. O. (1985). *Statistical Decision Theory and Bayesian Analysis*. New York: Springer, 2 edition.

BRAND M. (1999). Structure learning in conditional probability models via an entropic prior and parameter extinction. *Neural Computation*, **11**(5), 1155–1182.

CASTELLI V. & COVER T. M. (1996). The relative value of labeled and unlabeled samples in pattern recognition with an unknown mixing parameter. *IEEE Trans. on Information Theory*, **42**(6), 2102–2117.

CORDUNEANU A. & JAAKKOLA T. (2003). On information regularization. In *Proceedings of the 19th conference on Uncertainty in Artificial Intelligence (UAI)*.

FRIEDMAN J., HASTIE T. & TIBSHIRANI R. (2000). Additive logistic regression: a statistical view of boosting. *The Annals of Statistics*, **28**(2), 337–407.

GRANDVALET Y. (2002). Logistic regression for partial labels. In *9th Information Processing and Management of Uncertainty in Knowledge-based Systems – IPMU'02*, p. 1935–1941.

JIN R. & GHAHRAMANI Z. (2003). Learning with multiple labels. In *Advances in Neural Information Processing Systems 15*: MIT Press.

JOACHIMS T. (1999). Transductive inference for text classification using support vector machines. In *International Conference on Machine Learning (ICML)*, p. 200–209.

KANADE T., COHN J. & TIAN Y. (2000). Comprehensive database for facial expression analysis. In *4th IEEE International Conference on Automatic Face and Gesture Recognition*.

MCLACHLAN G. J. (1992). *Discriminant analysis and statistical pattern recognition*. Wiley.

NIGAM K. & GHANI R. (2000). Analyzing the effectiveness and applicability of co-training. In *Ninth International Conference on Information and Knowledge Management*, p. 86–93.

NIGAM K., MCCALLUM A. K., THRUN S. & MITCHELL T. (2000). Text classification from labeled and unlabeled documents using EM. *Machine learning*, **39**(2/3), 103–134.

O'NEILL T. J. (1978). Normal discrimination with unclassified observations. *Journal of the American Statistical Association*, **73**(364), 821–826.

ROSE K., GUREWITZ E. & FOX G. (1990). A deterministic annealing approach to clustering. *Pattern Recognition Letters*, **11**(9), 589–594.

SEEGER M. (2002). *Learning with labeled and unlabeled data*. Rapport interne, Institute for Adaptive and Neural Computation, University of Edinburgh.

SZUMMER M. & JAAKKOLA T. S. (2003). Information regularization with partially labeled data. In *Advances in Neural Information Processing Systems 15*: MIT Press.

TONG S. & KOLLER D. (2000). Restricted bayes optimal classifiers. In *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI)*, p. 658–664.

VAPNIK V. N. (1998). *Statistical Learning Theory*. Adaptive and learning systems for signal processing, communications, and control. New York: Wiley.

ZHOU D., BOUSQUET O., NAVIN LAL T., WESTON J. & SCHÖLKOPF B. (2004). Learning with local and global consistency. In *Advances in Neural Information Processing Systems 16*.

ZHU X., GHAHRAMANI Z. & LAFFERTY J. (2003). Semi-supervised learning using Gaussian fields and harmonic functions. In *20th Int. Conf. on Machine Learning*, p. 912–919.

Apprentissage semi-supervisé asymétrique et estimation d'affinités locales dans les protéines

Christophe Nicolas Magnan

Laboratoire d'Informatique Fondamentale de Marseille (LIF), UMR CNRS 6166[†]
magnan@cmi.univ-mrs.fr

Résumé : Cet article présente une étude en apprentissage automatique semi-supervisé asymétrique, c'est-à-dire à partir de données positives et non-étiquetées, ainsi qu'une application à un problème bio-informatique. Nous montrons que sous des hypothèses très naturelles, le classifieur naïf de Bayes peut être identifié à partir de données positives et non étiquetées. Nous en déduisons des algorithmes que nous étudions sur des données artificielles. Enfin, nous présentons une application de ces travaux sur le problème de l'extraction d'affinités locales dans les protéines pour la prédiction des ponts disulfures. Les résultats permettent d'étayer une hypothèse sur la manière de formaliser les données biologiques pour des cas d'interactions physiques locales.

Mots-clés : Apprentissage semi-supervisé, Naïve Bayes, E.M., Ponts disulfures.

1 Introduction

De nombreux thèmes contemporains de recherche en biologie et en bio-informatique, liés à l'étude des protéines, sont étroitement en rapport avec des phénomènes d'interactions physiques locales. Les brins beta ou encore les ponts disulfures dans les protéines sont des exemples de phénomènes liés à des interactions physiques. Pouvoir prédire avec un maximum de précision et de pertinence ces interactions améliorerait de manière significative la prédiction de la structure tridimensionnelle des protéines, elle-même étant liée à sa fonction biologique. Déterminer cette structure expérimentalement, par résonance magnétique nucléaire, est une tâche longue, difficile, coûteuse et de plus, non applicable à certaines familles de protéines. Or on sait par ailleurs que la structure d'une protéine est déterminée par sa structure primaire, séquence d'acides aminés, et par son milieu d'occupation. Il est donc naturel de mettre au point des algorithmes de prédiction de la structure 3D à partir de la séquence primaire. Actuellement, près de 2 millions de séquences protéiques sont disponibles pour moins de 20000 structures 3D.

Nous nous sommes intéressés à un élément de la structure 3D : les ponts disulfures. Ces ponts sont des liaisons covalentes qui se forment entre deux cystéines suite à leur oxydation. La cystéine est un des 20 acides aminés qui constituent la séquence primaire

[†]Ces travaux sont en partie financés par l'A.C.I. Masses de Données GENOTO3D

des protéines. Une cystéine peut former un pont avec une autre cystéine proche ou distante sur la séquence, et est contrainte à une unicité de liaison.

La prédiction des ponts disulfures peut se décomposer en deux étapes : la prédiction des cystéines oxydées et la prédiction des ponts eux-mêmes. De nombreux travaux ont permis d'élaborer des méthodes adaptées à la première tâche, mais il y a peu de résultats pour la deuxième. C'est donc à la prédiction des ponts disulfures, sachant l'état d'oxydation des cystéines, que nous nous sommes intéressés.

Des acides aminés proches dans l'espace interagissent entre eux. On peut donc imaginer que les interactions entre les acides aminés situés autour de deux cystéines contribuent à ce que nous appellerons une *affinité* entre ces cystéines. Il est clair que cette information seule ne suffit pas à déterminer les ponts, mais nous cherchons à l'extraire au mieux dans le but de l'intégrer dans des processus de prédiction des ponts disulfures.

Déterminer si deux segments d'une protéine ont de l'affinité l'un pour l'autre peut être considéré comme un problème de classification supervisée, où les segments appariés sont vus comme ayant de l'affinité, et les segments non appariés comme n'en ayant pas. On peut également voir le problème de manière différente, en considérant comme précédemment que les couples de segments appariés ont de l'affinité l'un pour l'autre, mais que des couples de segments non appariés peuvent ou non en avoir. En effet, une cystéine ne pouvant appartenir qu'à un seul pont disulfure, le nombre de ponts dans une protéine est donc contraint. Nous modélisons cette situation en supposant que les paires de segments appariés appartiennent à la classe des segments qui ont de l'affinité l'un pour l'autre et que les paires de segments non appariés n'apportent pas d'informations sur la notion d'affinité.

C'est alors un cas d'apprentissage semi-supervisé, généralement rencontré sous le nom *d'apprentissage à partir de données positives et non étiquetées*, que nous appelons *asymétrique*. Ce contexte particulier d'apprentissage nécessite de montrer que le problème est bien posé (les données permettent-elles à la limite d'identifier la cible) et l'élaboration de nouveaux algorithmes.

Nous donnons dans la section 2 une brève description des méthodes utilisées lors de nos travaux ainsi que des résultats de la littérature sur ce thème de travail. En section 3, nous exposons notre étude théorique de l'apprentissage asymétrique et en section 4 et 5 nous présentons des résultats expérimentaux sur des données artificielles et réelles.

2 Préliminaires

Cette section présente les méthodes et algorithmes utilisés dans nos travaux, ainsi que divers résultats sur l'apprentissage à partir de données positives et non étiquetées.

2.1 La règle de Bayes et le classifieur naïf de Bayes

Soit $X = \prod_{i=1}^{m} X^i$ un domaine défini par m attributs symboliques. Pour tout $x \in X$, on notera x^i la projection de x sur X^i. Soit P une loi de distribution sur X et soit Y un ensemble de classes muni des lois de distributions conditionnelles $P(.|x)$ pour tout $x \in X$. La règle de décision optimale pour attribuer une classe à tout objet $x \in X$ est

la *règle de Bayes* qui sélectionne la classe $y \in Y$ possédant la plus grande probabilité sachant x. On peut formuler cette règle de la façon suivante :

$$\boxed{C_{Bayes}(x) \;=\; \underset{y}{argmax}\;\; P(y|x) \;=\; \underset{y}{argmax}\;\; P(x|y) \cdot P(y) \quad (x \in X, y \in Y)}$$

En règle générale, les quantités $P(x|y)$ ne peuvent pas être estimées à partir d'un échantillon d'apprentissage. En revanche, si les attributs sont indépendants deux à deux conditionnellement à chaque classe, on a alors $P(x|y) = \prod_{i=1}^{m} P(x^i|y)$ et dans ce cas, le nombre de paramètres à estimer devient raisonnable. Pour des classes et attributs binaires, le nombre de paramètres à estimer passe de $O(2^m)$ à $O(m)$. Que l'hypothèse d'indépendance soit vérifiée ou non, on appelle *classifieur naïf de Bayes* la règle définie par :

$$\boxed{C_{NB}(x) \;=\; \underset{y}{argmax}\;\; P(y) \prod_{i=1}^{m} P(x^i|y) \;\; (x \in X, y \in Y)}$$

L'hypothèse d'indépendance n'est pas vérifiée dans la plupart des problèmes réels. Néanmoins, le classifieur naïf de Bayes est connu pour donner de bons résultats pour des tâches de classification (Domingos & Pazzani, 1996).

Les paramètres nécessaires à l'évaluation de C_{NB} lorsque $Y = \{0, 1\}$ sont les probabilités $\alpha = P(y = 1)$, la probabilité d'observer un exemple de la classe notée 1 que nous appelons classe *positive* et l'ensemble des $\lambda_{ikj} = P(x^i = k|y = j)$, les probabilités d'observer l'attribut i de x égal à k sachant que x est de la classe j ($j \in \{0, 1\}$). Une instance de ces paramètres sera appelée *modèle* et sera notée θ.

2.2 Le principe du maximum de vraisemblance et application au classifieur naïf de Bayes

L'objectif de l'apprentissage automatique est de construire un modèle qui rend compte des données. Le principe du maximum de vraisemblance définit un critère permettant de choisir un tel modèle.

2.2.1 Principe du maximum de vraisemblance

Pour un échantillon $S = \{(x_s, y_s), s \in 1, ..., l\}$ de données indépendamment et identiquement distribuées selon la loi jointe $P(x, y) = P(x)P(y|x)$ et un modèle θ, on appelle *vraisemblance* (resp. *log-vraisemblance*) de S pour le modèle θ et on note $L(\theta, S)$ (resp. $l(\theta, S)$) les valeurs :

$$\boxed{L(\theta, S) = \prod_{s=1}^{l} P(x_s, y_s|\theta) \;\text{ et }\; l(\theta, S) = log\; L(\theta, S)}$$

Le principe du maximum de vraisemblance recommande de trouver un modèle θ tel que $L(\theta, S)$ - et donc aussi $l(\theta, S)$ - soit maximale.

2.2.2 Application au classifieur naïf de Bayes

Si on note n_0 le nombre de données classées '0' dans l'échantillon S d'apprentissage, n_1 le nombre de données classées '1' ($n_0 + n_1 = l$), n_{ij}^k le nombre d'exemples tels que $x^i = k$ et $y = j$ et $Dom(x^i)$ l'ensemble des valeurs que peut prendre l'attribut x^i, on peut écrire la vraisemblance et la log-vraisemblance de S dans le modèle θ en fonction de α et des λ_{ikj} :

$$L(\theta, S) = \prod_{s=1}^{l} P(y_s) \left[\prod_{i=1}^{m} P(x_s^i | y_s) \right] = \alpha^{n_1} \cdot (1-\alpha)^{n_0} \cdot \prod_{\substack{1 \le i \le m, 0 \le j \le 1 \\ k \in Dom(x^i)}} \lambda_{ikj}^{n_{ij}^k}$$

$$\begin{aligned} l(\theta, S) &= log\ L(\theta, S) \\ &= n_1 \cdot log\alpha + n_0 \cdot log(1-\alpha) + \sum_{\substack{1 \le i \le m, 0 \le j \le 1 \\ k \in Dom(x^i)}} n_{ij}^k \cdot log\ \lambda_{ikj} \end{aligned}$$

Cette fonction trouve son maximum pour le modèle θ_{mv}^S suivant :

- $\alpha = \frac{n_1}{n_0 + n_1}$, la proportion d'exemples positifs dans les données d'apprentissage,
- $\lambda_{ikj} = \frac{n_{ij}^k}{\sum_{r \in Dom(x^i)} n_{ij}^r}$, le rapport du nombre de données d'apprentissage de classe j tel que $x^i = k$ par le nombre de données étiquetées j.

2.2.3 Cas semi-supervisé

En contexte semi-supervisé, on dispose de deux échantillons de données : $S_{lab} = \{(x_1, y_1), ..., (x_l, y_l)\}$, un échantillon de données étiquetées, et $S_{unl} = \{x'_1, ..., x'_{l'}\}$, un échantillon de données non étiquetées. On modélise la présence de ces deux échantillons par l'existence d'un oracle qui, avec une certaine probabilité β fournit un exemple étiqueté et avec une probabilité $1 - \beta$ procure un exemple non étiqueté. Le paramètre β complète le modèle θ vu précédemment. Dans ce nouveau modèle, que nous notons θ', les probabilités d'avoir $z = (x, y) \in S_{lab}$ et d'avoir $z = x \in S_{unl}$ se calculent de la manière suivante :

$$P(z = (x, y) | \theta', z \in S_{lab}) = \beta . P(x, y | \theta)$$

$$P(z = x | \theta', x \in S_{unl}) = (1 - \beta) . P(x | \theta)$$

avec $P(x|\theta) = P(y = 1|\theta) . P(x, y | y = 1, \theta) + P(y = 0|\theta) . P(x, y | y = 0, \theta)$

La vraisemblance de $S_{lab}\ et\ S_{unl}$ pour le modèle θ' s'écrit alors :

$$\begin{aligned} L(\theta', S_{lab}, S_{unl}) &= \prod_{s=1}^{l} \beta P(x_s, y_s | \theta) \prod_{r=1}^{l'} (1-\beta) P(x'_r | \theta) \\ &= \beta^l L(\theta, S_{lab}) (1-\beta)^{l'} \prod_{r=1}^{l'} \left(\alpha \prod_{\substack{1 \le i \le m \\ k / x_r^i = k}} \lambda_{ik1} + (1-\alpha) \prod_{\substack{1 \le i \le m \\ k / x_r^i = k}} \lambda_{ik0} \right) \end{aligned}$$

Cette formule permet de trouver analytiquement la valeur optimale de β pour maximiser la vraisemblance : $\beta = \frac{l}{l+l'}$, soit la proportion d'exemples étiquetés dans l'ensemble d'apprentissage. En revanche, elle ne permet pas de trouver les paramètres α et λ_{ikj} du modèle θ' qui maximisent la vraisemblance, comme dans le cas supervisé du modèle naïf de Bayes. La méthode E.M. permet de pallier ce problème (*cf.* section 2.3).

2.2.4 Cas semi-supervisé asymétrique

Le cas semi-supervisé asymétrique est proche du cas semi-supervisé classique. L'ensemble S_{lab} est réduit à $S_{pos} = \{(x_1, 1), ..., (x_l, 1)\}$, cela entraîne que le paramètre β représente maintenant la probabilité d'observer un exemple positif. Avec les mêmes notations que précédemment, la vraisemblance de S_{pos} et S_{unl} pour le modèle θ' s'écrit :

$$\boxed{\begin{array}{l} L(\theta', S_{pos}, S_{unl}) \\ = \beta^l L(\theta, S_{pos})(1-\beta)^{l'} L(\theta, S_{unl}) \\ = \beta^l \alpha^l \prod_{r=1}^{l} \left(\prod_{\substack{1 \leq i \leq m \\ k/x_r^i = k}} \lambda_{ik1} \right) (1-\beta)^{l'} \prod_{r=1}^{l'} \left(\alpha \prod_{\substack{1 \leq i \leq m \\ k/x_r^i = k}} \lambda_{ik1} + (1-\alpha) \prod_{\substack{1 \leq i \leq m \\ k/x_r^i = k}} \lambda_{ik0} \right) \end{array}}$$

2.3 La méthode E.M. (Expectation, Maximisation)

La méthode E.M. a été élaborée par (Dempster *et al.*, 1977) pour l'inférence de modèles de mélange de densités.

2.3.1 Méthode

Cette section décrit la méthode E.M. en suivant (Hastie *et al.*, 2001). Soient θ' un modèle, Z l'ensemble des données observées, Z_m les données manquantes, et T l'ensemble de données complètes d'un problème, $T = (Z, Z_m)$. Si on note :

- $l_0(\theta', T)$ la log-vraisemblance de T dans le modèle θ',
- $l_1(\theta', Z_m|Z)$ la log-vraisemblance de Z_m dans le modèle θ' sachant Z,
- $l(\theta', Z)$ la log-vraisemblance de Z dans le modèle θ',

alors $l(\theta', Z) + l_1(\theta', Z_m|Z) = l_0(\theta', T)$, soit :

$$\boxed{l(\theta', Z) = l_0(\theta', T) - l_1(\theta', Z_m|Z)}$$

En supposant que les données sont générées selon θ et que Z a été observé, les termes de l'égalité précédente sont des variables aléatoires dépendantes de Z_m, on peut donc calculer leur espérance :

$$\boxed{E(\ l(\theta', Z)|Z, \theta\ ) = E(\ l_0(\theta', T)|Z, \theta\ ) - E(\ l_1(\theta', Z_m)|Z, \theta\ )}$$

Soit, en posant $Q(\theta', \theta) = E(\ l_0(\theta', T)|Z, \theta\ )$ et $R(\theta', \theta) = E(\ l_1(\theta', Z_m)|Z, \theta\ )$ et en remarquant que $E(\ l(\theta', Z)|Z, \theta\ ) = l(\theta', Z)$:

$$\boxed{l(\theta', Z) = Q(\theta', \theta) - R(\theta', \theta)}$$

La méthode du maximum de vraisemblance demande de chercher un modèle θ' qui maximise $l(\theta', Z)$. La méthode E.M. est une heuristique, basée sur le résultat suivant qui énonce que maximiser Q ne peut pas faire décroître la vraisemblance.

Théorème 1
Si $Q(\theta', \theta) > Q(\theta, \theta)$ *alors* $l(\theta', Z) > l(\theta, Z)$ *(Dempster* et al.*, 1977)*

La méthode E.M. peut être décrite par l'algorithme suivant :

Algorithme 1
Entrée : Z
1) Choisir un modèle $\hat{\theta}^0$
2) Calculer $Q(\hat{\theta}^i, \hat{\theta}^i)$ *pour le* i *courant (phase d'estimation)*
3) Trouver $\hat{\theta}^{i+1}$ *tel que* $Q(\hat{\theta}^{i+1}, \hat{\theta}^i) > Q(\hat{\theta}^i, \hat{\theta}^i)$ *(phase de maximisation)*
4) Itérer à l'étape deux jusqu'à convergence
Sortie : *un modèle* θ^c

L'algorithme converge vers un maximum local de la vraisemblance. (Dempster *et al.*, 1977) proposent de répéter l'expérience et de choisir le modèle θ^c de plus grande vraisemblance.

2.3.2 Application au classifieur naïf de Bayes

Pour le classifieur naïf de Bayes en contexte semi-supervisé, $Z = \{S_{lab}, S_{unl}\}$ avec $S_{lab} = \{(x_1, y_1), ..., (x_l, y_l)\}$ et $S_{unl} = \{x'_1, .., x'_{l'}\}$, les données manquantes Z_m sont les étiquettes des données de S_{unl} et $T = (Z, Z_m)$. Avec les mêmes notations que précédemment, les paramètres $\alpha = P(y = 1)$ et $\lambda_{ikj} = P(x^j = k|y = i)$ se calculent de la manière suivante à chaque itération de l'algorithme (McCallum *et al.*, 1999) :

$$\alpha = \frac{n_1 + \sum_{s=1}^{l'} \hat{P}(y'_s = 1|x'_s, \hat{\theta})}{l + l'}, \quad \lambda_{ikj} = \frac{n_{ij}^k + \sum_{s=1}^{l'} \hat{P}(y'_s = j|x_s'^i = k, \hat{\theta})}{\sum_{r \in Dom(x^i)} [n_{ij}^r + \sum_{s=1}^{l'} \hat{P}(y'_s = j|x_s'^i = r, \hat{\theta})]}$$

où les $\hat{P}$ sont estimées en fonction du modèle courant $\hat{\theta}$. (McCallum *et al.*, 1999) proposent l'algorithme suivant :

Algorithme 2 (EM+NB semi-supervisé)
Entrée : S_{lab}, S_{unl}
1) $\hat{\theta}^0 =$ *N.B.(*S_{lab}*), le modèle appris sur les données étiquetées*
2) $\forall x' \in S_{unl}$ *calculer* $P(y' = j|x', \hat{\theta}^k)$*, avec le modèle* $\hat{\theta}^k$ *courant,* $j \in \{0, 1\}$
3) Maximiser $Q(\hat{\theta}^{k+1}, \hat{\theta}^k)$
4) Itérer à l'étape 2 jusqu'à convergence
Sortie : *un modèle* θ^c

Les résultats sur un problème de classification de textes montrent une amélioration sensible des résultats lors de l'ajout de données non étiquetées aux données étiquetées.

2.4 Apprentissage à partir de données positives et non étiquetées

Le thème de l'apprentissage à partir de données positives et non étiquetées a déjà été abordé par divers chercheurs (Denis, 1998; Denis *et al.*, 1999; Liu & Li, 2003). Dans ce contexte, on trouve également l'utilisation du classifieur naïf de Bayes pour une application à la classification de textes (Denis *et al.*, 2003). Dans ces travaux, les auteurs partent de l'hypothèse que le paramètre $\alpha = P(y = 1)$ est connu, ce paramètre étant indispensable pour calculer le classifieur (*cf.* section 3.1). Or, ce paramètre étant généralement inconnu, l'estimation de celui-ci est donc un problème latent.

Il a été montré dans (Whiley & Titterington, 2002; Geiger *et al.*, 2001) que, sous l'hypothèse d'indépendance des attributs conditionnellement à chaque classe, les paramètres du modèle sont identifiables à partir de la distribution $P(.)$ sur X lorsque le nombre d'attributs est supérieur à deux et à la détermination des classes près, c'est-à-dire $P(.)$ détermine l'ensemble $\{\alpha, 1 - \alpha\}$. Des formules analytiques permettant d'estimer les paramètres du modèle à partir d'échantillons d'exemples non étiquetés sont également fournies. Mais dans le cas qui nous intéresse, nous disposons aussi d'exemples positifs qui doivent permettre d'identifier les classes et d'obtenir de meilleures estimations.

3 Estimation des paramètres d'un classifieur naïf de Bayes

Après avoir exposé quelques propriétés sur le cas général de l'apprentissage asymétrique (section 3.1), nous établissons une formule qui montre que les paramètres du modèle sont identifiables lorsque le nombre d'attributs est supérieur ou égal à deux (section 3.2). Cette formule permet de définir un estimateur consistant du paramètre $P(y = 1)$. En section 3.3, nous proposons une adaptation de l'algorithme 2 au cas semi-supervisé asymétrique en vue de comparer les deux algorithmes sur des données artificielles.

3.1 Cas général

Le cadre général de l'apprentissage statistique suppose l'existence de distributions $P(.)$ sur X et $P(.|x)$ sur Y pour tout $x \in X$. Dans le cas $Y = \{0, 1\}$, ces distributions sont déterminées par la donnée de $P(.)$ et $P(.|y = 1)$ sur X, et $P(y = 1)$. En effet :

$$\boxed{P(y = 1|x) = \frac{P(x|y = 1) \cdot P(y = 1)}{P(x)} \text{ et } P(y = 0|x) = 1 - P(y = 1|x)}$$

Des échantillons de données positives et non étiquetées, S_{pos} et S_{unl}, permettent de déterminer des estimations des distributions $P(.)$ sur X et $P(.|y = 1)$ sur X. Dans le cas général le paramètre $P(y = 1)$ doit être connu.

Propriété 1
En règle générale, P(y=1) n'est pas déterminé par la donnée de $P(x)$ et $P(x|y = 1)$.

Soit $r = Inf\{\frac{P(x)}{P(x|y=1)} \mid x \in X \text{ et } P(x|y=1) \neq 0\}$. Alors pour tout $\lambda \in]0, r]$, il existe P' tel que pour tout $x \in X$, $P'(x) = P(x)$, $P'(x|y=1) = P(x|y=1)$ et $P'(y=1) = \lambda$. En effet, soit λ tel que $0 < \lambda \leq r$. Alors, en posant :

$$\boxed{P'(x|y=0) = \frac{P'(x) - P'(x|y=1).\lambda}{1-\lambda} \ \forall x \in X}$$

on obtient $P'(y=1) = \lambda$. L'ensemble des λ acceptables étant l'intervalle $]0, r]$, le paramètre $P(y=1)$ n'est donc pas déterminé.

Remarque
Dans certains cas, $P(y=1)$ est déterminé par les paramètres $P(x)$ et $P(x|y=1)$. Un cas trivial est celui des modèles déterministes : pour tout x, $P(y=1|x) = 1$ ou $P(y=1|x) = 0$. Dans ce cas $P(y=1) = \sum_{x \in X} P(x)P(y=1|x) = \sum_{P(x|y=1)\neq 0} P(x)$.

3.2 Déterminisme et identification du paramètre P(y=1)

Nous montrons dans cette section que les distributions $P(.)$ sur X et $P(.|y=1)$ sur X déterminent le paramètre $P(y=1)$ pour les distributions suivant l'hypothèse naïve de Bayes et nous donnons un estimateur consistant pour ce paramètre. L'apprentissage à partir de données positives et non étiquetées est donc envisageable sans hypothèse supplémentaire.

Théorème 2
Pour les distributions de probabilité satisfaisant l'hypothèse naïve de Bayes, la donnée de $P(.)$ sur X et de $P(.|y=1)$ sur X détermine le paramètre $P(y=1)$ sous réserve qu'il existe au moins deux attributs distincts x^i et x^j tels que $P(x^i = .|y=1) \neq P(x^i = .|y=0)$ et $P(x^j = .|y=1) \neq P(x^j = .|y=0)$.

Démonstration
Avant le cas général, nous traitons les deux cas limites du paramètre $P(y=1)$:

- Remarquons tout d'abord que le cas $P(y=1) = 0$ est impossible puisque l'on suppose l'existence de l'ensemble S_{pos}.
- De plus, sous les conditions du théorème, nous montrons que $P(y=1) = 1 <=> P(.) = P(.|y=1)$ pour tout $x \in X$, en effet,
  - l'implication $=>$ est triviale,
  - supposons que $P(.) = P(.|y=1)$ et $P(y=1) < 1$, alors :
    $P(.|y=1) \cdot (1 - P(y=1)) = P(.|y=0) \cdot (1 - P(y=1))$ et donc :
    $P(.|y=1) = P(.|y=0)$, ce qui contredit l'hypothèse.

Nous considérons maintenant le cas général $0 < P(y=1) < 1$.

Soient $p_{ik} = P(x^i = k|y=1)$ et $q_{ik} = P(x^i = k|y=0)$ $\forall i \in \{1, ..., m\}$. Pour tout couple (i, j) d'attributs distincts, et pour tout couple k, l de valeurs respectives des attributs x^i et x^j, on peut déduire le système d'équations suivant :

$$\begin{cases} \alpha_{ik} = P(x^i = k) = p_{ik}.P(y=1) \ + \ q_{ik}.(1 - P(y=1)) \\ \alpha_{jl} = P(x^j = l) = p_{jl}.P(y=1) \ + \ q_{jl}.(1 - P(y=1)) \\ \alpha_{ik,jl} = P(x^i = k \cap x^j = l) = p_{ik}.p_{jl}.P(y=1) \ + \ q_{ik}.q_{jl}.(1 - P(y=1)) \end{cases}$$

Considérons un couple (i, j) d'attributs distincts et un couple (k, l) de valeurs d'attributs respectives pour x^i et x^j tels que $p_{ik} \neq q_{ik}$ et $p_{jl} \neq q_{jl}$, avec les égalités provenant des deux premières équations du système, on peut écrire :

$$\boxed{q_{ik} = \frac{\alpha_{ik} - p_{ik}.P(y=1)}{1 - P(y=1)} \quad q_{jl} = \frac{\alpha_{jl} - p_{jl}.P(y=1)}{1 - P(y=1)}}$$

En remplaçant q_{ik} et q_{jl} dans la troisième équation du système, on obtient, après simplification, l'équation du premier degré en $P(y = 1)$ suivante :

$$\boxed{P(y=1)(p_{ik}p_{jl} - \alpha_{ik}p_{jl} - \alpha_{jl}p_{ik} + \alpha_{ik,jl}) = \alpha_{ik,jl} - \alpha_{ik}\alpha_{jl}}$$

Pour obtenir une expression analytique de $P(y = 1)$ à partir de cette équation, il est nécessaire de montrer que $p_{ik}p_{jl} - \alpha_{ik}p_{jl} - \alpha_{jl}p_{ik} + \alpha_{ik,jl} \neq 0$ et que $\alpha_{ik,jl} \neq \alpha_{ik} \cdot \alpha_{jl}$ (sans quoi $P(y = 1) = 0$).

- On peut écrire ($p_{ik}p_{jl} - \alpha_{ik}p_{jl} - \alpha_{jl}p_{ik} + \alpha_{ik,jl}$) en fonction de p_{ik}, p_{jl}, q_{ik}, q_{jl}, et $P(y = 1)$ en remplaçant α_{ik}, α_{jl}, $\alpha_{ik,jl}$ par leur définition. On obtient :

$$\boxed{p_{ik}p_{jl} - \alpha_{ik}p_{jl} - \alpha_{jl}p_{ik} + \alpha_{ik,jl} = (1 - P(y=1)).(p_{ik} - q_{ik}).(p_{jl} - q_{jl})}$$

Donc sous les conditions du théorème : $p_{ik}p_{jl} - \alpha_{ik}p_{jl} - \alpha_{jl}p_{ik} + \alpha_{ik,jl} \ \neq 0$.

- $\alpha_{ik,jl} = \alpha_{ik} \cdot \alpha_{jl}$ signifie que les attributs descriptifs sont indépendants. Or, nous travaillons sous l'hypothèse d'indépendance des attributs conditionnellement à chaque classe. Les deux conditions réunies ont pour conséquence directe (développement non précisé faute de place) : $P(y = 1) \cdot (1 - P(y = 1)) \cdot (p_{ik} - q_{ik}) \cdot (p_{jl} - q_{jl}) = 0$. Sous les conditions du théorème, cette égalité n'est jamais vérifiée.
On peut donc écrire :

$$\boxed{(1) \quad P(y=1) = \frac{\alpha_{ik,jl} - \alpha_{ik}\alpha_{jl}}{p_{ik}p_{jl} - \alpha_{ik}p_{jl} - \alpha_{jl}p_{ik} + \alpha_{ik,jl}}}$$

Les paramètres $\alpha_{ik,jl}, \alpha_{ik}, \alpha_{jl}, p_{ik}, p_{jl}$ pouvant être calculés à partir des distributions $P(.)$ et $P(.|y = 1)$, $P(y = 1)$ est bien déterminé par $P(.)$ et $P(.|y = 1)$.■

Cette formule conduit à une estimation naturelle du paramètre $P(y = 1)$. Soient $\hat{\alpha}_{ik,jl}, \hat{\alpha}_{ik}, \hat{\alpha}_{jl}, \hat{p}_{ik}, \hat{p}_{jl}$ des estimateurs des paramètres $\alpha_{ik,jl}, \alpha_{ik}, \alpha_{jl}, p_{ik}, p_{jl}$ respectivement, on considère :

$$(2) \quad \hat{P}(y=1) = \frac{\sum_{i,j,k,l} |\hat{\alpha}_{ik,jl} - \hat{\alpha}_{ik}\hat{\alpha}_{jl}|}{\sum_{i,j,k,l} |\hat{p}_{ik}\hat{p}_{jl} - \hat{\alpha}_{ik}\hat{p}_{jl} - \hat{\alpha}_{jl}\hat{p}_{ik} + \hat{\alpha}_{ik,jl}|}$$

avec $i \neq j$ et k et l un couple de valeurs respectives des attributs x^i et x^j. On en déduit l'algorithme d'apprentissage suivant :

Algorithme 3 (NB semi-sup. asymétrique)
Entrée : *S_{pos}, S_{unl}*
1) Calculer les estimateurs $\hat{\alpha}_{ik,jl}, \hat{\alpha}_{ik}, \hat{\alpha}_{jl}, \hat{p}_{ik}, \hat{p}_{jl}$ des paramètres $\alpha_{ik,jl}, \alpha_{ik}, \alpha_{jl}, p_{ik}, p_{jl}$ sur S_{pos} et S_{unl}
2) Calculer $\hat{P}(y=1)$ par la formule (2)
3) Calculer les estimateurs des paramètres manquants du modèle : $\hat{q}_{ik}$, $\hat{q}_{jl}$
Sortie : *un modèle $\hat{\theta}$*

En l'absence de résultats théoriques sur la vitesse de convergence de l'estimateur (2), nous avons décidé de comparer les résultats obtenus par l'algorithme 3 à ceux que l'on obtient en maximisant la vraisemblance au moyen de la méthode E.M.. Pour cela, nous proposons un algorithme adapté de l'algorithme 2 à la section suivante.

3.3 Algorithme naïf de Bayes en contexte semi-supervisé asymétrique

Nous présentons ici un algorithme itératif, adapté de l'algorithme 2 (McCallum *et al.*, 1999) qui permet d'estimer le paramètre $P(y=1)$ sur le critère du maximum de vraisemblance. Nous proposons la solution suivante :

Algorithme 4 (EM+NB asym. + aléa)
Entrée : *S_{pos}, S_{unl}*
M=$\emptyset$
Estimer les quantités $P(x^i = k|y=1)$ et $P(x^i = k)$ avec S_{pos} et S_{unl}.
Répéter
1) Tirer aléatoirement un paramètre $P(y=1)$
2) Calculer un modèle θ^0 à partir des estimations des $P(x^i = k|y=1)$ et $P(x^i = k)$ et du paramètre $P(y=1)$ tiré alátoirement.
3) $\forall x' \in S_{unl}$ calculer $P(y=j|x', \theta^k)$, avec le k courant et où $j \in \{0,1\}$
4) Maximiser $Q(\theta^{k+1}, \theta^k)$ (cf section 2.3.2 pour le détail des calculs)
5) Itérer à l'étape 3 jusqu'à convergence
6) Insérer le modèle θ final dans M
Choisir un modèle θ dans M de vraisemblance maximale.
Sortie : *un modèle θ*

On peut également dériver de cet algorithme un cinquième algorithme en remplaçant le tirage aléatoire de $P(y=1)$ à la phase 1 de l'algorithme par l'estimation de $P(y=1)$ donnée par la formule (2). Dans ce cas, la boucle répéter est inutile. Nous noterons **Algorithme 5 (EM+NB asym. + (2))** cette variante.

4 Résultats expérimentaux sur des données artificielles

4.1 Exemple

Cette section présente un exemple de déroulement de l'algorithme 4 et expose le protocole expérimental utilisé dans le cadre des expériences sur des données artificielles.

Le modèle cible θ_c est tiré aléatoirement, il satisfait l'hypothèse naïve de Bayes. Les données possèdent 50 attributs binaires, l'ensemble S_{pos} contient 20 données $(x_i, 1),\ i \in \{1, ..., 20\}$, tirées aléatoirement selon les distributions $P(x|y = 1)$ du modèle cible, l'ensemble S_{unl} contient 1000 données $x'_i,\ i \in \{1, ..., 1000\}$, tirées aléatoirement selon les distributions $P(x)$ et enfin, pour tester la performance des modèles inférés, un ensemble S_{test} contenant 1000 données $(x''_i, y''_i), i \in \{1, ..., 1000\}$, $y''_i \in \{0, 1\}$, est généré selon les distributions $P(y = 1)$, $P(x|y = 1)$ et $P(x|y = 0)$.

Les trois critères observés lors du déroulement de l'algorithme sont : le paramètre $P(y = 1)$, le taux d'erreur sur les données test des différents modèles et la log-vraisemblance des modèles inférés sur l'ensemble S_{test}. Le modèle θ_c généré aléatoirement prend les valeurs suivantes pour ces paramètres : $P(y = 1) = 0,5798$, $Erreur(\theta_c, S_{test}) = 0,045$ et $l(\theta_c, S_{test}) = -5651,84$.

La phase d'initialisation est numérotée 0. Le paramètre $P(y = 1)$ tiré aléatoirement à la phase 1 de l'algorithme est indiqué en gras. Chaque étape correspond à une phase complète d'itération (indices 3, 4 et 5 de l'algorithme 4). La dernière ligne, marquée en gras, indique la dernière étape avant convergence.

étape k	P(y=1)	log-vraisemblance(θ^k,S_{test})	Erreur(θ^k,S_{test})
0	**0,7868**	-6813,95	0,248
1	0,8286	-5909,26	0,188
2	0,7762	-5804,40	0,124
3	0,7221	-5735,62	0,111
4	0,6744	-5694,87	0,046
5	0,6368	-5673,13	0,046
6	0,6104	-5663,93	0,045
7	0,5934	-5661,03	0,045
8	0,5830	-5660,45	0,044
9	**0,5767**	**-5660,38**	**0,045**

Tableau 1

Une seule itération de la boucle "répéter" est indiquée, nous avons constaté que différents $P(y = 1)$ tirés aléatoirement menaient la plupart du temps au même modèle.

4.2 Comparaison de l'algorithme naïf de Bayes dans les cas semi-supervisés classique et asymétrique

Cette section présente les résultats expérimentaux obtenus sur des données générées artificiellement selon le protocole exposé section 4.1. Différentes tailles des ensembles S_{lab} et S_{unl} et différents nombres d'attributs binaires par donnée sont testés. Ces résultats permettent de comparer les performances des algorithmes 2, 3, 4 et 5.

L'algorithme 2 utilise un ensemble S_{lab} de données étiquetées et un ensemble S_{unl} de données dont la classe n'est pas connue, les tailles de ces ensembles sont signalées dans le tableau 2. Les algorithmes 3, 4 et 5 utilisent les données positives de l'ensemble S_{lab} et le même ensemble S_{unl}. La taille de S_{pos} varie en fonction du paramètre $P(y = 1)$ du modèle cible et de la taille de S_{lab} et vaut approximativement $P(y = 1) * |S_{lab}|$.

Enfin, chaque résultat présenté dans le tableau 2 est une moyenne calculée sur 200 expériences. La lecture en ligne du tableau permet d'observer l'évolution des performances des algorithmes en fonction du nombre d'attributs. En colonne, elle permet d'observer les changements induits par des modifications de la taille des ensembles S_{lab} et S_{unl}, et permet également de comparer les algorithmes entre eux. Entre parenthèses sont indiqués les écarts-types des taux d'erreurs et en gras, les moyennes des vitesses de convergence apparentes pour les algorithmes utilisant la méthode E.M. (2,4 et 5), soit la moyenne du nombre d'itérations avant stabilisation.

	Nb attributs x^i	20	50
Performance Modèle cible		0.0410 (0.0265)	0.0035 (0.0038)
$\|S_{lab}\| = 50$ $\|S_{unl}\| = 100$ $\|S_{pos}\| =$ $\alpha \cdot \|S_{lab}\|$	Algo 2 EM+NB semi-supervisé	0.0787 (0.1481) **24.46**	0.0621 (0.2166) **8.22**
	Algo 3 NB semi-sup. asym.	0.1236 (0.0753)	0.1037 (0.0662)
	Algo 4 EM+NB asym.+aléa	0.0592 (0.0601) **30.70**	0.0346 (0.1040) **9.99**
	Algo 5 EM+NB asym.+(2)	0.0653 (0.0531) **22.64**	0.0465 (0.1364) **7.63**
$\|S_{lab}\| = 100$ $\|S_{unl}\| = 1000$ $\|S_{pos}\| =$ $\alpha \cdot \|S_{lab}\|$	Algo 2 EM+NB semi-supervisé	0.0536 (0.0978) **23.19**	0.0140 (0.0936) **6.53**
	Algo 3 NB semi-sup. asym.	0.0911 (0.0527)	0.0514 (0.0478)
	Algo 4 EM+NB asym.+aléa	0.0460 (0.0425) **30.83**	0.0221 (0.0962) **9.03**
	Algo 5 EM+NB asym.+(2)	0.0460 (0.0288) **24.09**	0.0170 (0.0656) **6.15**
$\|S_{lab}\| = 1000$ $\|S_{unl}\| = 5000$ $\|S_{pos}\| =$ $\alpha \cdot \|S_{lab}\|$	Algo 2 EM+NB semi-supervisé	0.0434 (0.0294) **21.61**	0.0037 (0.0040) **4.11**
	Algo 3 NB semi-sup. asym.	0.0490 (0.0285)	0.0140 (0.0182)
	Algo 4 EM+NB asym.+aléa	0.0434 (0.0294) **28.56**	0.0103 (0.0573) **7.74**
	Algo 5 EM+NB asym.+(2)	0.0440 (0.0289) **21.13**	0.0088 (0.0690) **5.14**

Tableau 2

L'algorithme 2 est celui qui utilise le plus de données, c'est l'algorithme le plus efficace lorsque le nombre de données est grand. L'estimateur (2), utilisé pour l'étape initiale de l'algorithme 5, permet d'augmenter la rapidité de convergence par rapport à un choix aléatoire de $P(y = 1)$ (algorithme 4). On observe également une amélioration rapide des performances de l'algorithme 3 lorsque le nombre de données augmente.

5 Prédiction de ponts disulfures dans les protéines

Les ponts disulfures sont des liaisons covalentes, entre deux acides aminés (cystéines) de la chaîne protéique, qui contraignent la structure 3D de cette protéine. Des travaux préliminaires ont montré que l'information recherchée, tel couple de cystéines forme-t-il un pont, est vraisemblablement dispersée et portée par des éléments très divers : environnement, propriétés chimiques, etc... Mais il est vraisemblable que cette information soit aussi en partie portée par le voisinage des cystéines. C'est cette part de l'information que nous cherchons à extraire, sans prétendre pouvoir résoudre le problème de la prédiction des ponts disulfures avec cette seule donnée.

Nous pensons qu'il est préférable de considérer un couple de cystéines non observées appariées comme un exemple de classe indéterminée plutôt que de le considérer comme un exemple *négatif*, c'est-à-dire n'étant pas compatible. En effet, chaque cystéine est contrainte à une unicité de liaison et il est pourtant probable qu'elle soit compatible avec plusieurs autres cystéines de la même protéine. C'est pourquoi nous cherchons à montrer qu'il ne faut pas considérer les couples non observés appariés comme des représentants de couples qui ne peuvent pas s'apparier.

5.1 Les données

Les données sont extraites de la *Protein Data Bank* (PDB) pour les besoins du groupe de travail de l'ACI GENOTO3D (*http ://www.loria.fr/˜guermeur/GdT/*). Le fichier de données dont nous disposons contient 227 séquences protéiques (mots sur un alphabet à 20 lettres) qui ont toutes leurs cystéines oxydées et appariées par un pont disulfure. La répartition des protéines en fonction de leur taille (nombre d'acides aminés de la séquence) est indiquée figure 1 et en fonction du nombre de ponts figure 2.

figure 1

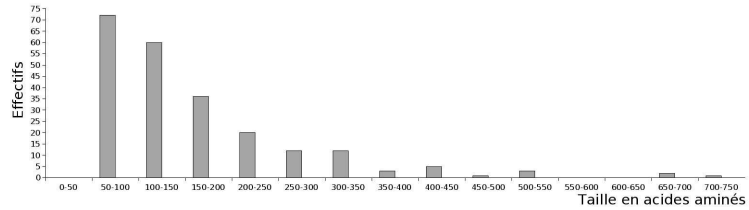
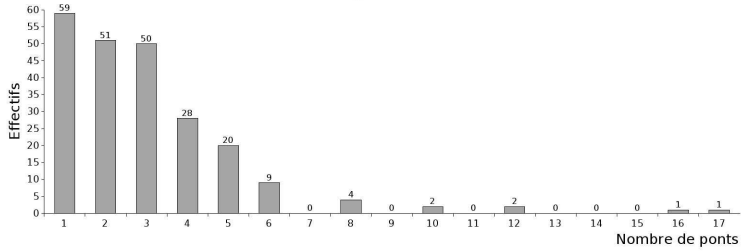


figure 2



5.2 Protocole expérimental

5.2.1 Modélisation des données

Nous cherchons à estimer des affinités locales dans les protéines. Dans le cas des ponts disulfures, ces attractions se font entre les acides aminés proches des deux cystéines appariées. C'est pourquoi nous avons extrait de chaque séquence protéique des fragments de taille fixe centrés sur les cystéines. Nous appelons ces fragments des *fenêtres* et notons $x_{-n}, .., x_{-1}, x_0, x_1, ..., x_n$ une fenêtre de rayon n (x_0 est donc une cystéine). Nous travaillons sur un alphabet de 231 lettres (nombre de couples ordonnés sur un alphabet à 21 lettres : les 20 acides aminés et un caractère pour les fins de chaîne). Pour représenter l'affinité entre deux segments f et f', trois codages sont testés :

- Codage simple : $\{(x_i, x'_i)\},\ i \in \{-n, ..., n\}, i \neq 0, x_i \in f, x'_i \in f'$
- Codage double : $\{(x_i, x'_i)\} \cup \{(x_i, x'_{-i})\},\ i \in \{-n, ..., n\}, i \neq 0, x_i \in f, x'_i \in f'$
- Codage croisé : $\{(x_i, x'_j)\},\ i, j \in \{-n, ..., n\}, x_i \in f, x'_i \in f'$

Une donnée est donc un couple de fenêtres representé par 231 attributs n-aires. Chacun de ces attributs représentant le nombre d'occurrences du couple dans la donnée.

5.2.2 Protocole d'apprentissage

Pour une protéine contenant n ponts, on compte $n(2n-1)$ couples de fenêtres potentiellement en interaction. Si un couple est observé comme formant un pont, on le considère comme un exemple positif. Quand aux autres couples, nous les considérons dans un premier temps comme des exemples ne pouvant pas former un pont, et dans un deuxième temps comme des exemples non étiquetés. Pour le premier cas, nous avons utilisé l'algorithme naïf de Bayes (section 2.1), et pour le deuxième l'algorithme 4.

L'apprentissage se fait sur des protéines ayant le même nombre n de ponts. Nous avons étudié $n = 2, 3, 4$ et 5. Le cas $n = 1$ étant trivial, il est pas étudié. Pour $n > 5$, nous ne disposons pas d'assez de données pour que les résultats soient significatifs.

5.2.3 Protocoles de test

Pour tester la pertinence des estimations d'affinité issues des deux algorithmes, nous proposons un protocole de test ne prennant en compte que cette information et permettant de comparer la qualité des estimations des deux algorithmes :

- calculer pour chaque couple de fenêtres d'une protéine test l'affinité entre ces deux fenêtres dans le modèle généré par l'algorithme d'apprentissage ;
- trouver la configuration la plus vraisemblable. Cela revient à trouver dans un graphe complet le couplage parfait de poids maximal, où les sommets sont les fenêtres d'une protéine et les arêtes les affinités. Ceci se fait en temps polynomial.

Nous avons effectué des validations croisées 10-folds pour chacun des codages proposés. Nous comparons ces résultats avec ceux d'un tirage aléatoire d'une configuration de ponts. Pour une protéine contenant n ponts, l'espérance mathématique du nombre de ponts correctement prédits par un choix aléatoire est $\frac{n}{2n-1}$. Ce résultat a été utilisé dans d'autres études sans jamais avoir été démontré. Nous l'avons prouvé mais notre démonstration est fastidieuse, aussi nous avons choisi de ne pas la faire figurer.

5.3 Résultats expérimentaux

Les performances des deux algorithmes sur les données biologiques sont maximales pour le codage croisé et très inférieures pour les autres codages. Nous donnons donc les résultats pour le codage croisé. Le tableau suivant présente les moyennes des résultats obtenus sur des séries de 100 expériences faites selon le protocole présenté section 5.3.

Nb de ponts/cystéines par protéine	2/4	3/6	4/8	5/10
Nb de protéines	51	50	28	20
Nb et % de ponts correctement prédits aléatoirement (espérance)	34 33,33%	30 20%	16 14,3%	11 11,1%
Nb et % de ponts correctement prédits Algorithme NB (supervisé)	41 40,2%	26,25 17,5%	14,22 12,7%	5,8 5,8%
Nb et % de ponts correctement prédits Algorithme 4 (semi-sup. asymétrique)	60 **58,8%**	50,1 **33,4%**	18,26 **16,3%**	13,2 **13,2%**

Résultats expérimentaux sur les données biologiques

Les résultats connus pour ce problème d'apprentissage sont (Fariselli & Casadio, 2001; Fariselli *et al.*, 2002; Vullo & Frasconi, 2004). Les meilleurs de ces résultats (Fariselli *et al.*, 2002) sont plus élevés que les notres. Ces résultats ont été obtenus par des méthodes plus sophistiquées (réseaux de neurones récursifs), avec plus de données, et en intégrant d'autres informations comme l'information évolutionnaire, c'est-à-dire un codage des segments selon des profils. Il est donc difficile de comparer nos résultats aux leurs, mais voici un tableau synthétique de leurs résultats à titre indicatif :

Nb de ponts par protéine	2 ponts	3 ponts	4 ponts	5 ponts
Nb de protéines	156	146	99	45
% de ponts correctement prédits	**73**	**56**	**37**	**30**

Résultats obtenus par (Fariselli *et al.*, 2002)

Néanmoins, nos résultats sont suffisants pour conclure sur deux points importants :
- l'apprentissage semi-supervisé asymétrique donne des résultats tout à fait satisfaisants (données artificielles et biologiques), ce qui nous encourage à poursuivre nos travaux dans cette voie ;
- notre hypothèse semble vérifiée : il est préférable de considérer les couples de cystéines non appariées comme des exemples non étiquetés plutôt que négatifs. Cette hypothèse devrait pouvoir être intégrée à des méthodes plus sophistiquées (réseaux de neurones, SVMs) de façon à exploiter au mieux l'information locale.

6 Conclusion

Nous montrons dans cet article que le problème de l'apprentissage semi-supervisé asymétrique lorsque les attributs descriptifs suivent l'hypothèse naïve de Bayes est bien posé. Nous fournissons un estimateur consistant qui permet d'identifier à la limite le modèle cible. Nous proposons également un algorithme itératif de construction de modèles basé sur le critère du maximum de vraisemblance.

Les résultats obtenus sur des données biologiques étayent une hypothèse biologique qui se veut originale quant à la façon de modéliser les données. Nous cherchons actuellement à appliquer ce procédé à d'autres données (brins beta en particulier) ainsi qu'à améliorer nos résultats sur les ponts disulfures en intégrant plus d'informations sur ces ponts. Nous essayons également d'intégrer les estimations d'affinités locales issues de notre méthode à d'autres méthodes d'apprentissage comme les SVM.

7 Remerciements

Je tiens à remercier François Denis (LIF, Marseille) et Cécile Capponi (LIF, Marseille - LMGM, Toulouse) pour leur aide et leurs conseils durant cette étude. Mais également Liva Ralaivola (LIF, Marseille), Christophe Geourjon (IBCP, Lyon) et Laurent Brehelin (LIRMM, Montpellier) pour leur participation et les diverses idées proposées.

Références

DEMPSTER A., N.M.LAIRD & D.B.RUBIN (1977). Maximum likelihood from incomplete data via the em algorithm. In *Journal of the Royal Statistical Society*, p. 39 :1–38.

DENIS F. (1998). Pac learning from positive statistical queries. In *The 9th International Workshop on Algorithmic Learning Theory*.

DENIS F., DECOMITE F., GILLERON R. & LETOUZEY F. (1999). Positive and unlabeled examples help learning. In *The 10th International Workshop on Algorithmic Learning Theory*.

DENIS F., GILLERON R., LAURENT A. & TOMMASI M. (2003). Text classification and co-training from positive and unlabeled examples. In *Proceedings of the ICML 2003 Workshop : The Continuum from Labeled to Unlabeled Data*, p. 80–87.

DOMINGOS P. & PAZZANI M. (1996). Simple bayesian classifiers do not assume independance. In A. P. . M. PRESS, Ed., *Proceedings of the Thirteenth National Conference on Artificial Intelligence and the Eighth Innovative Applications of Artificial Intelligence Conference*.

FARISELLI P. & CASADIO R. (2001). Prediction of disulfide connectivity in proteins. In *Bioinformatics*, number 17(10), p. 957–964.

FARISELLI P., MARTELLI P. & CASADIO R. (2002). A neural network-based method for predicting the disulfide connectivity in proteins. In *Proceedings of KES 2002, Knowledga based intelligent information engineering systems and allied technologies*, number 1, p. 464–468.

GEIGER D., HECKERMAN D., KING H. & MEEK C. (2001). Stratified exponential families : Graphical models and model selection. In *The Annals of Statistics*, number 29(2), p. 505–529.

HASTIE T., TIBSHIRANI R. & FRIEDMAN J. (2001). *The elements of statistical learning*.

LIU B. & LI X. (2003). Learning to classify text using positive and unlabeled data. In *Proceedings of Eighteenth International Joint Conference on Artificial Intelligence (IJCAI)*.

MCCALLUM A., THRUN S. & MITCHELL T. (1999). Text classification from labeled and unlabeled documents using e.m.

VULLO A. & FRASCONI P. (2004). Disulfide connectivity prediction using recursive neural networks and evolutionary information. In *Bioinformatics*, number 20(5), p. 653–659.

WHILEY M. & TITTERINGTON D. (2002). Model identifiability in naive bayesian networks. In *Technical Report*.

Approximation de collections de concepts formels par des bi-ensembles denses et pertinents

Jérémy Besson[1,2], Céline Robardet[3] et Jean-François Boulicaut[1]

[1] INSA Lyon, LIRIS CNRS UMR 5205, F-69621 Villeurbanne cedex, France
http://liris.cnrs.fr

[2] UMR INRA/INSERM 1235, F-69372 Lyon cedex 08, France

[3] INSA Lyon, PRISMA, F-69621 Villeurbanne cedex, France
http://prisma.insa-lyon.fr

Résumé : Le calcul de concepts formels, et plus généralement l'usage des treillis de Galois pour l'extraction de connaissances, a motivé de très nombreuses recherches. Grâce à des progrès algorithmiques récents, ces techniques fournissent des motifs particulièrement intéressants pour l'analyse de grandes matrices codant l'expression de milliers de gènes dans des situations biologiques variées. Dans cet article, nous considérons le contexte réaliste, notamment en biologie, où les concepts formels reflètent des associations trop fortes et donc très sensibles au bruit dans les données. Nous étudions l'extraction de bi-ensembles denses et pertinents pour approximer des collections de concepts formels. Le travail est formalisé dans le cadre de l'extraction de motifs sous contraintes par des algorithmes complets. Plusieurs validations expérimentales confirment la valeur ajoutée de notre approche.

Mots-clés : Découverte de connaissances, extraction de motifs sous contraintes, concepts formels, bioinformatique.

1 Introduction

L'extraction de concepts formels dans des contextes booléens et plus généralement l'usage des treillis de Galois pour l'extraction de connaissances ont motivé de nombreuses recherches. Les contextes booléens, également appelés données transactionnelles[1], se retrouvent dans de nombreuses applications. Ainsi, nous travaillons à l'analyse du transcriptome (étude des mécanismes de régulation des gènes chez un organisme vivant) après codage de propriétés d'expression booléennes pour des (dizaines de) milliers de gènes dans des situations biologiques variées. En effet, des techniques

[1]Des données transactionnelles sont un multi-ensemble d'items. Ce type de données souvent étudié en "data mining", correspond à de (grandes) matrices booléennes où les lignes définissent les transactions et les colonnes représentent les items : la présence d'un item dans une transaction est codée par la valeur vrai.

expérimentales comme celles des puces ADN permettent de quantifier le niveau d'expression des gènes (voir, e.g., la matrice de gauche de la figure 1) et dont on peut dériver des données booléennes d'expression (e.g., la matrice de droite de la figure 1). Cette dernière code le fait que les gènes ont ou pas un fort niveau d'expression (ici une valeur >1.5[2]). Dans de tels contextes booléens, un concept formel, ou rectangle maximal de valeurs 1 (vrai), représente un motif a priori intéressant pour les biologistes : il informe sur une association forte entre un ensemble maximal de gènes qui sont co-exprimés et un ensemble maximal de situations biologiques donnant lieu à cette co-expression. L'extraction de tels motifs fournit alors des collections de modules de transcription potentiels permettant d'accélerer la découverte de nouvelles voies de régulation (Besson *et al.*, 2004b), i.e., l'un des objectifs majeurs de l'analyse du transcriptome.

	Gènes			
	g_1	g_2	g_3	g_4
s_1	1.8	2.3	1.6	2.0
s_2	2.1	2.4	0.3	1.1
s_3	1.1	1.6	0.2	0.1
s_4	0.3	0.3	2.1	1.1
s_5	0.25	0.5	0.5	1.0

	Gènes			
	g_1	g_2	g_3	g_4
s_1	1	1	1	1
s_2	1	1	0	0
s_3	0	1	0	0
s_4	0	0	1	0
s_5	0	0	0	0

FIG. 1 – Matrice d'expression de gènes (gauche) et une matrice booléenne $\mathbf{r}_1$ (droite)

Par définition, les concepts formels sont construits sur des ensembles fermés. En marge des algorithmes de calcul de concepts formels (voir (Fu & Nguifo, 2004) pour une synthèse récente), de nombreux chercheurs ont proposé des algorithmes de calcul d'ensembles fermés dits fréquents qui peuvent désormais s'appliquer à de très grandes matrices booléennes (Pasquier *et al.*, 1999; Pei *et al.*, 2000; Zaki & Hsiao, 2002; Goethals & Zaki, 2003). On peut alors calculer des collections de concepts fréquents au sens de (Stumme *et al.*, 2002) : seuls les concepts dont l'un des ensembles est suffisamment grand sont extraits. En s'intéressant aux dimensions très particulières des matrices d'expression booléennes (peu de lignes et de très nombreuses colonnes), (Rioult *et al.*, 2003) montre qu'il est possible d'utiliser n'importe quel algorithme efficace de calcul d'ensembles fermés fréquents[3] sur la plus petite des deux dimensions et ainsi calculer tous les concepts formels dans des données d'expression typiques. Pour traiter des cas plus difficiles, i.e., lorsqu'aucune des deux dimensions n'est suffisamment petite ou lorsque la densité du contexte (nombre de valeurs 1) est trop importante pour les algorithmes existants, nous avons proposé D-MINER, un algorithme complet d'extraction de concepts formels sous contraintes (Besson *et al.*, 2004a). Il permet d'exploiter efficacement les contraintes monotones sur les deux dimensions des concepts formels (e.g., une taille minimale pour chacun des deux ensembles, une "surface minimale", des contraintes d'inclusion).

Nous avons maintenant des preuves de l'intérêt des concepts formels pour l'analyse

[2]Il s'agit d'un codage naïf mais des approches plus réalistes ont été étudiées (Pensa *et al.*, 2004).

[3]On utilise ici avec un seuil de fréquence nulle.

du transcriptome et la découverte de connaissances biologiques (Besson *et al.*, 2004b; Meugnier *et al.*, 2005).

Cependant, dans un concept formel, on capture une association très forte entre un ensemble de gènes et un ensemble de situations. Intuitivement, un concept n'accepte aucune exception. Si le concept $c_1 = (\{s_1, s_2, s_3\}, \{g_1, g_2, g_3, g_4\})$ est considéré comme traduisant une association réelle et si, dans les données, g_3 ne vérifie plus la propriété booléenne pour s_2, alors on trouvera les deux concepts $(\{s_1, s_2, s_3\}, \{g_1, g_2, g_4\})$ et $(\{s_1, s_3\}, \{g_1, g_2, g_3, g_4\})$ mais pas le concept c_1. En fait, la présence de valeurs "indûment" mises à 0 va faire exploser le nombre de concepts formels à extraire. Notons également que l'on aura des problèmes avec des valeurs codées par 1 alors qu'elles auraient du prendre la valeur 0. Dans ces contextes bruités, non seulement les extractions peuvent devenir impossibles, mais aussi les interprétations des motifs calculés sont très difficiles. En d'autres termes, nous sommes en présence d'une très grande sensibilité au bruit. Or, non seulement les données d'expression numériques sont bruitées du fait de la complexité des techniques de mesure, mais aussi le prétraitement de codage des propriétés booléennes à partir des données numériques peut introduire du bruit.

Dans cet article, nous proposons de travailler avec un nouveau type de motif : des bi-ensembles contenant un nombre borné de 0 par ligne et par colonne, et tel que chaque ligne (resp. colonne) soit suffisamment différente de chaque ligne (resp. colonne) extérieure sur l'ensemble des colonnes (resp. lignes) du bi-ensemble. Nous montrons que ce type de motif, appelé *bi-ensemble dense et pertinent*, est plus robuste au bruit et permet en pratique de concentrer davantage d'information pertinente dans des collections de motifs plus petites.

Dans la section 2 nous présentons quelques travaux connexes. La section 3 formalise notre problème dans le cadre de l'extraction sous contraintes. Dans la section 4, nous décrivons succinctement l'algorithme développé pour l'extraction de tous les bi-ensembles denses et pertinents. La section 5 s'intéresse aux résultats expérimentaux obtenus, notamment dans le cas de données biologiques réelles. Nous montrons que même dans le cas où le calcul de tous les bi-ensembles denses et pertinents est trop difficile, on peut utiliser l'algorithme proposé pour étudier les extensions de certains concepts. Enfin, nous concluons dans la section 6.

2 Travaux connexes

Les récentes techniques de bi-partitionnement tendent à fournir des rectangles plus robustes au bruit mais au moyen de recherches heuristiques (optimisations locales) et surtout sans recouvrement (Dhillon *et al.*, 2003; Robardet, 2002). D'autres approches ont été proposées dans la communauté de l'extraction de motifs sous contraintes. Dans (Yang *et al.*, 2001), les auteurs étendent la définition des ensembles fréquents[4] à des ensembles tolérants au bruit. Ils proposent un algorithme par niveau pour les calculer. Malheureusement, ces motifs ne peuvent pas être extraits facilement car les contraintes qui les définissent ne sont ni anti-monotones ni monotones relativement à l'inclusion

[4]Dans notre contexte, un ensemble fréquent correspond à un ensemble de gènes suffisamment co-exprimés au regard d'un nombre minimal de situations biologiques impliquées.

ensembliste, des propriétés essentielles pour rendre les extractions faisables. Ils utilisent donc un algorithme glouton calculant une solution incomplète. Dans (Seppänen & Mannila, 2004), les auteurs recherchent une contrainte anti-monotone. Ils proposent un algorithme par niveau pour calculer les ensembles qui ont une densité de valeurs 1 supérieure à δ dans au moins σ situations. L'anti-monotonicité est obtenue en exigeant que tous leurs sous-ensembles vérifient également cette contrainte. L'extension de tels ensembles denses à des bi-ensembles est difficile : les correspondances qui associent les gènes aux situations biologiques, et réciproquement, ne sont ni croissantes ni décroissantes. En effet, l'ensemble des situations biologiques associé à un ensemble de gènes n'est pas nécessairement inclus dans celui de ses sur-ensembles. Dans (Gionis *et al.*, 2004), les auteurs calculent des motifs ("geometrical tiles") qui sont des rectangles denses (ayant une densité de valeurs 1 supérieure à un seuil fixé). Pour extraire ces motifs, ils utilisent un algorithme non déterministe d'optimisation locale qui ne garantit pas la qualité globale des motifs extraits. Ils exigent qu'il existe un ordre sur les deux dimensions de la matrice : les rectangles ne sont pas considérés à des permutations près des lignes et/ou des colonnes mais doivent concerner des éléments contigus au regard des ordres considérés. Cette hypothèse n'est clairement pas acceptable dans notre contexte.

Une autre approche importante consiste à étudier de façon systématique la notion de représentation condensée des collections de concepts formels ou de bi-ensembles denses, qu'il s'agisse de représentations exactes ou approximatives. L'objectif est alors de ne représenter, ou mieux de ne calculer, qu'un sous-ensemble des collections tout en pouvant retrouver, plus ou moins exactement mais à un faible coût, l'ensemble de la collection. On peut vouloir, par exemple, rechercher une collection de k motifs qui approxime le mieux des collections complètes (Afrati *et al.*, 2004). L'approche des représentations condensées doit aussi intégrer des approches de "zoom" comme, par exemple, les travaux présentés dans (Ventos *et al.*, 2004) pour construire des treillis de Galois à différents niveaux d'abstraction. Cette méthode utilise une partition sur les objets qui permet de réduire le nombre de motifs extraits. Ils utilisent une partition sur les lignes et ne conservent que les concepts qui sont en "accord" avec cette partition : une situation s appartient à l'extension d'un ensemble G si $\alpha\%$ des objets de la même classe que s satisfont G et que s satisfait aussi G. Nous souhaitons pour notre part avoir une approche duale entre les situations et les gènes où aucune des deux dimensions n'est privilégiée au cours de l'extraction.

3 Définitions

Nous notons $\mathcal{G}$ l'ensemble des gènes et $\mathcal{S}$ l'ensemble des situations biologiques. Le contexte à fouiller est booléen, i.e., la représentation d'une relation $\mathbf{r} \subseteq \mathcal{S} \times \mathcal{G}$. Ces situations peuvent correspondre à des expériences de type puce ADN (voir figure 1).

3.1 Bi-ensembles

Un bi-ensemble (S, G) est un couple d'ensembles de $2^{\mathcal{S}} \times 2^{\mathcal{G}}$. Certains bi-ensembles particuliers peuvent être extraits dans des matrices booléennes comme les 1-rectangles (tous les éléments de S sont en relation avec tous les éléments de G) ou les concepts formels qui sont des 1-rectangles maximaux (en fait, S et G sont des ensembles fermés). Les nombreux travaux sur le calcul d'ensembles d'items (typiquement les ensembles fréquents utilisés pour le calcul de règles d'association (Becquet *et al.*, 2002)) peuvent être considérés comme des calculs de bi-ensembles. On associe à un ensemble de gènes toutes les situations qui le "portent" et l'on a donc un 1-rectangle particulier appelé "itemset". D'une manière duale, on peut définir un motif similaire basé sur un ensemble de situations appelé "objectset".

Nous donnons quelques rappels sur les correspondances de Galois (voir notamment (Wille, 1982)) pour formaliser notre problème.

Définition 1 (Correspondance de Galois)
Soit $\phi : \mathcal{S} \rightarrow \mathcal{G}$ et $\psi : \mathcal{G} \rightarrow \mathcal{S}$ deux opérateurs entre deux ensembles partiellement ordonnés $(\mathcal{S}, \leq_{\mathcal{S}})$ et $(\mathcal{G}, \leq_{\mathcal{G}})$. Ces opérateurs forment une correspondance de Galois si :

$$
\begin{array}{ll}
1 & \forall v, w \in \mathcal{S},\ \textit{si}\ v \leq_{\mathcal{S}} w\ \textit{alors}\ \phi(w) \leq_{\mathcal{G}} \phi(v), \\
2 & \forall i, j \in \mathcal{G},\ \textit{si}\ i \leq_{\mathcal{G}} j\ \textit{alors}\ \psi(j) \leq_{\mathcal{S}} \psi(i), \\
3 & \forall v \in \mathcal{S}, \forall i \in \mathcal{G},\ v \leq_{\mathcal{S}} \psi(\phi(v))\ \textit{et}\ i \leq_{\mathcal{G}} \phi(\psi(i))
\end{array}
$$

où $\leq_{\mathcal{S}}$ et $\leq_{\mathcal{G}}$ sont deux relations de spécialisation respectivement sur $\mathcal{S}$ et $\mathcal{G}$.

Définition 2 (Correspondances ϕ et ψ)
Si $S \subseteq \mathcal{S}$ et $G \subseteq \mathcal{G}$, ϕ et ψ peuvent être définis ainsi : $\phi(S, \mathbf{r}) = \{g \in \mathcal{G} \mid \forall s \in S,\ (s, g) \in \mathbf{r}\}$ et $\psi(G, \mathbf{r}) = \{s \in \mathcal{S} \mid \forall g \in G,\ (s, g) \in \mathbf{r}\}$. ϕ renvoie l'ensemble des gènes qui satisfont la propriété d'expression dans toutes les situations biologiques de S. ψ fournit l'ensemble des situations biologiques pour lesquels on a la propriété d'expression de tous les gènes de G. (ϕ, ψ) forme une correspondance de Galois entre $\mathcal{S}$ et $\mathcal{G}$ munis de l'inclusion ensembliste $\subseteq$ (relation de spécialisation). Nous utilisons les notations classiques $h = \phi \circ \psi$ et $h' = \psi \circ \phi$ pour désigner les opérateurs de fermeture de Galois. Un ensemble $S \subseteq \mathcal{S}$ (resp. $G \subseteq \mathcal{G}$) est dit fermé dans $\mathbf{r}$ ssi $S = h'(S, \mathbf{r})$ (resp. $G = h(G, \mathbf{r})$).

On peut maitenant formaliser les types de motifs précités.

Définition 3 (1-rectangles, ensembles et concepts formels)
Un bi-ensemble (S, G) est un 1-rectangle dans un contexte $\mathbf{r}$ ssi $\forall s \in S$ et $\forall g \in G$, $(s, g) \in \mathbf{r}$. Quand un bi-ensemble n'est pas un 1-rectangle, on dit qu'il contient des valeurs 0. Un bi-ensemble (S, G) est un concept dans $\mathbf{r}$ ssi $S = \psi(G, \mathbf{r})$ et $G = \phi(S, \mathbf{r})$. Ceci est équivalent à $S = h'(S, \mathbf{r})$ et $G = \phi(S, \mathbf{r})$ ou à $G = h(G, \mathbf{r})$ et $S = \psi(G, \mathbf{r})$. Une propriété importante de la correspondance de Galois est que chaque ensemble fermé sur l'une des deux dimensions est associé à un unique ensemble fermé de l'autre dimension.

Exemple 1
$(\{s_1\},\{g_1,g_3\})$ et $(\{s_1,s_2\},\{g_2\})$ sont des 1-rectangles dans $\mathbf{r}_1$ mais ne sont pas des concepts. Un exemple de concept dans $\mathbf{r}_1$ est $(\{s_1,s_2\},\{g_1,g_2\})$. Nous avons $h(\{g_1,g_2\},\mathbf{r}_1)=\{g_1,g_2\}$, $h'(\{s_1,s_2\},\mathbf{r}_1)=\{s_1,s_2\}$, $\phi(\{s_1,s_2\},\mathbf{r}_1)=\{g_1,g_2\}$, et $\psi(\{g_1,g_2\},\mathbf{r}_1)=\{s_1,s_2\}$. On peut associer à l'ensemble de gènes $\{g_1\}$ l'ensemble des situations $\{s_1,s_2\}=\psi(\{g_1\},\mathbf{r}_1)$ et nous pouvons alors parler du 1-rectangle $(\{s_1,s_2\},\{g_1\})$ comme d'un itemset. Notons qu'avec nos définitions, le 1-rectangle $(\{s_1,s_2\},\{g_2\})$ n'est pas un itemset : il faudrait ajouter s_3 à sa première composante.

Nous avons motivé dans l'introduction l'intérêt de travailler avec des bi-ensembles qui soient moins sensibles au bruit que les concepts formels et plus pertinents vis-à-vis des données globales. La faisabilité des extractions dépend de l'existence de contraintes monotones et anti-monotones (voir définition 4) permettant de définir les motifs recherchés. En fait, monotonicité et anti-monotonicité sont des propriétés duales qui sont très bien exploitées pour des extractions complètes de motifs sous contraintes, même en présence de grands espaces de recherche.

Définition 4 (Relation de spécialisation et monotonicité)
La relation de spécialisation $\preceq$ que nous utilisons sur les bi-ensembles de $2^{\mathcal{S}}\times 2^{\mathcal{G}}$ est définie par $(S_1,G_1)\preceq(S_2,G_2)$ ssi $S_1\subseteq S_2$ and $G_1\subseteq G_2$. Une contrainte $\mathcal{C}$ est dite anti-monotone par rapport à $\preceq$ ssi $\forall X,Y\in 2^{\mathcal{S}}\times 2^{\mathcal{G}}$ tels que $X\preceq Y, \mathcal{C}(Y)\Rightarrow\mathcal{C}(X)$. $\mathcal{C}$ est dite monotone par rapport à $\preceq$ ssi $\forall X,Y\in 2^{\mathcal{S}}\times 2^{\mathcal{G}}$ tel que $X\preceq Y, \mathcal{C}(X)\Rightarrow\mathcal{C}(Y)$.

Définition 5 (Exemple de contraintes monotones sur les bi-ensembles)
Contrainte de taille minimale : un bi-ensemble (S,G) satisfait $\mathcal{C}_{ms}(\mathbf{r},\sigma_1,\sigma_2,(S,G))$ ssi $\sharp S\geq\sigma_1$ et $\sharp G\geq\sigma_2$ où $\sharp$ désigne le cardinal d'un ensemble.
Contraintes d'inclusion : un bi-ensemble (S,G) satisfait $\mathcal{C}_{Inclusion}(\mathbf{r},X,Y,(S,G))$ ssi $X\subseteq S$ and $Y\subseteq G$.
Contrainte de surface minimale : un bi-ensemble (S,G) satisfait $\mathcal{C}_{area}(\mathbf{r},\sigma,(S,G))$ ssi $\sharp S\times\sharp G\geq\sigma$.

A la recherche de bi-ensembles denses, nous avons proposé dans (Besson *et al.*, 2005) une première approche visant à calculer des bi-ensembles ayant un nombre borné de valeurs 0. La méthode proposée consistait en un post-traitement de la collection de tous les concepts formels. L'idée était de procèder à une fusion de certains concepts de telle sorte que le nombre de valeurs 0 par ligne et par colonne soit borné. Cette contrainte étant anti-monotone suivant $\preceq$, ce procédé peut être réalisé en adaptant un algorithme d'extraction d'ensembles maximaux. Malheureusement, les motifs ainsi extraits ne sont pas munis d'une correspondance de Galois : le même ensemble de situations biologiques peut être associé à plusieurs ensembles de gènes différents. Nous proposons maintenant d'extraire un nouveau type de motif appelé *bi-ensemble dense et pertinent* muni d'une telle correspondance. Il s'agit de calculer tous les bi-ensembles qui satisfont la conjonction des contraintes introduites ci-dessous.

3.2 Bi-ensembles denses

Le concept de densité peut être envisagé sous deux angles selon que l'on mesure le nombre de 0 par ligne/colonne ou sur l'ensemble du bi-ensemble (densité forte versus faible) et selon que l'on considère ce nombre de manière absolue ou relativement à la taille du bi-ensemble (densité absolue versus relative).

La contrainte de "densité forte absolue" impose une limitation du nombre de 0 par ligne et par colonne, mais, relativement à la taille du bi-ensemble, elle borne aussi supérieurement le nombre de 0 total du bi-ensemble. De plus, lorsque le seuil de densité choisi est petit devant la taille minimale du bi-ensemble, ces bi-ensembles ne contiennent pas de lignes et de colonnes presque vides (avec presque que des 0) contrairement à ce qui peut se produire avec la densité faible.

D'autre part, on peut obtenir un résultat similaire sans devoir pousser de contrainte de taille minimale et en utilisant seulement une contrainte de "densité forte relative" : en fixant la proportion de 0 par ligne et par colonne on ne peut obtenir de ligne ou de colonne pleines de 0.

Ainsi, nous souhaitons extraire des bi-ensembles ayant un nombre maximum α de valeurs 0 et contenant au moins γ fois plus de 1 que de 0 par ligne et par colonne. Cette contrainte est notée $\mathcal{C}_d(\mathbf{r}, \alpha, \gamma, (S, G))$.

3.3 Bi-ensembles pertinents

Nous voulons extraire des bi-ensembles composés de situations biologiques ayant une densité sur les gènes du bi-ensemble supérieure à celle sur les gènes n'appartenant pas au bi-ensemble. Réciproquement, le bi-ensemble doit contenir des gènes dont la densité sur les situations biologiques du bi-ensemble est supérieure à celle des situations biologiques n'appartenant pas au bi-ensemble.

De manière plus formelle, étant donné deux paramètres δ, un bi-ensemble (S, G) est dit *pertinent* ssi

$$\begin{aligned} \max_{s \in S}(\sharp\{g \in G \mid (s,g) \notin \mathbf{r}\}) + \delta &\leq \min_{s \in \mathcal{S} \setminus S}(\sharp\{g \in G \mid (s,g) \notin \mathbf{r}\}) \\ \max_{g \in G}(\sharp\{s \in S \mid (s,g) \notin \mathbf{r}\}) + \delta &\leq \min_{g \in \mathcal{G} \setminus G}(\sharp\{s \in S \mid (s,g) \notin \mathbf{r}\}) \end{aligned}$$

Par la suite, cette contrainte sera désignée par $\mathcal{C}_s(\mathbf{r}, \delta, (S, G))$.

Par construction, plus δ augmentent, plus la différence entre la densité du bi-ensemble et chacune des situations biologiques extérieures au bi-ensemble et chacun des gènes extérieurs au bi-ensemble doit être grande.

3.4 Bi-ensembles denses et pertinents

Les contraintes $\mathcal{C}_d$ et $\mathcal{C}_s$ sont complémentaires et peuvent être utilisées conjointement pour augmenter la qualité des motifs extraits.

Etant donné les paramètres α, δ et γ, nous voulons donc calculer les bi-ensembles denses et pertinents, i.e., tous les bi-ensembles satisfaisant $\mathcal{C}_d \wedge \mathcal{C}_s$ dans $\mathbf{r}$. Nous désignons

cette collection par $\mathcal{SAT}_{\alpha\delta\gamma}$. Un bi-ensemble ($S$,$G$) $\in \mathcal{SAT}_{\alpha\delta\gamma}$ ssi :

$$\max_{s\in S}(\sharp\{g \in G \mid (s,g) \notin \mathbf{r}\}) \leq \begin{cases} \alpha \\ |G|/(\gamma+1) \\ \min\limits_{s\in\mathcal{S}\setminus S}(\sharp\{g \in G \mid (s,g) \notin \mathbf{r}\}) - \delta \end{cases}$$

$$\max_{g\in G}(\sharp\{s \in S \mid (s,g) \notin \mathbf{r}\}) \leq \begin{cases} \alpha \\ |S|/(\gamma+1) \\ \min\limits_{g\in\mathcal{G}\setminus G}(\sharp\{s \in S \mid (s,g) \notin \mathbf{r}\}) - \delta \end{cases}$$

Les paramètres α, δ et γ peuvent être différenciés selon que l'on considère ces contraintes sur les lignes et les colonnes. On notera d'un $\prime$ ces paramètres sur les colonnes.

Lorsque $\alpha = \alpha' = 0$, on retrouve des collections déjà bien étudiées :

- $\mathcal{SAT}$ est la collection des 1-rectangles lorsque $\delta = \delta' = 0$.
- $\mathcal{SAT}$ est la collection des itemsets (au sens défini dans la section 2.1) lorsque $\delta = 1$ et $\delta' = 0$.
- $\mathcal{SAT}$ est la collection des objectsets lorsque $\delta = 0$ et $\delta' = 1$.
- $\mathcal{SAT}$ est la collection des concepts formels lorsque $\delta = \delta' = 1$.

Dans le cas où $\alpha = \alpha' = 0$, ces collections correspondent aux bi-ensembles les plus denses et ayant le plus petit seuil de pertinence. Lorsque $\alpha > 0$, les collections de 1-rectangles, d'ensembles et de concepts formels sont généralisées en introduisant un certain nombre d'exceptions (valeur 0) dans les motifs.

La figure 2 montre la collection $\mathcal{SAT}$ lorsque $\alpha = 5$, $\alpha' = 4$, $\delta = \delta' = 1$ et $\gamma = \gamma' = 0$ pour $\mathbf{r}_1$ ordonnée par la relation $\preceq$. Chaque niveau indique le nombre maximum d'exceptions par ligne et par colonne. Par exemple, si une seule exception est autorisée ($\alpha = \alpha' = 1$) et avec $\delta = \delta' = 1$, cinq motifs sont extraits.

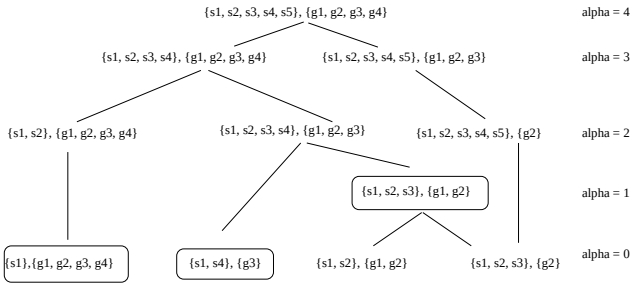


FIG. 2 – Motifs de $\mathbf{r}_1$ avec $\delta = 1$ et $\gamma = 0$. Les motifs entourés sont ceux de $\mathcal{M}_{110}$.

Il peut être pertinent d'étendre les motifs de base (itemset et concepts) avec des exceptions de telle sorte qu'ils conservent les propriétés de maximalité associées à ces motifs au sens de la correspondance de Galois. Cette propriété est très importante car elle permet de mieux appréhender la collection extraite, c'est le cas en particulier pour les biologistes. Pour préserver les correspondances de Galois, nous introduisons une nouvelle contrainte notée $\mathcal{C}_m$.

Définition 6 (Contrainte de maximalité $\mathcal{C}_m$)
Un bi-ensemble (X,Y) $\in \mathcal{SAT}_{\alpha\delta\gamma}$ *satisfait* $\mathcal{C}_m$ *dans* $\mathbf{r}$ *ssi :*
- $\delta = 1$ *et* $\delta' = 0 \Rightarrow \nexists\, (X', Y') \in \mathcal{SAT}_{\alpha\delta\gamma}$ *tel que* $Y = Y'$ *et* $X \subset X'$
- $\delta = 0$ *et* $\delta' = 1 \Rightarrow \nexists\, (X', Y') \in \mathcal{SAT}_{\alpha\delta\gamma}$ *tel que* $X = X'$ *et* $Y \subset Y'$
- $\delta \geq 1$ *et* $\delta' \geq 1 \Rightarrow \nexists\, (X', Y') \in \mathcal{SAT}_{\alpha\delta\gamma}$ *tel que (X,Y)* $\preceq (X', Y')$

La collection des bi-ensembles qui satisfont $\mathcal{C}_d \wedge \mathcal{C}_s \wedge \mathcal{C}_m$ est notée $\mathcal{M}_{\alpha\delta\gamma}$. Sur la figure 2, les trois motifs entourés forment la collection $\mathcal{M}_{110}$. Deux motifs de $\mathcal{SAT}_{110}$ ont été éliminés.

Le tableau 1 montre quelques collections $\mathcal{SAT}$ et $\mathcal{M}$ en fonction des paramètres α et δ.

	$\delta = 1$	
α	$\mathcal{SAT}_{\alpha\delta\gamma}$	$\mathcal{M}_{\alpha\delta\gamma}$
0	$\{\{s_1\},\ \{g_1, g_2, g_3, g_4\}\}$ $\{\{s_1, s_4\}, \{g_3\}\}$ $\{\{s_1, s_2\}, \{g_1, g_2\}\}$ $\{\{s_1, s_2, s_3\}, \{g_2\}\}$	$\{\{s_1\},\ \{g_1, g_2, g_3, g_4\}\}$ $\{\{s_1, s_4\}, \{g_3\}\}$ $\{\{s_1, s_2\}, \{g_1, g_2\}\}$ $\{\{s_1, s_2, s_3\}, \{g_2\}\}$
1	$\{\{s_1\},\ \{g_1, g_2, g_3, g_4\}\}$ $\{\{s_1, s_4\}, \{g_3\}\}$ $\{\{s_1, s_2\}, \{g_1, g_2\}\}$ $\{\{s_1, s_2, s_3\}, \{g_2\}\}$ $\{\{s_1, s_2, s_3\}, \{g_1, g_2\}\}$	$\{\{s_1\},\ \{g_1, g_2, g_3, g_4\}\}$ $\{\{s_1, s_4\}, \{g_3\}\}$ $\{\{s_1, s_2, s_3\}, \{g_1, g_2\}\}$
	$\delta = 2$	
0	$\{\{s_1\},\ \{g_1, g_2, g_3, g_4\}\}$	$\{\{s_1\},\ \{g_1, g_2, g_3, g_4\}\}$
1	$\{\{s_1\},\ \{g_1, g_2, g_3, g_4\}\}$ $\{\{s_1, s_2, s_3\}, \{g_1, g_2\}\}$	$\{\{s_1\},\ \{g_1, g_2, g_3, g_4\}\}$

TAB. 1 – Collections $\mathcal{SAT}_{\alpha\delta\gamma}$ et $\mathcal{M}_{\alpha\delta\gamma}$ sur $\mathbf{r}_1$.

La collection $\mathcal{M}_{\alpha\delta\gamma}$ est muni d'une correspondance de Galois. En effet, dans nos applications, les ensembles de situations permettent d'expliquer l'association des gènes (la co-expression) et inversement. Ainsi, les biologistes recherchent des associations bijectives et décroissantes. Les bi-ensembles extraits vérifient cette propriété.

Propriété 1
Pour $\alpha_1 \leq \alpha$ *et* $\alpha'_1 \leq \alpha'$, δ, δ', γ *et* γ' *fixés, alors* $\forall X \in \mathcal{M}_{\alpha\alpha'\delta\delta'}$, $\exists X_1 \in \mathcal{M}_{\alpha_1\alpha'_1\delta\delta'}$ *tel que* $X_1 \preceq X$. *De plus,* $\forall X_1 \in \mathcal{M}_{\alpha_1\alpha'_1\delta\delta'}$, $\exists X \in \mathcal{M}_{\alpha\alpha'\delta\delta'}$ *tel que* $X_1 \preceq X$.

Propriété 2
Pour α, α', γ *et* γ' *fixés, et* $\delta \leq \delta_1$ *et* $\delta' \leq \delta'_1$ *alors* $\mathcal{SAT}_{\alpha\alpha'\delta_1\delta'_1} \subseteq \mathcal{SAT}_{\alpha\alpha'\delta\delta'}$.

D'après la propriété 1, plus α et α' augmentent, plus la taille de chaque motif extrait de $\mathcal{M}_{\alpha\alpha'\delta_1\delta'_1}$ augmente tout en conservant les associations extraites dans les collections avec α et α' plus petits. En pratique, une réduction importante de la taille de la collection

est observée lorsque les paramètres de l'extraction sont judicieusement choisis (voir section 5). Par conséquent, un effet de zoom est observé lorsque α et α' varient.

Les paramètres δ et δ' permettent de sélectionner les motifs les plus pertinents (voir propriété 2).

$\mathcal{M}_{0,0,0,0}$ et $\mathcal{M}_{0,0,1,1}$ correspondent respectivement aux collections des 1-rectangles et des concepts couvrant toutes les valeurs 1 de la matrice. Ainsi, d'après la propriété 1, $\forall \alpha \geq 0$, $\forall \alpha' \geq 0$ et $\delta, \delta' \in \{0, 1\}$, la collection $\mathcal{M}_{\alpha,\alpha',\delta,\delta'}$ couvre tous les 1 de la matrice.

4 Un algorithme complet

L'algorithme construit un arbre d'énumération binaire, sur les situations biologiques et les gènes, en procédant en profondeur. En s'inspirant du principe de l'algorithme DUAL-MINER (Bucila *et al.*, 2003), chaque nœud de l'arbre est constitué de trois bi-ensembles :

- $O = (O_s, O_g)$ est composé des éléments qui appartiendront aux motifs construits par cette branche,
- $N = (N_s, N_g)$ contient les éléments qui n'appartiendront pas aux motifs engendrés par cette branche,
- $P = (P_g, P_g)$ contient les éléments qui restent à énumérer.

Chaque élément de $\mathcal{S}$ et de $\mathcal{G}$ appartient à un et un seul ensemble parmi O, P et N. Les bi-ensembles O et N sont générés de $(\emptyset, \emptyset)$ au bi-ensemble $(\mathcal{S}, \mathcal{G})$ en exploitant la relation d'ordre $\preceq$.

Pour pouvoir utiliser activement les contraintes $\mathcal{C}_s$ et $\mathcal{C}_d$, on associe à chaque situation biologique s (resp. chaque gène g) deux valeurs notées min_s et max_s (resp. min_g et max_g). min_s correspond au nombre de valeurs 0 de s sur les gènes appartenant à O_g. max_s correspond au nombre de valeurs 0 de s sur les gènes de $O_g \cup P_g$. min_s et max_s correspondent respectivement aux bornes inférieure et supérieure du nombre de 0 à un niveau donné de l'énumération.

4.1 Vérification et propagation des contraintes

A tout moment, les éléments des trois ensembles O, P et N doivent vérifier les contraintes suivantes :

- soit une situation s telle que $\min_s > \alpha$ alors s doit appartenir à N_s. Ainsi, si s était dans O_s, on élague la branche. Sinon s est déplacé dans N_s. En effet, les situations qui ont plus de α valeurs 0 ne peuvent pas appartenir à un bi-ensemble solution.
- soit une situation s telle que

$$max_s < \max_{t \in O_s}\{min_t\} + \delta$$

alors s doit appartenir à O_s. Ainsi, si s appartenait à N_s, le nœud est élagué. Sinon, s est déplacé dans O_s. Dans ce cas, la situation ne contient pas suffisamment de valeurs 0 pour être à l'extérieur du bi-ensemble.

De manière tout à fait similaire, ces contraintes doivent être vérifiées sur les gènes.

D'autres contraintes peuvent également être poussées lors de l'extraction de telle sorte à élaguer l'espace de recherche ou bien à forcer l'appartenance d'un élément à O ou à N. Par exemple, les contraintes monotones et anti-monotones sur $\preceq$ peuvent être exploitées. Les contraintes monotones vont être basée sur $O \cup P$ et les contraintes anti-monotones sur O. Les définitions 7 et 8 donnent des exemples de contraintes.

Définition 7 (Exemple de contraintes monotones)
- $\mathcal{C}_{ms}(\mathbf{r}, \sigma_1, \sigma_2, (S, G))$ *si* $\sharp(O_s \cup P_s) \geq \sigma_1$ *et* $\sharp(O_g \cup P_g) \geq \sigma_2$
- $\mathcal{C}_{Inclusion}(\mathbf{r}, X, Y, (S, G))$ *si* $X \subseteq O_s \cup P_s$ *et* $Y \subseteq O_g \cup P_g$
- $\mathcal{C}_{area}(\mathbf{r}, \sigma, (S, G))$ *si* $\sharp(O_s \cup P_s) * \sharp(O_g \cup P_g) \geq \sigma$

Définition 8 (Exemple de contraintes anti-monotones)
- $\mathcal{C}_{mins}(\mathbf{r}, \sigma_1, \sigma_2, (S, G))$ *si* $\sharp(O_s) \leq \sigma_1$ *et* $\sharp(O_g) \leq \sigma_2$
- $\mathcal{C}_{Inc}(\mathbf{r}, X, Y, (S, G))$ *si* $O_s \subseteq X$ *et* $O_g \subseteq Y$

Si un nœud ne vérifie pas une de ces contraintes alors aucun de ces fils ne la vérifiera et ainsi l'espace de recherche peut être élagué. Ce type d'algorithme permet d'exploiter un grand nombre de contraintes, même des contraintes qui ne sont ni monotones ni anti-monotones sur $\preceq$ comme $\mathcal{C}_d \wedge \mathcal{C}_s$.

4.2 Optimisation

Pour des raisons d'efficacité, nous utilisons une heuristique importante pour l'énumération des gènes et des situations biologiques : l'élément e (gène ou situation biologique) utilisé pour l'énumération est celui qui possède le nombre de valeurs 0 potentiels ($\max_e$) le plus grand. Ce choix tend à réduire la taille du bi-ensemble P le plus rapidement possible. Cela diminue l'espace de recherche tout en préservant la complétude des extractions.

5 Expérimentations

5.1 Evaluation de la robustesse au bruit sur données synthétiques

Pour montrer la pertinence des $\mathcal{M}_{\alpha\alpha'\delta\delta'}$ dans les données bruitées, nous avons tout d'abord généré des jeux de données synthétiques. Notre but est de montrer que l'extraction des $\mathcal{M}_{\alpha\alpha'\delta\delta'}$ permet de retrouver les concepts, introduits dans le jeu de données avant qu'il ne soit bruité. Ainsi, les jeux de données construits sont composés de 4 concepts disjoints comportant chacun 10 éléments sur chaque dimension. Ensuite, un bruit aléatoire uniforme a été introduit dans les données, aussi bien sur les concepts qu'à l'extérieur. Nous avons généré 10 jeux de données pour chaque niveau de bruit : 5%, 10%, 15% et 20%. Le tableau 2 indique le nombre moyen suivi de l'écart-type du nombre de motifs extraits pour chaque niveau de bruit pour $\alpha = \alpha'$ variant de 0 à 3, $\delta = \delta' = 3$ et contenant au moins 4 éléments sur chaque dimension. Ces contraintes permettent de ne pas considérer les petits motifs dus au bruit et de ne conserver que

ceux qui sont très pertinents. Dans le tableau 2, nous donnons également le nombre moyen de concepts pour chaque niveau de bruit.

α		0		1		2		3	
	Nb concepts	Moy	σ	Moy	σ	Moy	σ	Moy	σ
5%	228.6	0	0	1.3	0.82	3.3	0.95	4	0
10%	663.8	0	0	0.1	0.32	1.7	1.16	3	0.94
15%	1292.5	0	0	0	0	0.4	0.70	1.3	0.95
20%	2191.7	0	0	0	0	0	0	3.1	3

TAB. 2 – Moyenne et écart-type du nombre de motifs extraits (sur 10 essais) en fonction de $\alpha = \alpha'$ et du pourcentage de bruit dans les données ($\delta = \delta' = 3$ et $\mathcal{C}_{ms}(\mathbf{r}, 4, 4, (S, G))$).

Lorsqu'il y a 5% de bruit, on retrouve systématiquement les 4 concepts originaux avec $\alpha = \alpha' = 3$. Pour un pourcentage de bruit plus élevé (10% et 15%), seulement certains des concepts originaux sont retrouvés. Lorsque le bruit est trop important (20%), le nombre de motifs extraits est assez variable (l'écart-type vaut 3). Sur certains jeux de données, quelques concepts parmi les 4 d'origine sont retrouvés ; sur d'autres jeux de données, la démultiplication du nombre de concepts réapparaît un peu. En revanche, de très nombreux concepts générés par l'introduction du bruit ont été éliminés.

5.2 Impact des paramètres sur les collections extraites

5.2.1 L'influence des paramètres α et α'

Pour voir l'influence des paramètres α et α' sur $\mathcal{M}_{\alpha\alpha'\delta\delta'}$, nous avons réalisé plusieurs extractions sur le jeu de données CAMDA (Bozdech *et al.*, 2003). Ce jeu de données montre l'évolution des niveaux d'expression de 3719 gènes (colonnes) de Plasmodium falciparum (responsable de la malaria) durant son invasion des globules rouges. La série temporelle comporte 46 mesures du niveau d'expression des gènes.

Nous avons fixé $\delta = \delta' = 1$ et nous avons fait varier $\alpha = \alpha'$ de 0 à 4. De plus, les motifs doivent satisfaire la contrainte $\mathcal{C}_{ms}(\mathbf{r}, \sigma_1, \sigma_2, (S, G))$ avec $\sigma_2 = 3$ et σ_1 qui varie de 19 à 24. Comme la contrainte de fréquence habituellement utilisée lors de l'extraction des ensembles fréquents, la contrainte $\mathcal{C}_{ms}$ permet de rendre les extractions faisables.

Le nombre de motifs extraits pour $\alpha = \alpha'$ de 0 à 2 diminue globalement. Certains motifs sont enrichis et deviennent des sur-ensembles de motifs pour $\alpha = \alpha'$ plus petits. Ensuite, pour $\alpha = \alpha' > 2$, le nombre de motifs extraits tend a augmenter de nouveau. Ceci peut s'expliquer par deux phénomènes :

- Tout d'abord, la taille de certains motifs, initialement non comptabilisés car étant trop petits, augmentent de telle sorte qu'ils satisfont la contrainte de taille
- Lorsque $\alpha \geq 3$, le nombre d'erreurs accepté par ligne est supérieur ou égal au nombre de colonnes minimum du motif, ce qui conduit à accepter des concepts pouvant avoir très peu de 1 par ligne. Cela induit une augmentation du nombre de

α	0	1	2	3	4
$\sigma_1 = 24$	0	4	4	5	5
$\sigma_1 = 23$	9	10	8	9	12
$\sigma_1 = 22$	35	23	22	24	251
$\sigma_1 = 21$	97	68	66	69	-
$\sigma_1 = 20$	241	202	197	213	-
$\sigma_1 = 19$	578	511	513	608	-

TAB. 3 – Nombre de motifs satisfaisant la contrainte $\mathcal{C}_{ms}(\mathbf{r}, \sigma_1, \sigma_2, (S, G))$ avec $\sigma_2 = 3$, σ_1 entre 19 et 24, $\delta = \delta' = 1$ et $\alpha = \alpha'$ qui varie.

motifs. En pratique, il faut imposer une contrainte de taille minimale sur les deux dimensions nettement supérieure à α et α'.

Lorsque α augmente, l'extraction des motifs denses et pertinents devient de plus en plus difficile. Nous n'avons pas réussi à extraire ces motifs pour $\alpha = \alpha' = 4$ et $\sigma_1 \leq 21$.

5.2.2 L'influence des paramètres δ et δ'

Pour montrer l'influence des paramètres δ et δ' sur $\mathcal{M}_{\alpha\alpha'\delta\delta'}$, nous avons réalisé des extractions sur un jeu de données UCI (Internet Advertisements) de dimension 3279×1555. Il ne 'agit pas d'une matrice d'expression mais nous avons cherché un contexte booléen peu dense pour mieux illustrer les variations du nombre de concepts lorsque δ et δ' augmentent.

Pour ces extractions, α et α' sont fixés à 1, δ et δ' varient de 1 à 10 et les motifs extraits (S,G) doivent satisfaire la contrainte $\mathcal{C}_{ms}(\mathbf{r}, \sigma_1, \sigma_2, (S, G))$ avec $\sigma_2 = 0$ et $\sigma_1 \in \{31, 78, 155, 330\}$.

$\delta = \delta'$	1	2	3	4	5	6	7	8	9	10
$\sigma_1 = 31$	549	56	16	7	5	5	2	2	2	2
$\sigma_1 = 78$	131	17	3	2	2	2	1	1	1	1
$\sigma_1 = 155$	43	7	1	1	1	1	1	1	1	1
$\sigma_1 = 330$	6	1	1	1	1	1	1	1	1	1

TAB. 4 – Taille des collections extraites sur le jeu de données de l'UCI :Internet Advertisements, pour $\alpha = \alpha' = 1$ sous la contrainte $\mathcal{C}_{ms}(\mathbf{r}, \sigma_1, \sigma_2, (S, G))$ avec $\sigma_2 = 0$ et $\sigma_1 \in \{31, 78, 155, 330\}$

Les extractions du tableau 4 montrent une diminution importante du nombre de concepts extraits au fur et à mesure de l'augmentation de δ et δ'.

5.3 Extension des concepts

La complexité de l'extraction des motifs denses et pertinents peut augmenter très fortement avec α et α' rendant certaines extractions infaisables. Il est néanmoins possible,

dans ce cas, d'utiliser l'algorithme présenté pour enrichir certains concepts formels intéressant l'utilisateur final. En effet, il suffit pour étendre un concept (S,G) d'extraire les motifs (S', G') de $\mathcal{M}_{\alpha\alpha'\delta\delta'}$ avec α et β supérieur à 0 et tel que (S, G) est un sur-ensemble de (S', G') (il satisfait $\mathcal{C}_{Inclusion}(\mathbf{r}, S', G', (S, G))$). Pour réduire efficacement la complexité du calcul, il faut que le concept que l'on cherche à étendre ait suffisamment d'éléments (relativement à la taille et à la densité du jeu de données utilisé). Dans ce cas, la contrainte d'inclusion devient suffisamment sélective pour réduire l'espace de recherche.

Pour illustrer ce procédé, nous avons utilisé le jeu de données CAMDA qui représente une série temporelle de 46 mesures correspondant à l'évolution du niveau d'expression des 483 gènes dont la fonction biologique est connue parmi 3719 gènes de la matrice d'origine. On peut distinguer trois phases dans le développement de Plasmodium falciparum au cours de l'infection. Elle sont appelées "ring", "trophozoite" et "shizont". Tous les concepts formels ont pu être extraits de cette matrice après discrétisation. Parmi ces 3800 concepts, on s'est intéressé à un concept contenant huit situations relatives à la phase "ring" et quatre gènes dont trois sont connus pour avoir une fonction cytoplasmique. Les gènes ayant cette fonction ont tendance à être sur-exprimés au cours de cette phase. Nous avons essayé d'étendre ce concept pour l'enrichir (voir figure 3). Par exemple en utilisant $\alpha = \alpha' = 2$ et $\delta = \delta' = 1$ on obtient un motif qui contient neuf gènes, onze situations biologiques et 7% de valeurs 0 dans le motif. Les trois situations biologiques ajoutées correspondent à la phase "ring" et parmi les cinq gènes ajoutés, quatre ont une fonction cytoplasmique. Parmi les motifs étendus de la figure 3, cinq des sept nouveaux gènes sont connus pour avoir une fonction cytoplasmique et les huit situations biologiques ajoutées appartiennent à la phase "ring". La prise en compte des exceptions dans les données a permis d'augmenter la taille du motif extrait en ajoutant des éléments cohérents d'un point de vue biologique avec ceux du concept initial.

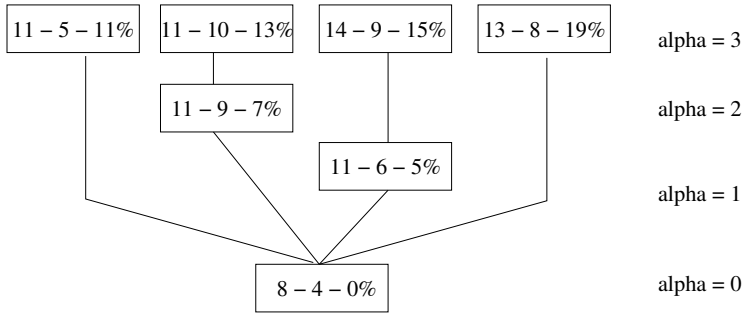


FIG. 3 – Extensions d'un concept : chaque triplet représente le nombre de situations, le nombre de gènes et la densité faible relative de 0.

6 Conclusion

Pour extraire des connaissances dans de grandes matrices booléennes, nous avons défini un nouveau type de motifs appelé bi-ensembles denses et pertinents. Cette recherche a été motivée par des applications en analyse du transcriptome où les concepts formels dans des matrices d'expression de gènes suggèrent aux biologistes des modules de transcription potentiels. Nous nous sommes alors intéressés à la trop grande sensibilité au bruit des extractions de concepts formels pour proposer l'extraction de bi-ensembles qui peuvent être vus comme des concepts formels avec un nombre borné d'exceptions (bi-ensemble dense) mais aussi avec un critère de qualité sur leurs pertinences (singularité des éléments retenus dans le bi-ensemble au regard de l'ensemble des données).

L'extraction de ce nouveau type de motifs est, dans certains cas, plus difficile en pratique que celle de tous les concepts formels. L'applicabilité de l'algorithme complet dans des contextes variés nous paraît donc peu vraisemblable. Pour autant, nous avons proposé une méthode très simple pour exploiter l'algorithme lors de l'extension de certains concepts déjà découverts. Cette direction de recherche nous parait très prometteuse dans l'optique d'une assistance à la découverte de connaissances dans des données réelles, que ce soit dans le cadre de la biologie moléculaire ou plus généralement pour le traitement de données transactionnelles bruitées, denses et/ou très corrélées (i.e., de nombreux domaines d'application où les données sont transactionnelles mais pas le classique contexte de l'analyse du "panier de la ménagère" pour lequel les données sont peu bruitées, peu denses et peu corrélées).

Remerciements

Ce travail est partiellement financé par l'ACI Masse de Données Bingo (ACI MD 46, CNRS STIC).

Références

AFRATI F. N., GIONIS A. & MANNILA H. (2004). Approximating a collection of frequent sets. In *Proceedings ACM SIGKDD'04*, p. 12–19, Seattle, WA, USA : ACM.

BECQUET C., BLACHON S., JEUDY B., BOULICAUT J.-F. & GANDRILLON O. (2002). Strong association rule mining for large gene expression data analysis : a case study on human SAGE data. *Genome Biology*, **12**. See http ://genomebiology.com/2002/3/12/research/0067.

BESSON J., ROBARDET C. & BOULICAUT J.-F. (2004a). Constraint-based mining of formal concepts in transactional data. In *Proceedings PaKDD'04*, volume 3056 of *LNAI*, p. 615–624, Sydney, Australia : Springer-Verlag.

BESSON J., ROBARDET C. & BOULICAUT J.-F. (2005). *Mining formal concepts with a bounded number of exceptions from transactional data*, In *Post-Workshop proceedings KDID'04*, volume 3377 of *LNCS*, p. 33–45. Springer-Verlag.

BESSON J., ROBARDET C., BOULICAUT J.-F. & ROME S. (2004b). Constraint-based bi-set mining for biologically relevant pattern discovery in microarray data. *Intelligent Data Analysis journal*, **9(1)**. In Press.

BOZDECH Z., LLINÁS M., PULLIAM B. L., WONG E., ZHU J. & DERISI J. (2003). The transcriptome of the intraerythrocytic developmental cycle of plasmodium falciparum. *PLoS Biol*, **1**(e5).

BUCILA C., GEHRKE J. E., KIFER D. & WHITE W. (2003). Dualminer : A dual-pruning algorithm for itemsets with constraints. *Data Mining and Knowledge Discovery*, **7**(4), 241–272.

DHILLON I., MALLELA S. & MODHA D. (2003). Information-theoretic co-clustering. In *Proceedings ACM SIGKDD 2003*, p. 1–10 : ACM.

FU H. & NGUIFO E. M. (2004). Etude et conception d'algorithmes de génération de concepts formels. In *Extraction de motifs dans les bases de données*, volume 9(3/4) of *RSTI série ISI*, p. 109–132. Hermès.

GIONIS A., MANNILA H. & SEPPÄNEN J. K. (2004). Geometric and combinatorial tiles in 0-1 data. In *Proceedings PKDD'04*, volume 3202 of *LNAI*, p. 173–184, Pisa, Italy : Springer-Verlag.

GOETHALS B. & ZAKI M. (2003). *Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations FIMI 2003*. Melbourne, USA : IEEE Computer Press.

MEUGNIER E., BESSON J., BOULICAUT J.-F., LEFAI E., DIF N., VIDAL H. & ROME S. (2005). Resolving transcriptional network from microarray data with constraint-based formal concept mining revealed new target genes of SREBP1. *Submitted*.

PASQUIER N., BASTIDE Y., TAOUIL R. & LAKHAL L. (1999). Efficient mining of association rules using closed itemset lattices. *Information Systems*, **24**(1), 25–46.

PEI J., HAN J. & MAO R. (2000). CLOSET an efficient algorithm for mining frequent closed itemsets. In *Proceedings ACM SIGMOD Workshop DMKD'00*.

PENSA R. G., LESCHI C., BESSON J. & BOULICAUT J.-F. (2004). Assessment of discretization techniques for relevant pattern discovery from gene expression data. In *Proceedings ACM BIOKDD'04 co-located with SIGKDD'04*, p. 24–30, Seattle, USA.

RIOULT F., BOULICAUT J.-F., CRÉMILLEUX B. & BESSON J. (2003). Using transposition for pattern discovery from microarray data. In *Proceedings ACM SIGMOD Workshop DMKD'03*, p. 73–79, San Diego, USA.

ROBARDET C. (2002). *Contribution à la classification non supervisée : proposition d'une méthode de bi-partitionnement*. PhD thesis, University Claude Bernard - Lyon 1, F-69622 Villeurbanne cedex.

SEPPÄNEN J. K. & MANNILA H. (2004). Dense itemsets. In *Proceedings ACM SIGKDD'04*, p. 683–688, Seattle, WA, USA : ACM.

STUMME G., TAOUIL R., BASTIDE Y., PASQUIER N. & LAKHAL L. (2002). Computing iceberg concept lattices with titanic. *Data and Knowledge Engineering*, **42**, 189–222.

VENTOS V., SOLDANO H. & LAMADON T. (2004). Treillis de galois alpha. In *Actes CAp 2004*, p. 175–190, Montpellier, F.

WILLE R. (1982). Restructuring lattice theory : an approach based on hierarchies of concepts. In I. RIVAL, Ed., *Ordered sets*, p. 445–470. Reidel.

YANG C., FAYYAD U. & BRADLEY P. S. (2001). Efficient discovery of error-tolerant frequent itemsets in high dimensions. In *Proceedings ACM SIGKDD'01*, p. 194–203, San Francisco, CA, USA : ACM Press.

ZAKI M. J. & HSIAO C.-J. (2002). CHARM : An efficient algorithm for closed itemset mining. In *Proceedings SIAM DM'02*, Arlington, USA.

Discovering "Factual" and "Implicative" generic association rules

Gh. Gasmi[1], S. Ben Yahia[1,2], E. Mephu Nguifo[2] and Y. Slimani[1]

[1] Départment des Sciences de l'Informatique
Faculté des Sciences de Tunis
Campus Universitaire, 1060 Tunis, Tunisie.
{sadok.benyahia,yahya.slimani}@fst.rnu.tn
[2] Centre de Recherche en Informatique de Lens-IUT de Lens
Rue de l'Université SP 16, 62307 Lens Cedex
mephu@cril.univ-artois.fr

Abstract : Le nombre de plus en plus grandissant de rËgles associatives extraites– mÍme ‡ partir de contextes de taille raisonnable– a encouragÈ le dÈveloppement de mÈthodes et/ou techniques pour rÈduire la liste des rËgles associatives extraites. Dans ce contexte, la batterie de rÈsultats thÈoriques fournie par l'Analyse de Concepts Formels (AFC) a permis de dÈgager un "noyau irrÈductible" de rËgles associative, mieux connu sous le nom de *base gÈnÈrique*. A partir de cet ensemble condensÈ, de taille rÈduite, nous sommes en mesure d'infÈrer toutes les rËgles associatives valides par la biais d'un systËme axiomatique adÈquat. Dans ce papier, nous proposons une nouvelle base gÈnÈrique, convoyant une nouvelle caractÈrisation: RËgles "Factuelles" et RËgles "implicatives", sans nÈgliger l'interprÈtation sÈmantique de ce type de connaissance. Nous proposons ainsi un systËme axiomatique valide et complet, permettant d'infÈrer **toutes** les rËgles associatives valides. Les rÈsultats des expÈrimentations effectuÈes sur des contextes d'extraction rÈels ont montrÈ une importante rÈduction en terme de compacitÈ de la taille de l'ensemble des rËgles associatives extraites.

Mots-clÈs: Base gÈnÈrique de rËgles associatives, Connaissance factuelle et implicative, Connexion de Galois, SystËme Axiomatique.

1 Introduction

Extracting "condensed" patterns is grasping the interest of the Data Mining community. In fact, avoiding the extraction of an overwhelming knowledge is of primary importance as it guaranties extra value knowledge usefulness and reliability. In the association rule extraction topic, rule selection is mainly based on user-defined syntactic templates or on user-defined statistical interestingness metrics (Ohsaki *et al.*, 2004). Lossless selection is mainly based on the determination of a generic subset of all association rules, called *generic basis*, from which the remaining (redundant) association rules are generated.

Compared to the stampede algorithmic effort for extracting frequent (condensed) patterns, a few works only focused on extracting generic bases, from which only that defined by Bastide *et al* is considered as informative, *i.e.*, support and confidence of inferred rules can be retrieved exactly. However, as a drawback, such generic basis may be over-sized specially for dense datasets.

In this paper, we introduce a new generic basis of association rules called $\mathcal{IGB}$. Through $\mathcal{IGB}$, we introduce a novel characterization of generic association rules instead of the traditional one, *i.e.*, exact and approximative. In fact, we distinguish the "*factual*" from the "*implicative*" generic association rule. Indeed, a factual generic association rule, fulfilling the premise part emptiness, allows to highlight item correlations without any conditionality. However, for an implicative generic association rule, where the premise part is not empty, item correlations are conditioned by the existence of those of premise items.
The introduced $\mathcal{IGB}$ generic basis fulfills the "informativeness property", *i.e.*, the support and the confidence of the derived association rules can be retrieved accurately. In order to derive valid association rules from the $\mathcal{IGB}$ basis, we introduce an axiomatic system, and show that it is valid and complete.

We conducted several experiments on typical benchmarking datasets to assess the $\mathcal{IGB}$ compactness. Reported statistics highlighted that $\mathcal{IGB}$ is more compact than informative generic bases found in the literature. The introduced generic rule characterization permitted to explain the "atypical" behavior of the variation of the reported generic association rule set cardinality versus the variation of the *minconf* value, *i.e.*, the number of the reported generic association rules does not necessarily decrease when the *minconf* value increases.

The remainder of the paper is organized as follows: Section 2 sketches the basic mathematical foundations for the derivation of association rule generic bases. We devote section 3 to a review of the literature relating to the extraction of generic bases. Section 4 introduces a novel informative basis of generic association rules and the associated axiomatic system. Results of the experiments carried out on real-life databases are reported in section 5. The conclusion and future work are presented in section 6.

2 Mathematical background

In the following, we briefly recall some key results from the Formal Concept Analysis (FCA) (Ganter & Wille, 1999) and its connection with generic association rules mining.

Formal context: A formal context (extraction context) is a triplet $\mathcal{K} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$, where $\mathcal{O}$ represents a finite set of objects (or attributes), $\mathcal{I}$ is a finite set of items and $\mathcal{R}$ is a binary (incidence) relation (*i.e.*, $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{I}$). Each couple $(o, i) \in \mathcal{R}$ expresses that the transaction $o \in \mathcal{O}$ contains the item $i \in \mathcal{I}$.

We define two functions, summarizing links between subsets of objects and subsets of attributes induced by R, that map sets of objects to sets of attributes and *vice versa*. Thus, for a set $O \subseteq \mathcal{O}$, we define $\phi(O) = \{i \mid \forall o, o \in O(o, i) \in \mathcal{R}\}$; and for $I \subseteq \mathcal{I}$, $\psi(I) = \{o \mid \forall i, i \in I(o, i) \in \mathcal{R}\}$. Both functions ϕ and ψ form a Galois connection between the sets $\mathcal{P}(\mathcal{I})$ and $\mathcal{P}(\mathcal{O})$ (Barbut & Monjardet, 1970). Consequently, both compound operators of ϕ and ψ are closure operators, particularly $\omega = \phi \circ \psi$ is a

closure operator.
Frequent closed itemset (FCI): An itemset $I \subseteq \mathcal{I}$ is said to be *closed* if $I = \omega(I)$, and is said to be *frequent* with respect to the *minsup* threshold if support(I)$= \frac{|\psi(I)|}{|\mathcal{O}|} \geq$ *minsup* (Pasquier *et al.*, 1999).
Formal Concept: A formal concept is a pair $c = (O, I)$, where O is called *extent*, and I is a closed itemset, called *intent*. Furthermore, both O and I are related through the Galois connection, *i.e.*, $\phi(O) = I$ and $\psi(I) = O$.

Minimal generator: An itemset $g \subseteq \mathcal{I}$ is said to be *minimal generator* of a closed itemset I, if and only if $\omega(g) = I$ and there is no such g$' \subset g$ such that $\omega(g') = I$ (Bastide *et al.*, 2000).
Equivalence classes: The closure operator ω induces an equivalence relation on items power set, *i.e.*, the power set of items is portioned into disjoint subsets (also called *classes*). In each distinct class, all elements are equal support value. The set of minimal generators is the smallest set of incomparable elements in this equivalence, while the closed itemset is the largest one.
Galois lattice: Given a formal context $\mathcal{K}$, the set of formal concepts $\mathcal{C}_\mathcal{K}$ is a complete lattice $\mathcal{L}_c = (\mathcal{C}_\mathcal{K}, \leq)$, called the *Galois (concept) lattice*, when $\mathcal{C}_\mathcal{K}$ is considered with inclusion between closed itemsets (Ganter & Wille, 1999; Barbut & Monjardet, 1970). Such structure must verify these two properties:

- A partial order on formal concepts is defined as follows $\forall\, c_1, c_2 \in \mathcal{C}_\mathcal{K}$, $c_1 \leq c_2$ iif $intent(c_2) \subseteq intent(c_1)$, or equivalently $extent(c_1) \subseteq extent(c_2)$. The partial order is used to generate the lattice graph, called *Hasse diagram*, in the following manner: there is an arc (c_1, c_2), if $c_1 \preceq c_2$ where $\preceq$ is the transitive reduction of $\leq$, i.e., $\forall c_3 \in \mathcal{C}_\mathcal{K}$, $c_1 \leq c_3 \leq c_2$ implies either $c_1 = c_3$ or $c_2 = c_3$ (Ganter & Wille, 1999).

- All subsets of $\mathcal{L}_c$ have one greatest lower bound, the *join* operator, an one lowest upper bound, the *meet* operator.

Iceberg Galois lattice: When only frequent closed itemsets are considered with set inclusion, the resulting structure $(\hat{\mathcal{L}}, \subseteq)$ only preserves the *join* operator. This is called a join semi-lattice or upper semi-lattice. In the remaining of the paper, such structure is referred to as "*Iceberg Galois Lattice*" (Stumme *et al.*, 2002).
Therefore, given an Iceberg Galois lattice in which each closed itemset is "decorated" with its associated list of minimal generators, generic bases of association rules can be derived in a straightforward manner. Indeed, generic approximative[1] rules represent "inter-node" implications, assorted with the confidence measure, between two comparable equivalence relation classes[2], *i.e.*, from a sub-closed-itemset to a super-closed-itemset when starting from a given node in the partially ordered structure. For example, the generic approximative association rule C$\stackrel{\frac{1}{2}}{\Rightarrow}$ABE is generated from the two equivalence relation classes topped respectively by the closed itemsets 'C' and 'ABCE'. Whereas, generic exact[3] association rules are "intra-node" implications, with a confi-

[1]With confidence value less than 1.

[2]The closure operator ω induces an equivalence relation on items power set.

[3]With confidence value equal to 1.

dence value equal to 1, extracted from each node in the partially ordered structure (*e.g.*, from the closed itemset "ABCE", we obtain the following generic exact association rule: AB⇒CE).

Example 1
Let us consider the formal context $\mathcal{K}$ given by Figure 1 (Left). Figure 1 (Right) sketches equivalence classes of the induced equivalence relation from the extraction context $\mathcal{K}$. The associated Iceberg Galois lattice, for minsup = $\frac{2}{5}$, is depicted by Figure 1 (Bottom)[4]*. Each node in the Iceberg is represented as a couple (closed itemset, support) and is decorated with its associated minimal generator list. Figure 1 (Center) sketches equivalence classes of the induced equivalence relation from the extraction context $\mathcal{K}$.*

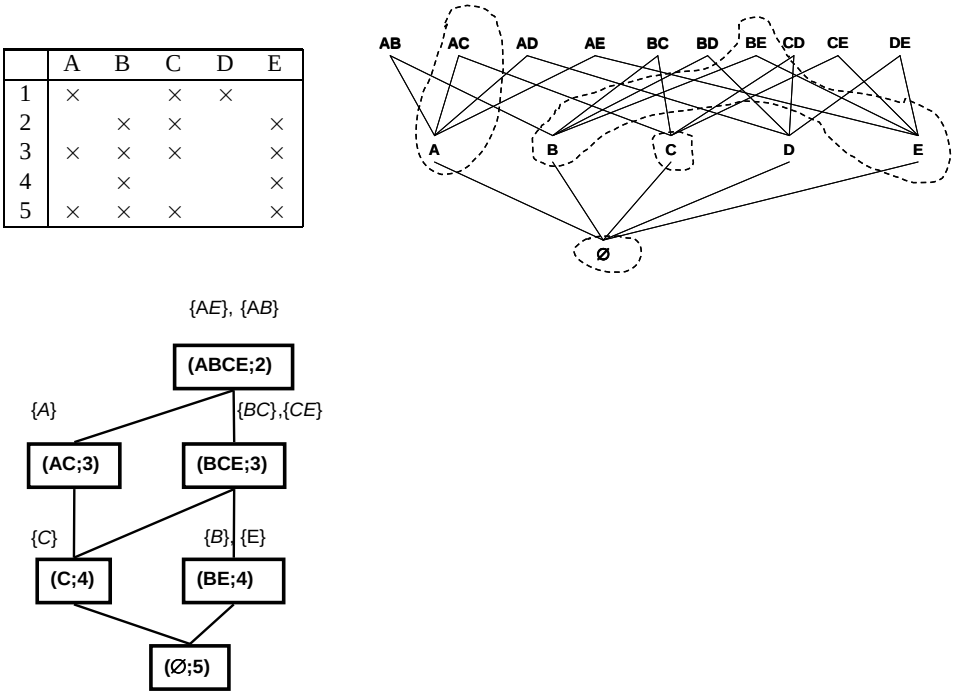


Figure 1: **Left:** Formal context $\mathcal{K}$ **Right:** Equivalence classes **Bottom:** Associated Iceberg Galois lattice for *minsup* =$\frac{2}{5}$.

3 Extraction of generic bases of association rules

Association rule derivation is achieved from a set F of frequent itemsets in an extraction context $\mathcal{K}$, for a minimal support *minsup*. An association rule R is a relation between

[4]We use a separator-free form for sets, e.g., *AB* stands for $\{A, B\}$.

itemsets of the form R: $X \Rightarrow (Y - X)$, in which X and Y are frequent itemsets, and $X \subset Y$. Itemsets X and $(Y - X)$ are called, respectively, *premise* and *conclusion* of the rule R. The valid association rules are those whose strength metric, confidence(R)= $\frac{support(Y)}{support(X)}$, is greater than or equal to the minimal threshold of confidence *minconf*. If confidence(R)=1 then R is called *exact association rule*, otherwise it is called *approximative association rule*.

The problem of relevance and usefulness of the association rules is highly important. This is due to the high number of association rules extracted from real-life databases and the presence of a high percentage of redundant rules conveying the same information. In the literature, we can witness the presence of techniques to prune such a set of rules, mainly based on statistical metrics. In what follows, we focus on the results issued from the FCA, to retrieve an information lossless reduced set of rules. Indeed, this reduced set, called *basis*, is composed of generic rules that have to fulfill the following requirements:

- "***Informativeness***": The generic basis of association rules allows to retrieve exactly the support and confidence of the derived (redundant) association rules.
- "***Derivability***": An inference mechanism should be provided (*e.g.*, an axiomatic system). The axiomatic system has to be valid (*i.e.*, should forbid derivation of non valid rules) and complete (*i.e.*, should enable derivation of **all** valid rules).

A critical review of the dedicated literature allowed mainly to split previous works into two pools:

1. **Non informative generic bases**: approaches described below are not informative, *i.e.*, support and confidence of redundant rules can not be determined exactly. In (Kryszkiewicz, 1998; Kryszkiewicz, 2002), Kryszkiewicz introduced a new syntactic derivation operator, called the "*Cover*". Based on the Cover operator, the author defined a minimal basis of rules called "*representative rules*" ($\mathcal{RR}$). $\mathcal{RR}$ basis was redefined in (Luong, 2001) under the name of *"representative basis*" ($\mathcal{RB}$). However, the premise and the conclusion parts of the generic rules of $\mathcal{RB}$ are not necessarily disjoint. To derive redundant rules from $\mathcal{RB}$, the author proposed a sound axiomatic system composed of *Left augmentation* and *Decomposition* Axioms.

   In (Zaki, 2004), Zaki defined a generic basis of association rules, called $\mathcal{NRR}$. The $\mathcal{NRR}$ basis is composed of rules having minimal premise and conclusion parts. To generate all redundant association rules, the author uses Augmentation and Transitivity axioms. However, association rules inferred by the application of the augmentation axiom are not always valid.

2. **Informative generic basis** Bastide *et al.* (Bastide *et al.*, 2000) characterized what they called "*Generic basis for exact association rules*" which is defined as follows:

   Definition 2
   Let $\mathcal{FCI}$ be the set of frequent closed itemsets extracted from the context and, for each frequent closed itemset I, let us denote $\mathcal{G}_I$ the set of minimal generators

of I.
$\mathcal{GBE}$ = $\{R : g \Rightarrow (I - g) \mid I \in \mathcal{FCI} \wedge g \in \mathcal{G}_I \wedge g \neq I\}$.

The authors also characterized what they called "*Informative basis for approximative association rules*" which is defined as follows:

Definition 3
Let $\mathcal{FCI}$ be the set of frequent closed itemsets extracted from the context and let us denote $\mathcal{G}$ the set of minimal generators.
$\mathcal{GBA}$ = $\{R : g \stackrel{c}{\Rightarrow}$ (Y-g), $Y \in \mathcal{FCI} \wedge g \in \mathcal{G} \wedge \omega(g) \subset Y \wedge c$ = confidence(R) $\geq$ *minconf}*.

As pointed out in (Kryszkiewicz, 2002), by using the Cover operator as axiomatic system, the couple ($\mathcal{GBE}$,$\mathcal{GBA}$) forms a subset of association rules which is information lossless. A couple of valid and complete axiomatic systems for $\mathcal{GBE}$ and $\mathcal{GBA}$, respectively, was given in (BenYahia & Nguifo, 2004b).

4 A new generic basis

As we have seen, $\mathcal{RR}$ and $\mathcal{NRR}$ generic bases are unfortunately not information lossless. The couple ($\mathcal{GBE}$, $\mathcal{GBA}$) is information lossless. However, as a drawback, the couple ($\mathcal{GBE}$, $\mathcal{GBA}$) is oversized specially for dense datasets. Hence, our contribution consists in introducing a new approach to extract an informative generic basis of association rules, which is by far more compact than ($\mathcal{GBE}$,$\mathcal{GBA}$).
Thus, we introduce the following definition of the new generic basis called $\mathcal{IGB}$:

Definition 4
Let $\mathcal{FCI}$ be the set of frequent closed itemsets and $\mathcal{G}_I$ the set of minimal generators of a frequent closed itemset I.
$\mathcal{IGB}$ = {R : $g_s \Rightarrow$ (I-g_s) | $I \in \mathcal{FCI} \wedge I \neq \emptyset \wedge g_s \in \mathcal{G}_{I'}$, $I' \in \mathcal{FCI} \wedge I' \subseteq I \wedge$ confidence(R) $\geq$ minconf $\wedge \nexists\, g' / g' \subset g_s \wedge$ confidence($g' \Rightarrow$ I-g')$\geq$ minconf }.

Proposition 5
The $\mathcal{IGB}$ generic basis is informative, i.e.*, the support and the confidence of all derived rules can be retrieved exactly from $\mathcal{IGB}$.*

Proof. Our approach consists in finding for each non empty frequent closed itemset I, the smallest minimal generator g_s of a frequent closed itemset I' subsumed by I and fulfilling the *minconf* constraint. Thus, generic association rules of $\mathcal{IGB}$ have the following form: $g_s \Rightarrow$ I-g_s. Therefore, we are able to reconstitute all frequent closed itemset by concatenation of the premise and the conclusion parts of a generic rule. Since the support of an itemset is equal to the support of the smallest closed itemset containing it, then the support and the confidence of all derived rules can be retrieved exactly. ■

	C$\overset{\frac{1}{2}}{\Rightarrow}$ABE ($\frac{2}{5}$)	A$\overset{\frac{2}{3}}{\Rightarrow}$BCE ($\frac{2}{5}$)	B$\overset{\frac{1}{2}}{\Rightarrow}$ACE ($\frac{2}{5}$)	
	E$\overset{\frac{1}{2}}{\Rightarrow}$ABC ($\frac{2}{5}$)	$\emptyset \overset{\frac{3}{5}}{\Rightarrow}$BCE ($\frac{3}{5}$)	$\emptyset \overset{\frac{3}{5}}{\Rightarrow}$AC ($\frac{3}{5}$)	
	$\emptyset \overset{\frac{4}{5}}{\Rightarrow}$BE ($\frac{4}{5}$)	$\emptyset \overset{\frac{4}{5}}{\Rightarrow}$C ($\frac{4}{5}$)		
E⇒B ($\frac{4}{5}$)	B⇒E ($\frac{4}{5}$)	C$\overset{\frac{3}{4}}{\Rightarrow}$A ($\frac{3}{5}$)	B$\overset{\frac{3}{4}}{\Rightarrow}$CE ($\frac{3}{5}$)	C$\overset{\frac{3}{4}}{\Rightarrow}$BE ($\frac{3}{5}$)
A⇒C ($\frac{3}{5}$)	BC⇒E ($\frac{3}{5}$)	E$\overset{\frac{1}{2}}{\Rightarrow}$ABC ($\frac{2}{5}$)	C$\overset{\frac{1}{2}}{\Rightarrow}$ABE ($\frac{2}{5}$)	B$\overset{\frac{1}{2}}{\Rightarrow}$ACE ($\frac{2}{5}$)
CE⇒B ($\frac{3}{5}$)	AB⇒CE ($\frac{2}{5}$)	A$\overset{\frac{2}{3}}{\Rightarrow}$BCE ($\frac{2}{5}$)	BC$\overset{\frac{2}{3}}{\Rightarrow}$AE ($\frac{2}{5}$)	E$\overset{\frac{3}{4}}{\Rightarrow}$BC ($\frac{3}{5}$)
AE⇒BC ($\frac{2}{5}$)		CE$\overset{\frac{2}{3}}{\Rightarrow}$AB ($\frac{2}{5}$)		

Table 1: (**Up**) $\mathcal{IGB}$ generic basis. **Bottom**) ($\mathcal{GBE}$, $\mathcal{GBA}$) generic bases for minsup=$\frac{2}{5}$ and minconf=$\frac{1}{2}$ (the support value is indicated between brackets.)

4.1 $\mathcal{IGB}$ generic basis construction

In what follows, we propose to present the $\mathcal{IGB}$ construction algorithm, whose pseudo-code is depicted by Algorithm 1. $\mathcal{IGB}$ construction algorithm takes as input the set of all FCI, $\mathcal{FCI}$, extracted by using one of the dedicated algorithms[5], *e.g.*, CLOSE, CHARM or CLOSET+ algorithms.

Proposition 6
Let I be a non empty frequent closed itemset, if support(I)≥ minconf, then the generic association rule R: $\emptyset \Rightarrow I \in \mathcal{IGB}$.

Proof. Proposition 6 derives straightforwardly from Definition 4. Since confidence (R:$\emptyset \Rightarrow$I)=support(I), then the generic rule $\emptyset \Rightarrow$I is valid. Hence, R presents the largest conclusion that can be drawn from the frequent closed itemset I, since there is no such another rule R':X'⇒Y' such that X' $\subset \emptyset$ and I $\subseteq$ Y'. ■

The $\mathcal{IGB}$ construction algorithm (see Algorithm 1)is based on Proposition 6. So, it considers the set of frequent closed itemsets $\mathcal{FCI}$. For each non empty closed itemset I, it checks whether its support is greater than or equal to *minconf*. If it is the case, then we generate the generic rule R:$\emptyset \Rightarrow$I. Otherwise, it has to look for the smallest minimal generator g_s, associated to a frequent closed itemset subsumed by I, and then generates the generic rule R:$g_s \Rightarrow$I-g_s if the *minconf* threshold is reached.

Example 7
Let us consider the extraction context given by Figure 1 (Left). Table 1 (Up) shows the $\mathcal{IGB}$ basis extracted from the formal context $\mathcal{K}$ for minsup =$\frac{2}{5}$ and minconf = $\frac{1}{2}$. Whereas Table 1(Bottom) shows the couple ($\mathcal{GBE}$,$\mathcal{GBA}$) for the same minsup and minconf settings.

[5]A critical survey of these algorithms can be found in (BenYahia & Nguifo, 2004a)

Algorithm 1: $\mathcal{IGB}$ construction

Input: $\mathcal{FCI}$: set of frequent closed itemsets and their associated minimal generators; minconf
Output: $\mathcal{IGB}$: Informative generic basis
begin
  foreach *non empty frequent closed itemset* $I \in \mathcal{FCI}$ **do**
    if *(support(I)* $\geq$ *minconf)* **then**
      R= $\emptyset \Rightarrow$ I
      R.support=support(I)
      R.confidence=support(I)
      $\mathcal{IGB}$=$\mathcal{IGB} \cup$ R
    else
      $L_{smallest-gen}$={}
      foreach $I' \subseteq I$ *in increasing order of size* $\mid \frac{support(I)}{support(I')} \geq$ *minconf* **do**
        foreach $g \in \mathcal{G}_{I'}$ *and* $g \neq I$ **do**
          if $\nexists\, g_s \in L_{smallest-gen} \mid g_s \subset g$ **then**
            $L_{smallest-gen}$=$L_{smallest-gen} \cup$ g
      foreach $g_s \in L_{smallest-gen}$ **do**
        R= $g_s \Rightarrow$I-g_s
        R.support=support(I)
        R.confidence=$\frac{support(I)}{support(g_s)}$
        $\mathcal{IGB}$=$\mathcal{IGB} \cup$ R
  return($\mathcal{IGB}$)
end

4.2 Generic association rule semantics

In the following, we have to discuss about the semantic attached to an association rule R:X$\stackrel{c}{\Rightarrow}$(Y-X). Usually, R expresses that the probability of finding Y with a value c depends on the presence of X. Thus, X constitutes a constraint for Y item correlations.
In the $\mathcal{IGB}$ basis, we can find generic association rules whose premise part can be empty. Such rules were considered in (Kryszkiewicz, 2002; Luong, 2001), but no attention was paid to a semantic interpretation attached to this type of knowledge.
Let us consider the extraction context given by Figure 1 (Left). For *minconf*=$\frac{2}{5}$ and applying Bastide *et al.*'s approach, we obtain among the possibly extracted generic association rules, C $\Rightarrow$ ABE, E $\Rightarrow$ ABC, B $\Rightarrow$ ACE, A $\Rightarrow$ BCE. However, does the probability of finding A, B, C and E together depend on the presence of A, B, C or E? Actually, the probability of finding A, B, C and E with a value greater than or equal to *minconf* = $\frac{2}{5}$ does not depend on any condition. Thus, we propose to represent such type of knowledge by only one generic association rule, *i.e.*, R: $\emptyset \Rightarrow$ ABCE. The generic basis $\mathcal{IGB}$ contains then two types of rules:(i) "*Implicative rules*" represented by generic association rules whose premise part is not empty. (ii) "*Factual rules*" represented by

generic association rules whose premise part is empty.

4.3 Redundant association rule derivation

In order to derive the set of all valid redundant association rules, we propose in what follows an axiomatic system and we prove that it is valid (*i.e.*, should forbid derivation of non valid rules) and that it is complete (*i.e.*, should enable derivation of all the valid rules).

Proposition 8
Let us consider the generic basis denoted by $\mathcal{IGB}$ and the set of all valid association rules extracted from $\mathcal{K}$, denoted by $\mathcal{AR}$. The following axiomatic system is valid.

A0. Conditional reflexivity: *If $X \overset{c}{\Rightarrow} Y \in \mathcal{IGB} \wedge X \neq \emptyset$ then $X \overset{c}{\Rightarrow} Y \in \mathcal{AR}$*

A1. Augmentation *If $X \overset{c}{\Rightarrow} Y \in \mathcal{IGB}$ then $X \cup Z \overset{c'}{\Rightarrow}$ Y-$\{Z\} \in \mathcal{AR}$,$Z \subset Y$.*

A2. Decomposition: *If $X \overset{c}{\Rightarrow} Y \in \mathcal{AR}$ then $X \overset{c}{\Rightarrow} Z \in \mathcal{AR}$, $Z \subset Y \wedge \omega(XZ) = XY$.*

Proof.

A0. Conditional reflexivity: follows from the proper definition of the $\mathcal{IGB}$.

A1. Augmentation Since R: X$\overset{c}{\Rightarrow}$Y $\in \mathcal{IGB}$ then confidence(R: X$\overset{c}{\Rightarrow}$Y)=c and support(R: X$\overset{c}{\Rightarrow}$Y) $\geq$ minsup. $\frac{support(XY)}{support(X)}$ = c $\geq$ minconf. Since X $\subset$ XZ, then support(X)$\geq$ support(XZ) and minconf $\leq \frac{support(XY)}{support(X)} \leq \frac{support(XY)}{support(XZ)}$. Thus, R': X $\cup$ Z$\overset{c'}{\Rightarrow}$Y-$\{Z\}$ is a valid association rule having a confidence value equal to c'= $\frac{support(XY)}{support(XZ)}$ and a support value equal to that of R.

A2. Decomposition: Since, R: X$\overset{c}{\Rightarrow}$Y $\in \mathcal{AR}$ then confidence(R: X$\overset{c}{\Rightarrow}$Y) = c $\geq$ minconf, and support(R: X$\overset{c}{\Rightarrow}$Y) = support(XY) $\geq$ minsup. c=$\frac{support(XY)}{support(X)}$ then support(XY)=c $\times$ support(X). Also, we have $\omega(XZ)$ = XY, then support(XZ) = support(XY) consequently, support(XZ) = c $\times$ support(X). Thus, R': X$\overset{c}{\Rightarrow}$Z is a valid association rule having support and confidence values equal to that of R .

Support and confidence values of the derived (redundant) rules are greater than or equal to those of the associated generic association rule. Thus, the proposed axiomatic system is valid ■

Remark 9
The conditional reflexivity is used by compliance with the constraint of non emptiness of the premise, in respect to an implicit "habit" stipulating that the premise part of an association rule is usually non empty.

Proposition 10
The proposed axiomatic system is complete: the set of all association rules extracted from $\mathcal{K}$ are derivable from $\mathcal{IGB}$ by using the proposed axiomatic system.

Proof. Let $\mathcal{IGB}$ be the generic basis extracted from the extraction context $\mathcal{K}$ for a given minsup and minconf. $\mathcal{AR}$ denotes the set of all association rules extracted from $\mathcal{K}$ and $\mathcal{FCI}$ the set of frequent closed itemsets.
Let R:X⇒Y-X ∈ $\mathcal{AR}$. In the following, we have to show that R can be derived from a generic association rule of $\mathcal{IGB}$ by the application of the proposed axiomatic system.

- If Y ∈ $\mathcal{FCI}$ then two cases are possible:
  1. there is no such a rule R':X'⇒Y-X' ∈ $\mathcal{AR}$ such that X'⊂X, then
     - if support(R)< minconf, then R:X⇒Y-X ∈ $\mathcal{IGB}$. R:X⇒Y-X ∈ $\mathcal{AR}$ by application of the conditional reflexivity axiom.
     - Else there is such a rule R":∅⇒Y∈ $\mathcal{IGB}$. By application of the augmentation axiom to R", we obtain the rule R:X⇒(Y-X).
  2. There is such a rule R":X"⇒Y-X"∈ $\mathcal{IGB}$ such that X"⊂X' ∧ X"⊂X. By application of the augmentation axiom to R":X"⇒Y-X", we obtain R:X⇒Y-X.
- Otherwise, there is such a rule R':X⇒Y'-X∈ $\mathcal{AR}$ such that Y'∈ $\mathcal{FCI}$ ∧ Y' = $\omega(Y)$. Then, it exists a rule R":X"⇒Y'-X"∈ $\mathcal{IGB}$ such that X"⊆X. We apply firstly, the augmentation axiom to R":X"⇒Y'-X"(if X"⊂X) in order to obtain R':X⇒Y'-X. Next, we apply the decomposition axiom to R' to find R:X⇒Y-X.

■

5 Experimental results

We carried out experimentations on benchmarking datasets, whose characteristics are summarized in Table 2, in order to evaluate the number of generic association rules. We implemented both algorithms in the C language under Linux Fedora Core 2. Physical characteristics of the machine are: a PC pentium 4 with a CPU clock rate of 3.06 Ghz and a main memory of 512 Mo. Table 3 reports the number of FCI extracted from the considered datasets. The number of reported generic association rules of $\mathcal{IGB}$ and the couple ($\mathcal{GBE}$, $\mathcal{GBA}$), for different values of *minconf*, are also given. We denote by FR the cardinality set of factual generic association rules and by IR the cardinality set of implicative generic association rules. To assess the compactness of the considered generic bases, the column labelled by $\mathcal{AR}$ reports the number of all valid association rules extracted by the Apriori algorithm (Agrawal *et al.*, 1996).

In the following, we focus on the variation of the reported generic rule number of the different generic bases versus the minconf value variation. Experimental results only stress on comparing $\mathcal{IGB}$ compactness versus that of the only informative generic basis pointed out by the state of the art review, *i.e.*, the couple ($\mathcal{GBE}$,$\mathcal{GBA}$). For the $\mathcal{IGB}$, when *minconf=minsup*, $\mathcal{IGB}$ contains only factual generic association rules. The number of factual generic association rules is equal to the number of FCI (*c.f.*, the third column of Table 3). This can be explained by the fact that all FCI supports are equal to or greater than *minconf*. As long as *minconf* value is increasing, the number

Base	Type	|Transaction|	|items|
T10I4D100K	Sparse	100000	1000
Mushrooms	Dense	8124	120
Connect	Dense	67557	130
Chess	Dense	3196	76

Table 2: Characteristics of datasets.

Dataset	minsup	$\#\mathcal{FCI}$	minconf	$\mathcal{AR}$	$\mathcal{IGB}$ (IR,FR)	($\mathcal{GBE}$,$\mathcal{GBA}$)
T10I4D100K	0.5%	1074	0.5%	2216	(0, 1074)	(0, 2216)
			1%	2216	(1210, 385)	(0, 2216)
			10%	2172	(1188, 0)	(0, 2172)
			50%	1145	(606, 0)	(0, 1145)
			100%	0	(0, 0)	(0, 0)
Mushrooms	30%	427	30%	94894	(0, 427)	(557, 7066)
			50%	79437	(922, 45)	(557, 5204)
			70%	58010	(954, 12)	(557, 3963)
			90%	24408	(794, 5)	(557, 1602)
			100%	8450	(557, 1)	(557, 0)
Connect	95%	809	95%	77816	(0, 809)	(682, 24654)
			96%	73869	(2092, 48)	(682, 23098)
			97%	60101	(2154, 284)	(682, 17788)
			98%	41138	(2463, 135)	(682, 11035)
			99%	19967	(2256, 28)	(682, 4568)
			100%	2260	(682, 0)	(682, 0)
Chess	87%	1194	87%	42740	(0, 1194)	(342, 31196)
			89%	40451	(1734, 689)	(342, 29362)
			91%	36098	(2293, 362)	(342, 25805)
			93%	29866	(2573, 193)	(342, 21008)
			95%	20312	(2681, 73)	(342, 14031)
			97%	10830	(1257, 29)	(342, 7353)
			100%	418	(342, 0)	(342, 0)

Table 3: Variation of generic association rules number vs minconf value variation (dense datasets are indicated in bold).

Dataset	minconf	$\frac{\mathcal{TGB}}{(\mathcal{GBE},\mathcal{GBA})}$	$\frac{\mathcal{TGB}}{\mathcal{AR}}$	$\frac{(\mathcal{GBE},\mathcal{GBA})}{\mathcal{AR}}$
T10I4D100K	0.5%	0.48	0.48	1
	1%	0.71	0.71	1
	10%	0.54	0.54	1
	50%	0.52	0.52	1
	100%	1	1	1
Mushrooms	30%	0.05	0.004	0.08
	50%	0.16	0.01	0.07
	70%	0.21	0.01	0.07
	90%	0.37	0.03	0.08
	100%	1	0.66	0.66
Connect	95%	0.03	0.01	0.32
	96%	0.1	0.03	0.32
	97%	0.13	0.04	0.3
	98%	0.22	0.06	0.28
	99%	0.43	0.11	0.26
	100%	1	0.3	0.3
Chess	87%	0.03	0.02	0.73
	89%	0.08	0.05	0.73
	91%	0.1	0.07	0.72
	93%	0.12	0.09	0.71
	95%	0.19	0.13	0.7
	97%	0.16	0.11	0.71
	100%	1	0.81	0.81

Table 4: Comparison of generic basis compactness.

of factual generic rules decreases until reaching 0 when *minconf*=1 (*c.f.*, Figure 2). Indeed, by varying *minconf*, each factual generic rule is substituted by a certain number of implicative generic rules. Thus, the higher this cardinality is, the more the number of $\mathcal{IGB}$ generic association rules increases.

A singularity for the MUSHROOM dataset is noteworthy. In fact, the number of factual generic rules is equal to 1 even for *minconf*=1 (usually, it is equal to 0). This can be explained by the fact that the item coded by '85' appears in all dataset transactions. Thus, for any value of *minconf*, the factual generic rule $\emptyset \rightarrow 85$ is always valid.

For the ***($\mathcal{GBE}$,$\mathcal{GBA}$)*** generic bases, we note that the number of $\mathcal{GBE}$ rules is insensitive to the variation of *minconf* value. However, the more *minconf* increases, the more the number of $\mathcal{GBA}$ rules decreases. Indeed, by increasing *minconf*, the number of minimal generators satisfying the *minconf* constraint decreases.

In what follows, we discuss the compactness degree of the considered generic bases. From the reported statistics in Table 4, we note that the gap between $\mathcal{IGB}$ and the set of all valid rules extracted using the APRIORI algorithm is more important for dense datasets. Indeed, the compactness degree of $\mathcal{IGB}$ ranges from 0.4% to 80%. However, for sparse datasets, the compactness degree is limited to a value varying between 48% and 100%. For sparse datasets, ($\mathcal{GBE}, \mathcal{GBA}$) contains as many generic rules as the set of all valid rules (*e.g.*, the two associated curves are merging for the T10I4D100K dataset). This can be explained by the fact that for sparse datasets, the set of frequent itemsets is equal to the set of FCI and to the set of minimal generators. However, for dense datasets, the compactness degree of ($\mathcal{GBE}, \mathcal{GBA}$) varies between 8% and 80%.

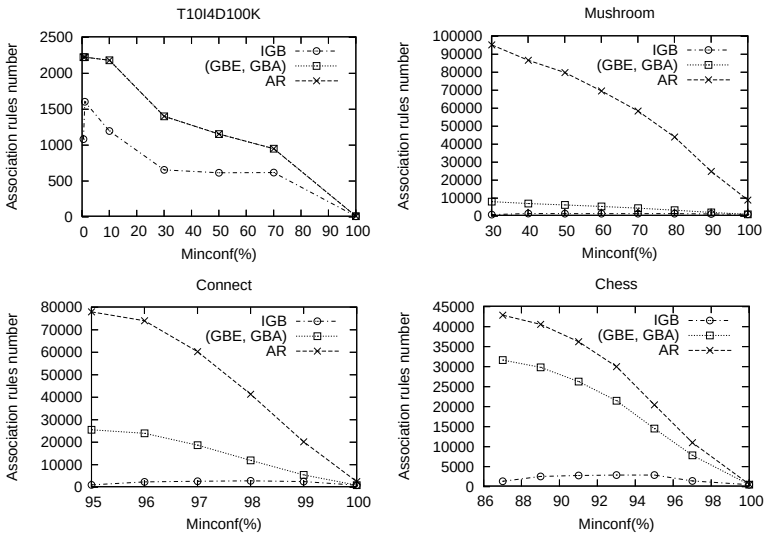


Figure 2: Generic association rule set cardinality variation versus minconf variation.

6 Conclusion

In this paper, we introduced an approach for the extraction, of an informative generic basis which is more compact than ($\mathcal{GBE}, \mathcal{GBA}$). We also provided a valid and complete axiomatic system, allowing to infer the set of all valid rules. We distinguished two types of generic rules; "*factual*" from "*implicative*" ones. We also implemented algorithms of $\mathcal{IGB}$ and ($\mathcal{GBE}, \mathcal{GBA}$) construction. Experimental results carried out on benchmarking datasets showed important profits in terms of compactness of the introduced generic basis. Of interest to mention that $\mathcal{IGB}$ extraction algorithm performances outperform those of ($\mathcal{GBE}, \mathcal{GBA}$) extraction algorithm.

In the near future, we plan to examine the potential benefits from integrating $\mathcal{IGB}$ basis in a query expansion system. In this context, we have to focus on assessing the trade-off between generic basis compactness and complexity of the associated axiomatic system, in case of association rule derivation. Also, we have to tackle the "pitfall" of representing factual rules in a generic basis visualization environment.

References

AGRAWAL R., MANNILA H., SRIKANT R., TOIVONEN H. & VERKAMO A. I. (1996). *Advances in Knowledge discovery and Data Mining*, chapter Fast discovery of association rules, p. 307–328. AAAI/MIT Press.

BARBUT M. & MONJARDET B. (1970). *Ordre et classification. Algèbre et Combinatoire*. Hachette, Tome II.

BASTIDE Y., PASQUIER N., TAOUIL R., LAKHAL L. & STUMME G. (2000). Mining minimal non-redundant association rules using frequent closed itemsets. In *Proceedings of the Intl. Conference DOOD'2000, LNCS, Springer-verlag*, p. 972–986.

BENYAHIA S. & NGUIFO E. M. (2004a). Approches d'extraction de règles d'association basées sur la correspondance de galois. *Ingénierie des Systèmes d'Information (ISI), Hermès-Lavoisier*, **3–4**(9), 23–55.

BENYAHIA S. & NGUIFO E. M. (2004b). Revisiting generic bases of association rules. In *Proceedings of 6th International Conference on Data Warehousing and Knowledge Discovery (DaWaK 2004),LNCS 3181, Springer-Verlag, Zaragoza, Spain*, p. 58–67.

GANTER B. & WILLE R. (1999). *Formal Concept Analysis*. Springer-Verlag.

KRYSZKIEWICZ M. (1998). Representative association rules. In *Research and Development in Knowledge Discovery and Data Mining. Proc. of Second Pacific-Asia Conference (PAKDD). Melbourne, Australia*, p. 198ñ209.

KRYSZKIEWICZ M. (2002). Concise representations of association rules. In D. J. HAND, N. ADAMS & R. BOLTON, Eds., *Proceedings of Pattern Detection and Discovery, ESF Exploratory Workshop, London, UK*, volume 2447 of *Lecture Notes in Computer Science*, p. 92–109: Springer.

LUONG V. P. (2001). Raisonnement sur les règles d'association. In *Proceedings 17ème Journées Bases de Données Avancées BDA'2001, Agadir (Maroc), Cépaduès Edition*, p. 299–310.

OHSAKI M., KITAGUCHI S., OKAMOTO K., YOKOI H. & YAMAGUCHI T. (2004). Evaluation of rule interestingness measures with a clinical dataset on hepatitis. In SPRINGER-VERLAG, Ed., *Proceedings of the Intl. Conference Actes PKDD, Pisa (Italy)*, p. 362–373.

PASQUIER N., BASTIDE Y., TAOUIL R. & LAKHAL L. (1999). Efficient Mining of Association Rules Using Closed Itemset Lattices. *Information Systems Journal*, **24**(1), 25–46.

STUMME G., TAOUIL R., BASTIDE Y., PASQUIER N. & LAKHAL L. (2002). Computing iceberg concept lattices with TITANIC. *J. on Knowledge and Data Engineering (KDE)*, **2**(42), 189–222.

ZAKI M. (2004). Mining Non-Redundant Association Rules. *Data Mining and Knowledge Discovery*, (9), 223–248.

Average Number of Frequent and Closed Patterns in Random Databases

Loïck LHOTE, François RIOULT, Arnaud SOULET

GREYC, CNRS - UMR 6072, Université de Caen
F-14032 Caen cedex France
{prenom.nom}@info.unicaen.fr

Résumé : Frequent and closed patterns are at the core of numerous Knowledge Discovery processes. Their mining is known to be difficult, because of the huge size of the search space, exponentially growing with the number of attributes. Unfortunately, most studies about pattern mining do not address the difficulty of the task, and provide their own algorithm. In this paper, we propose some new results about the *average* number of frequent patterns, by using probabilistic techniques and we extend these results to the number of closed patterns. In a first step, the probabilistic model is simple and far from the real life since the attributes and the objects are considered independent. Nevertheless according to this model, frequency threshold phenomena observed in practice are explained. We also prove that, for a fixed threshold, the number of frequent patterns is asymptotically exponential in the number of attributes and polynomial in the number of objects whereas, for a frequency threshold proportional to the number of objects, the number of frequent and closed patterns is asymptotically polynomial in the number of attributes without depending on the number of objects.

Mots-clés : data mining, average analysis, frequent and closed patterns

1 Introduction

In Knowledge Discovery in Databases, the goal is to find information in databases which describe the *objects* under study with their *attributes*. More precisely, we are trying to find interesting conjunctions of attributes, called *patterns*. These patterns are more or less present in the database, and are qualified by their *frequency* : it is the number of objects containing the pattern. When this quantity rises above a user-defined threshold, the pattern is said *frequent*.

Among others, frequent patterns are at the core of many data mining processes. They give a first piece of information, telling that some conjunctions of attributes are significantly present in the data. They are very useful, e.g. for the association rules discovery, which can ground classification methods. Frequent pattern mining has been well studied, because it is the first stage leading to association rules. Finding these patterns is algorithmically hard, while it is easy to derive association rules from them. In fact, the

search space is exponentially large with the number of attributes, and becomes rapidly intractable.

In this article, we are also interested in *closed* patterns. A closed pattern is the maximal pattern (w.r.t. the inclusion) of the set of patterns having the same frequency and sharing the same attributes. When they are associated with the corresponding pattern of objects containing the pattern of attributes and being also closed, both constitute a *concept*. Conceptual learning is a hot topic (Wille, 1992), and closed patterns is an easy way to non redundant association rules (Zaki, 2000). Their mining has then been widely examined (Kuznetsov & Obiedkov, 2002; Fu & Mephu Nguifo, 2004).

Unfortunately, most studies about pattern mining provide their own solution for solving the mining problem, and sometimes give the complexity of their algorithm, but the theoretical aspects of the difficulty of mining is rarely addressed. The exponential size of the search space is always recalled but only gives an upper bound on the number of frequent patterns, furthermore in the worst case.

In this article, we propose new results about the *average* number of frequent patterns, by using probabilistic techniques. We also give the average number of concepts (or closed patterns, see Section 2) for a frequency threshold proportional to the number of objects. We will see that these results confirm the intuition about the difficulty of the task, by showing that the number of patterns is exponentially large with the number of attributes, and polynomial with the number of objects. Besides, if the frequency threshold is a proportion of the number of objects (10% for example), the average number of frequent patterns is polynomial with the number of attributes, without depending on the number of objects.

The organization of this paper is as follows : we present in Section 2 some definitions and properties about pattern mining in databases, and give the main results of our work in Section 3. We change the model by adding more constraints in Section 4 and end the presentation with some open problems (Section 5). Section 6 is a short conclusion and Appendix A and B gather the proofs of the theorems.

2 Preliminaries

2.1 Notations

A database contains the objects under study, which are described by their attributes. It is usually a boolean matrix, where objects are drawn on the rows, and the binary attributes are the columns. We will not discuss here about the methods for obtaining such a boolean matrix, starting from continuous or multi-valued attributes (see (Srikant & Agrawal, 1996) for an example).
In this article, we will have to distinguish two frameworks :

1. the transactional (consumer bag) framework is the most classical : objects are called transactions and represent a list of purchase. Every bought product is an attribute, and is often called item. It is absent or present in the transaction ;
2. the attribute/value framework is related to the database domain : every continuous attribute is discretized and transformed into several new boolean attributes.

We will yet use the same notation for both frameworks considering that, at the end, we only use boolean attributes. We will come back again to the differences between both frameworks when specifying the probabilistic model (see Section 3.2). The set of attributes is denoted $\mathcal{A} = \{1..m\}$ and the set of objects is $\mathcal{O} = \{1..n\}$. A pattern is a subset of $\mathcal{A}$, and the collection of patterns is denoted by $2^{\mathcal{A}}$.

A database is a subset of $\mathcal{A} \times \mathcal{O}$ and can be represented by a $n \times m$ matrix $(\chi_{i,j})_{i=1..n,j=1..m}$. We can also consider that a database is a set of transactions ; then we will write that a pattern A is supported by a transaction T if $A \subset T$. The *support* of A is the set of all transactions containing A, and the frequency of A is the size of its support. A is said to be γ-frequent if its frequency is over a user-defined threshold γ :

Definition 1 (frequent pattern)
Let $\mathcal{B} = (\chi_{i,j})_{i=1..n,j=1..m}$ be a binary database with m items and n transactions and γ a strictly positive integer. A γ-frequent A is a pattern such that $|support(A)| \geq \gamma$.

During the demonstrations, we will use a matrix vision of the support : for all j in A and i in $support(A)$, $\chi_{i,j} = 1$, and for all i in $\mathcal{O} \backslash support(A)$, there exists j in A such that $\chi_{i,j} = 0$.
We now give, in this framework, the definition of a γ-closed pattern :

Definition 2 (frequent closed pattern)
Let $\mathcal{B} = (\chi_{i,j})_{i=1..n,j=1..m}$ be a binary database with m items and n transactions and γ a strictly positive integer. A pattern A is γ-closed if :
- *A is γ-frequent pattern,*
- *for all j in $\mathcal{A} \backslash A$, there exists i in $support(A)$ such that $\chi_{i,j} = 0$.*

2.2 Pattern mining

The first and most popular algorithm for mining frequent patterns is A-PRIORI (Agrawal & Srikant, 1994). The key idea is to use the anti-monotonous property of the frequency constraint, which entails that every subset of a frequent pattern is frequent as well, or reciprocally that a superset of an infrequent pattern is infrequent. Starting from frequent items, candidate patterns with two items are built and their frequency is checked in the database. When a candidate is not enough present, its supersets are pruned and will not ground any further candidate. New candidates are produced by joining two frequent patterns having the same prefix, and again checked in the database, etc.

With this method, patterns are mined with a level-wise strategy, computing them by increasing size. A-PRIORI requires only one database scan to check all candidates at each level : there will be as much database scans as the size of the largest frequent pattern. If we consider that the bottleneck of such a method lies in database accesses, the complexity of A-PRIORI regarding this criteria is good.

The concept of positive and negative border is very useful, in order to more precisely analyze the complexity. The positive border gathers the maximum frequent patterns, with respect to the inclusion order. The negative border offers a dual vision : it brings together the minimum infrequent patterns. Gunopulos et al. have shown that mining

the frequent patterns requires as many database accesses as there are elements in both borders (Gunopulos *et al.*, 1997a).

Since twenty years, closed pattern mining is well studied, but the known methods provide all closed patterns, while we are, in the context of data mining, only interested in the most frequent. Recent works combine both approaches, and the closed patterns can also be mined in a level-wise manner, by using the free (Calders & Goethals, 2003) or key patterns (Pasquier *et al.*, 1999), because they are the generators of the closed patterns.

2.3 Related work

We recall in this section some results about the complexity of frequent pattern mining. As we will see, we are aware about the difficulty of the mining task : Gunopulos et al. have shown that deciding whether there exists a frequent pattern with t attributes is NP-complete (Gunopulos *et al.*, 1997b; Purdom *et al.*, 2004). The associate counting problem is #P-hard. But we are not really aware about the number of frequent patterns. In fact, as far as we know, there does not exist such results in the literature.

The reason is that the search space is well known to be exponentially large with the number of attributes, and the worst case (e.g. a database where $\chi_{i,j} = 1$ for all i and j, see Figure 1-a) gives $2^m - 1$ frequent patterns (with the minimum frequency threshold $\gamma = 1$). In the middle matrix where $\chi_{i,i} = 0$ (see Figure 1-b), there are $2^m - 2$ closed patterns (with $\gamma = 1$). Finally, in the matrix of the Figure 1-c (Boros *et al.*, 2002), there are k maximal frequent patterns (k is such that $n = k\gamma$), $2^k - 2$ closed patterns, and more than $2^{k(l-1)}$ frequent patterns (l is such that $m = kl$). Of course, it is a pathological example, but we have here a situation where the number of closed patterns is exponentially larger than the number of maximal patterns, and the number of frequent patterns is again exponentially larger than the number of closed patterns.

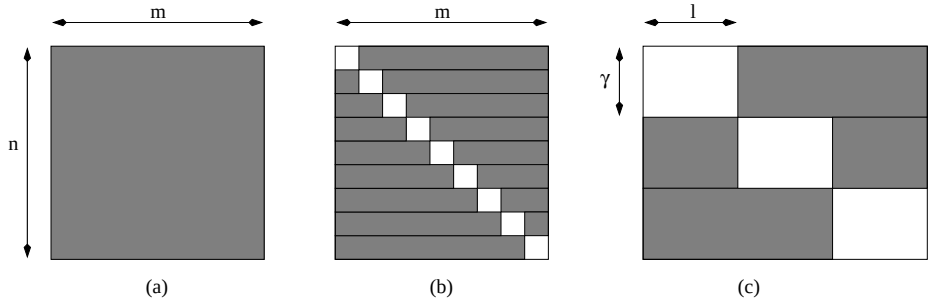


FIG. 1 – Three worst cases for pattern mining (when it is filled, it means that there are 1 in the matrix, otherwise there are 0).

Average analysis considerations might then provide interesting results. We found one such study (Purdom *et al.*, 2004), but it is related to the failure rate of A-PRIORI. It is useful for predicting the number of candidates that the algorithm will have to check.

This work confirms the results of (Geerts *et al.*, 2001), who used an upper bound. On other hand, in the seminal paper (Agrawal *et al.*, 1996), the authors of the A-PRIORI algorithm have explained that there are very few long patterns in a random database, and we will reuse the same probabilistic model.

We end this section with quoting (Dexters & Calders, 2004), which gives bounds on the size of the set of k-free patterns (Calders & Goethals, 2003). The authors provide a link between the number of free patterns and the maximum length of such a pattern. Even if this work is hard to relate to ours, we will have to investigate it further.

3 Results with the transactional framework

3.1 Hypothesis

In the following, we are interested in computing the average number of frequent and closed patterns, with respect to a certain minimum frequency threshold γ. All the provided results are asymptotic (i.e. for n and m large) so that the way the frequency threshold is growing with n is important. In practice, two cases are generally distinguished :

Hypothesis 1 (fixed case)
γ is fixed and small when compared to the number n of objects.

For example, γ can be fixed to ten transactions, when there are 100 000 transactions in the database.

Hypothesis 2 (proportional case)
γ is a ratio of n. In this case, we will say that there exists $r \in]0,1[$ such that $\gamma = rn$.

Since we do not have *infinite* databases, the percentage r must not be too small in practice. Nevertheless in our theoretical framework, r can be taken as small as we want but the speed of convergence of our asymptotics decelerates. The distinction between both hypothesis will be useful during the proof of our results : if γ is fixed and small, some approximation can be performed which could not be made if it is a percentage of n. When γ is a ratio of n, some threshold phenomena appears in an integral, which can be exploited by a Laplace's method.

Figure 2 shows the difference between a fixed γ and a variable one. The whole set of (closed) patterns is a lattice where all the patterns with the same cardinality are present on the same horizontal line. Besides, the most general patterns which have the lowest cardinality, are in the top of the lattice. Thus, the lattice may be represented by a rhombus and frequent (closed) patterns correspond to the grey superior part. This figure emphasizes the fact that a variable γ cuts the lattice with preserving the same proportion between both parts. With a fixed γ, this proportion is no longer preserved.

We also have to safely define the ratio between the number of objects and the number of attributes. We therefore require from n and m that they are polynomially linked, i.e. there exists a constant c such that $\log m \sim c \log n$. Let us note that this assumption is only useful for the asymptotic provided in the Theorem 1.

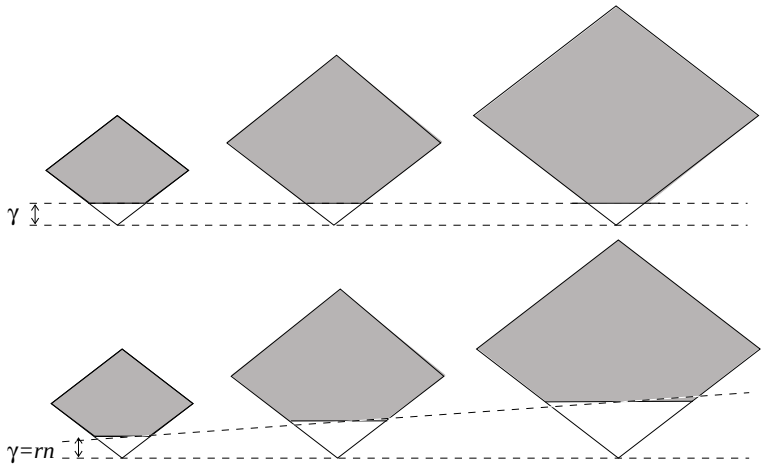


FIG. 2 – Difference between a fixed γ and a variable one (the empty set is at the top of the lattice, and the complete set of attributes is at the bottom. In the case where γ is fixed, the white parts of the lattices, which gather the infrequent patterns, seem to have an equal size, but it is false in practice)

3.2 Probabilistic model

We assume in this section that we are in the transactional framework. The probabilistic model we now describe is very simple. Since we can not appreciate *in advance* the correlations existing in real databases,we will suppose that :
The database $(\chi_{i,j})_{i=1..n,j=1..m}$ *forms an independent family of random variables which follows the same Bernoulli law of parameter* p *in* $]0,1[$.

Figure 3 provides an example of such a transactional database on the left chart : there is no constraint w.r.t. the columns on the number of 1 in each line. This model is far from the reality. Indeed, an equivalent in Information Theory is to modelize the French language with a memoryless source that respects the probability of each letter. The result is not very good but theoretical analysis can be yet lead. In the Section 4, this model is improved in order to handle items coming from continuous or multi-valued attributes. Nevertheless, we will again suppose that the objects are independent.

3.3 Results

This probabilistic model leads to a simple analysis of the average number of γ-closed and γ-frequent patterns. The next theorem sums up our first result for a fixed frequency threshold.

Theorem 1
If the positive integer γ *is fixed (hypothesis 1, Section 3.1) and if there exist a constant* c *such that* $\log m \sim c \log n$*, then for large* n *and* m*, the average number of* γ*-frequent*

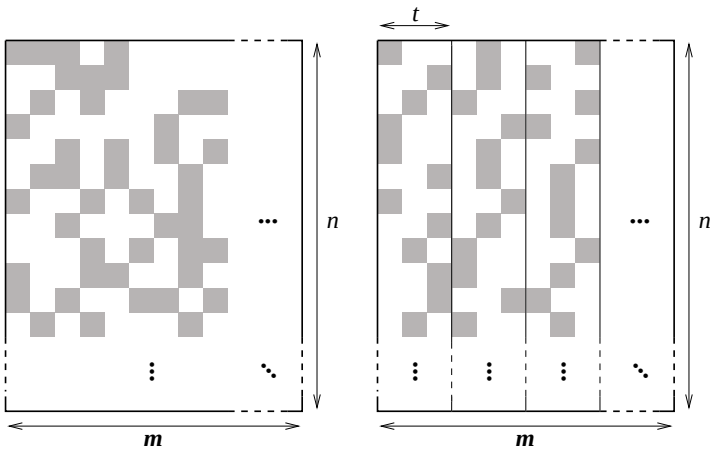


FIG. 3 – Transactional and attribute modelizations of databases (a grey square corresponding to a 1 in the matrix)

patterns $F_{m,n,\gamma}$ satisfies

$$F_{m,n,\gamma} \sim \binom{n}{\gamma}(1+p^\gamma)^m$$

This theorem states that the average number of γ-frequent patterns is asymptotically exponential in the number of attributes and polynomial in the number of objects. This is not really surprising, because we already had this intuition when we studied the search space (which is exponentially large with the number of attributes). Remark nonetheless that the average behavior is far from the worst case, which is 2^m. In addition, the denser the matrix is, the more frequent patterns there are : this is natural. Let us notice that the corresponding proof (see Appendix A) provides the exact asymptotic :

$$F_{m,n,\gamma} = \binom{n}{\gamma}(1+p^\gamma)^m \left[1 + O\left(n\left(\frac{1+p^{\gamma+1}}{1+p^\gamma}\right)^m\right)\right]$$

The following theorem gives a link between the average number of γ-closed patterns and the number of γ-frequent patterns :

Theorem 2
If γ satisfies $\gamma > \lfloor(1+\epsilon)\log m/|\log p|\rfloor$ for an ϵ strictly positive, then the average number of γ-frequent patterns and the average number of γ-closed patterns $C_{m,n,\gamma}$ are equivalent,

$$C_{m,n,\gamma} \sim F_{m,n,\gamma}.$$

We now detail the result of this theorem with the help of the database `T10I4D100K`, which has $n = 100000$ objects, $m = 1000$ attributes, and its density is $p = 0.01$. This dataset is generated by Srikant's synthetic data generator (Agrawal & Srikant, 1994),

and is available on the FIMI website[1]. We used this dataset to illustrate our aim since it has a large number of objects and attributes.

The threshold $\lfloor \log m / |\log p| \rfloor$ for γ involved in the theorem 2 is in practice very low w.r.t. the number of objects. For instance, the theorem applies on `T10I4D100K` when $\gamma > 1.5$. We mined the frequent and the closed patterns in this dataset with Uno's implementations for the FIMI (Uno & Satoh, 2003) and plotted on the Figure 4 the number of patterns, w.r.t. the threshold γ. When γ is greater than 20, we can see that the number of frequent patterns and the number of closed pattern are almost the same.

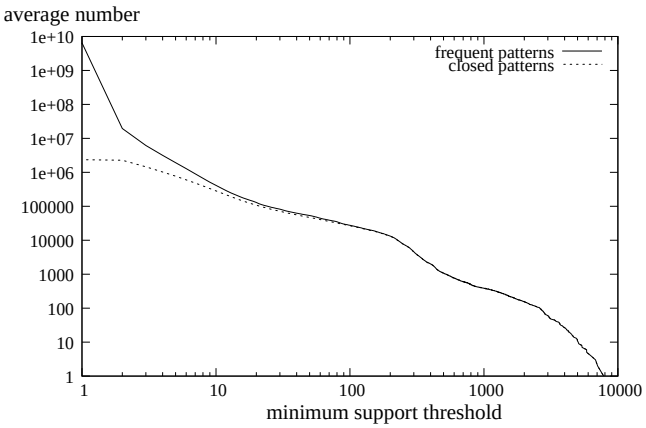


FIG. 4 – Average number of frequent/closed patterns on `T10I4D100K`

It is hard to give an intuition of this surprising result, because it is justified by an approximation which can be realized on the asymptotic (see demonstration on Appendix A). This phenomena can be explained by the poorness of our probabilistic model, which does not handle correlations. Closed patterns are normally useful, because they can summarize correlations. In the conditions of the theorem, almost all the frequent patterns are also closed and A-PRIORI has a better behavior than those algorithms based on the closed patterns.

Now, we consider the average number of patterns with the second hypothesis :

Theorem 3
If γ satisfies $\gamma = \lfloor rn \rfloor$ with $r \in]0,1[$ (hypothesis 2, Section 3.1), then the average number of γ-closed patterns and γ-frequent patterns satisfies for large m and n

$$C_{m,n,\gamma} \sim F_{m,n,\gamma} \sim \binom{m}{j_0}, \quad \textit{where} \quad j_0 = \left\lfloor \frac{\log r}{\log p} \right\rfloor .$$

In other words, j_0 is such that $p^{j_0+1} < r < p^{j_0}$.

[1]Frequent Itemset Mining Implementations is a workshop of the IEEE International Conference on Data Mining (ICDM) http ://fimi.cs.helsinki.fi/

This theorem is very important, because it states that the average number of frequent patterns (and closed patterns) is **polynomial with the number m of attributes** for a frequency threshold proportional to the number of objects. This is again surprising, because the search space is theoretically exponentially growing with m. Besides, this average number of frequent patterns does not depend on the number n of objects. In the future, we will reuse this result to justify applications of sampling techniques.

4 Results with the attribute/value framework

The preceding results show that our model can be improved. We now try to handle correlations in the data.

In practice, items often come from continuous attributes, that are split. For instance, the attribute *size of the patient* can be split into three items *small, medium, tall*. The previous modelization allowed a patient to be small and tall at the same time, while it is impossible. The new modelization considers these kinds of correlations. Nevertheless, we restrict ourselves to the case where all multi-valued or continuous attributes lead to the same number of boolean attributes $t > 1$. On the right chart, Figure 3 proposes an example of dataset where $t = 3$: there can be only one 1 in each triple of columns.

Since all the original attributes have the same size t, there exists one positive integer m_1 such that the number m of boolean attributes satisfies $m = m_1 t$. The new probabilistic model is based on the following hypothesis :

The database $(\Delta_{i,j} = (\chi_{i,tj+1}, \chi_{i,tj+2}, \ldots, \chi_{i,tj+t}))_{i=1..n, j=0..m_1-1}$ *forms an independent family of random variables with the same uniform law on the set composed with the sequences of size* t *with only one* one *and* $(t-1)$ *zeros (the density of the matrix is* $\frac{1}{t}$*).*

Once more, this model is far from the reality but its equivalent one in Information Theory would be to modelize the French language by a memoryless source that emits trigrams (if $t = 3$) according to their probability. The result is then better than our first modelization.

Using this model, our results are similar to the previous section. The proofs are also similar (see Appendix B).

Theorem 4
If the positive integer γ *is fixed, then the average number of* γ*-frequent patterns* $F_{m,n,\gamma}$ *satisfies for large* m *and* n

$$F_{m,n,\gamma} = \binom{n}{\gamma} \left(1 + \left(\frac{1}{t}\right)^{\gamma-1}\right)^{m_1} \left[1 + O\left(n \left(\frac{1 + (1/t)^{\gamma}}{1 + (1/t)^{\gamma-1}}\right)^{m_1}\right)\right]$$

Theorem 5
If γ *satisfies* $\gamma > \lfloor (1+\epsilon) \log m_1 / \log t \rfloor$ *for an* ϵ *strictly positive, then the average number of* γ*-frequent patterns and the average number of* γ*-closed patterns* $C_{m,n,\gamma}$ *are*

equivalent,

$$C_{m,n,\gamma} \sim F_{m,n,\gamma}.$$

Theorem 6
If γ satisfies $\gamma = \lfloor rn \rfloor$ with $r \in]0,1[$ which is not a power of p, then the average number of γ-closed patterns and γ-frequent patterns satisfies

$$C_{m,n,\gamma} \sim F_{m,n,\gamma} \sim \binom{m_1}{j_0} t^{j_0}, \quad \textit{where} \quad j_0 = \left\lfloor \frac{-\log r}{\log t} \right\rfloor.$$

Theorems 4, 5 and 6 show that the behavior of the asymptotics are very close to those proposed with the former modelization. Nevertheless, the number of γ-frequent patterns with the new model is exponentially lower for fixed γ. Indeed, the factor between the two modelizations is given by

$$\left(\frac{(1+(1/t)^{\gamma-1})^{1/t}}{1+(1/t)^{\gamma}} \right)^{m} = \delta^m \qquad \text{with } \delta < 1$$

It let us think that correlations entail an **exponential decay** on the number of frequent patterns (even if δ is near 1). Thus, this new model really refines the previous results.

Finally with $\gamma = \lfloor rn \rfloor$, the asymptotic is the same than in the first modelization and then, still polynomial.

5 Open problems

We now focus our intention on problems that we did not treat here or manage to solve :

1. What is the average number of γ-closed patterns for fixed γ ? Our feeling is that this number is asymptotically equivalent to the number of closed patterns of size around $\log n / |\log p|$ and frequency around $\log m / |\log p|$.
2. What is the average number of γ-closed/frequent patterns for other function γ such that $\gamma = \sqrt{n}$? The proof of Theorem 3 might be adapted to this context. In particular, the integer j_0 was chosen such $p^{j_0+1} < (\gamma-1)/(n-1) \approx r < p^{j_0}$. By extension, fixing $j_0 = \log_p(\gamma-1)/(n-1)$ we suppose that the number of frequent patterns is $\binom{m}{j_0}$.
3. What is the average size of the biggest frequent pattern ? It corresponds to the number of steps that A-PRIORI Algorithm performs.
4. The positive border is the set of γ-frequent patterns (or equivalently γ-closed ones) whose all supersets are infrequent. What is the average cardinal of the positive border ? This average is given by

$$\sum_{j=1}^{m} \binom{m}{j} \sum_{i=\gamma}^{n} \binom{n}{i} p^{ij} (1-p^j)^{n-i} \left(\sum_{u=0}^{\gamma-1} \binom{i}{u} p^u (1-p)^{i-u} \right)^{m-j}.$$

   For $r > p$, it tends to zero but for $r < p$, the term $(\sum)^{m-j}$ goes from 0 to 1 around $i = pn$. We did not manage to find the asymptotic.

5. The negative border is the set of patterns which are not γ-frequent, and whose all subsets are γ-frequent patterns. What is the average cardinal of the negative border ?

Of course, this list is not exhaustive.

6 Conclusion

In this paper, we gave the average number of frequent or closed patterns in a database, according to the frequency threshold, the number of attributes, objects, and the density of the database. We first used a simple model for the database, consisting in an independent family of Bernouilli random variables. We also provided the results with an improved modelization handling correlations in the attributes.

Our asymptotic results are useful in order to better understand the complexity of the frequent or closed pattern mining task. They explain the efficiency of the frequent pattern mining compared to the closed pattern one on databases close to our models. Furthermore, we emphasized the gap between two choices for the minimal frequency threshold (fixed or not) when the size of pattern lattice grows. In the first case, the average number of patterns is exponential with the number of attributes and polynomial with the number of objects. In the second case, it only polynomially depends on the number of attributes.

In further work, we want to take into account the correlations between objects in order to study the frequent and closed pattern mining on corresponding databases. Besides, we would like to propose a sampling method to estimate the number of patterns starting from a database and a minimum frequency threshold.

A Proofs with the transactional framework

We now prove the theorems. Let us recall that n and m are polynomially linked, i.e. there exist a constant c such that $\log m \sim c \log n$. Thus in the following, both parameters tend to infinity. The first lemma gives simple formulae directly deduced from the definitions for the average number of γ-frequent patterns and γ-concepts. This lemma is sufficient to show the result 2. The second lemma is an integral reformulation of a part of the previous formulae. The third lemma gives asymptotics for the integral part. Finally, we prove the theorem.

Lemma 1
The average number of γ-frequent patterns satisfies

$$F_{m,n,\gamma} = \sum_{j=1}^{m} \binom{m}{j} \sum_{i=\gamma}^{n} \binom{n}{i} p^{ij} (1-p^j)^{n-i}.$$

The average number of γ-concepts satisfies

$$C_{m,n,\gamma} = \sum_{j=1}^{m} \binom{m}{j} \sum_{i=\gamma}^{n} \binom{n}{i} p^{ij} (1-p^i)^{m-j} (1-p^j)^{n-i}.$$

Proof 1 (Lemma 1)
Fix (A, O) a γ-frequent pattern. The cardinal of a set E is noted $|E|$. Since O is the support of A, for all index in $A \times O$, there is a one in the matrix. But the probability o having a one is p so that the probability of having a one at each index of $A \times O$ is $p^{|A||O|}$. In addition, O is the greatest set containing all the items of A, so that for all transactions in $\mathcal{O}\backslash O$, there is at least one zero at an index of A. The probability of satisfying this last condition is $(1 - p^{|A|})^{n-|O|}$.

If (A, O) is a concept, the probability that A is the greatest set containing O is by symmetry $(1 - p^{|O|})^{m-|A|}$. Now, summing over all the possible cardinalities for A and O, we get both formulae.

Proof 2 (Theorem 2)
Both formulae are sufficient to prove Theorem 2. Indeed, if γ satisfies $\gamma > \lfloor (1 + \epsilon) \log m / |\log p| \rfloor$ for an ϵ strictly positive, then

$$\begin{aligned}(1-p^i)^{m-j} &\leq (1 - p^{(1+\epsilon)\log m/|\log p|-1})^{m-j} \\ &= (1 - \frac{1}{pm^{1+\epsilon}})^{m-j} \\ &\to 1.\end{aligned}$$

Theorem 2 follows from this equivalence.

The next lemma expresses the sum over i in $F_{m,n,\gamma}$ with an integral. This is the key point of all the proofs, since the way we approximate the integral leads to two different asymptotics for γ fixed or linear.

Lemma 2
One has the integral equality :

$$\sum_{i=\gamma}^{n} \binom{n}{i} x^i (1-x)^{n-i} = \gamma \binom{n}{\gamma} \int_0^x t^{\gamma-1}(1-t)^{n-\gamma} dt.$$

Proof 3 (Lemma 2)
Expanding $(1 - x)^{n-i}$ leads to

$$\sum_{i=\gamma}^{n} \binom{n}{i} x^i (1-x)^{n-i} = \sum_{i=\gamma}^{n} \binom{n}{i} \sum_{u=0}^{n-i} \binom{n-i}{u} (-1)^u x^{i+u}.$$

Now, the change of variable $v = u + i$ and the inversion of both signs sum gives the new equality

$$\sum_{i=\gamma}^{n} \binom{n}{i} x^i (1-x)^{n-i} = \sum_{v=\gamma}^{n} \binom{n}{v} x^v \sum_{i=\gamma}^{v} \binom{v}{i} (-1)^{v-i}.$$

A simple induction shows that the second sum simplifies into

$$(-1)^{v-\gamma} \binom{v-1}{\gamma-1}.$$

Hence, the previous inequality becomes

$$\sum_{i=\gamma}^{n}\binom{n}{i}x^i(1-x)^{n-i}=\sum_{v=\gamma}^{n}\binom{n}{v}x^v(-1)^{v-\gamma}\binom{v-1}{\gamma-1}.$$

Now, the binomials simplify, $\binom{n}{v}\binom{v-1}{\gamma-1}=\frac{\gamma}{v}\binom{n}{\gamma}\binom{n-\gamma}{v-\gamma}$ *and the change of variable* $w=v-\gamma$ *gives the new expression*

$$\sum_{i=\gamma}^{n}\binom{n}{i}x^i(1-x)^{n-i}=\gamma\binom{n}{\gamma}\sum_{w=0}^{n-\gamma}\binom{n-\gamma}{w}(-1)^w\frac{x^{w+\gamma}}{w+\gamma}.$$

To conclude, remark that the second sum is zero when $x=0$ *and that the derivative according to* x *is exactly* $x^{\gamma-1}(1-x)^{n-\gamma}$*. The lemma follows.*

We can now prove Theorem 1.

Proof 4 (Theorem 1)
Let f *be the function* $f(x)=(1-x)^{n-\gamma}$*. The sign of the derivatives of* f *alternates so that, the Taylor expansion of* f *entails the bounds,*

$$\sum_{l=0}^{2k+1}\frac{f^{(l)}(0)}{l!}x^l\leq f(x)\leq\sum_{l=0}^{2k}\frac{f^{(l)}(0)}{l!}x^l$$

for all positive integer k*. Now the derivatives satisfy* $f^{(l)}(0)=(-1)^l(n-\gamma)\dots(n-\gamma-l+1)x^{n-\gamma-l}$*. A bound of the integral formula is then*

$$\begin{aligned}\int_0^x t^{\gamma-1}(1-t)^{n-\gamma}dt &\approx \int_0^x\sum_{l=0}^{2k+1}\frac{f^{(l)}(0)}{l!}t^{l+\gamma-1}dt\\ &= \sum_{l=0}^{2k+1}\binom{n-\gamma}{l}(-1)^l\frac{x^{l+\gamma}}{l+\gamma}.\end{aligned}$$

Applying Lemma 2 *with* $F_{m,n,\gamma}$*, using the previous bounds and summing over* j *in* $F_{m,n,\gamma}$ *finally gives*

$$F_{m,n,\gamma}\approx\gamma\binom{n}{\gamma}\sum_{l=0}^{2k+1}\binom{n-\gamma}{l}(-1)^l\frac{(1+p^{l+\gamma})^m-1}{l+\gamma}.$$

In particular for $k=0$*, one has*

$$\binom{n}{\gamma}((1+p^\gamma)^m-1)-\gamma\binom{n}{\gamma}\binom{n-\gamma}{1}\frac{(1+p^{1+\gamma})^m-1}{1+\gamma}\leq F_{m,n,\gamma}\leq\binom{n}{\gamma}(1+p^\gamma)^m-1.$$

To conclude, the condition $\log m\sim c\log n$ *entails that the binomials are polynomial in* m *and it is negligible compared to the exponential part. This finishes the proof of Theorem* 1.

The last lemma describes the asymptotic of the integral when n is large.

Lemma 3
Suppose that γ satisfies $\gamma = \lfloor rn \rfloor$ with r a non-power of p.
For $x > r$,

$$\int_0^x t^{\gamma-1}(1-t)^{n-\gamma}dt = \frac{1}{\gamma\binom{n}{\gamma}}(1+\epsilon_n(x)),$$

with $(\epsilon_n)_n$ a sequence of decreasing functions that converges uniformly to zero.
For $x < r$,

$$\int_0^x t^{\gamma-1}(1-t)^{n-\gamma}dt = \frac{\exp(ng_n(x))}{ng_n'(x)}(1+\widetilde{\epsilon}_n(x)),$$

with $(\widetilde{\epsilon}_n)_n$ a sequence of increasing functions that converges uniformly to zero and

$$g_n(x) = \frac{\gamma-1}{n}\log x + \frac{n-\gamma}{n}\log(1-x).$$

Proof 5 (Lemma 3)
This lemma is the well known Laplace Method. The proof is then let to the reader.

We finally prove Theorem 3.

Proof 6 (Theorem 3)
Integer j_0 is (asymptotically) the lowest integer j such that $p^j > r$. By Lemma 1 and Lemma 2, the average number of frequent patterns satisfies

$$F_{m,n,\gamma} = \sum_{j=1}^{n}\binom{m}{j}\gamma\binom{n}{\gamma}\int_0^{p^j} t^{\gamma-1}(1-t)^{n-\gamma}dt.$$

The sum is then split into two sums $\sum_{j=1}^{j_0} + \sum_{j=j_0+1}^{m}$ and the use of lemma 3 provides the equivalence

$$F_{m,n,\gamma} \sim \sum_{j=1}^{j_0}\binom{m}{j} + \sum_{j=j_0+1}^{m}\binom{m}{j}\gamma\binom{n}{\gamma}\frac{\exp(ng_n(p^j))}{ng_n'(p^j)}.$$

The first sum is equivalent to $\binom{m}{j_0}$ since j_0 is constant. A simple upper bound gives the inequality for the second sum,

$$\sum_{j=j_0+1}^{m}\binom{m}{j}\frac{\exp(ng_n(p^j))}{ng_n'(p^j)} \leq \frac{1}{\gamma-1-p^{j_0}(n-1)}\sum_{j=j_0+1}^{m}\binom{m}{j}p^{\gamma j}(1-p^j)^{n-\gamma+1}.$$

Now, an equivalent of the right sum is

$$\sum_{j=j_0+1}^{m}\binom{m}{j}p^{\gamma j}(1-p^j)^{n-\gamma+1} \sim \binom{m}{j_0+1}p^{\gamma(j_0+1)}(1-p^{j_0+1})^{n-\gamma+1}. \quad (1)$$

Indeed, let $w_j = \binom{m}{j} p^{\gamma j}(1-p^j)^{n-\gamma+1}$. *The ratio* w_{j+1}/w_j *is decreasing with* j *and the ratio* w_{j_0+2}/w_{j_0+1} *satisfies*

$$\frac{w_{j_0+2}}{w_{j_0+1}} = \frac{m-j_0-2}{j_0+3}\exp(n\theta(\gamma,n,p,j_0))$$

$$\textit{with} \quad \theta(\gamma,n,p,j_0) = \frac{\gamma}{n}\log p + \frac{n-\gamma+1}{n}\log(1+p^{j_0+1}\frac{1-p}{1-p^{j_0+1}}).$$

Using that γ/n *tends to* r *as* n *tends to infinity and that* $p^{j_0+1} < r$, *the function* θ *is shown to converge to a strictly negative constant. Hence, the ratio* w_{j_0+2}/w_{j_0+1} *tends to zero as* n *tends to infinity what is sufficient to prove the equivalent of Formula (1). The Stirling formula applied with the binomial* $\binom{n}{\gamma}$ *entails the equivalent*

$$\binom{n}{\gamma} p^{(j_0+1)\gamma}(1-p^{j_0+1})^{n-\gamma} \sim \sqrt{\frac{1}{2\pi r(1-r)n}}\left(\frac{p^{j_0+1}n}{\gamma}\right)^{\gamma}\left(\frac{1-p^{j_0+1}}{1-(\gamma/n)}\right)^{n-\gamma}$$

$$= \exp(n\widetilde{\theta}(\gamma,n,p,j_0)),$$

$$\textit{where} \qquad \widetilde{\theta}(\gamma,n,p,j_0) = \frac{\gamma}{n}\log\frac{p^{j_0+1}n}{\gamma} + (n-\gamma)\log\frac{1-p^{j_0+1}}{1-(\gamma/n)}.$$

Finally, since for all positive x, $\log x \leq x-1$ *and* $\log 1+x \leq x$, *the function* $\widetilde{\theta}$ *is proved to converge to a strictly negative number. It follows that*

$$\gamma\binom{n}{\gamma}\sum_{j=j_0+1}^{m}\binom{m}{j}\frac{\exp(ng_n(p^j))}{ng_n'(p^j)} \to 0.$$

which finishes the proof of Theorem 3.

B Proofs with the attribute/value framework

The proofs are exactly identical. The only change is the first formula for the average number of frequent patterns or concepts.

Lemma 4
The average number of γ*-frequent patterns satisfies*

$$F_{m,n,\gamma} = \sum_{j=1}^{m_1} t^j \binom{m_1}{j}\sum_{i=\gamma}^{n}\binom{n}{i}\left(\frac{1}{t}\right)^{ij}(1-\left(\frac{1}{t}\right)^j)^{n-i}.$$

The average number of γ*-closed patterns satisfies*

$$C_{m,n,\gamma} = \sum_{j=1}^{m_1} \binom{m_1}{j}\sum_{i=\gamma}^{n}\binom{n}{i}\left(\frac{1}{t}\right)^{ij}(1-\left(\frac{1}{t}\right)^i)^{m_1-j}(1-\left(\frac{1}{t}\right)^j)^{n-i}.$$

All the previous proofs extend to these formulae.

Références

AGRAWAL R., MANNILA H., SRIKANT R., TOIVONEN H. & VERKAMO A. (1996). Fast discovery of association rules. In *Advances in Knowledge Discovery and Data Mining*.

AGRAWAL R. & SRIKANT R. (1994). Fast algorithms for mining association rules. In *Intl. Conference on Very Large Data Bases (VLDB'94), Santiago de Chile*.

BOROS E., GURVICH V., KHACHIYAN L. & MAKINO K. (2002). On the complexity of generating maximal frequent and minimal infrequent sets. In *Symposium on Theoretical Aspects of Computer Science*, p. 133–141.

CALDERS T. & GOETHALS B. (2003). Minimal k-free representations of frequent sets. In *Proceedings of PKDD'03*.

DEXTERS N. & CALDERS T. (2004). Theoretical bounds on the size of condensed representations. In *ECML-PKDD 2004 Workshop on Knowledge Discovery in Inductive Databases (KDID)*.

FU H. & MEPHU NGUIFO E. (2004). Etude et conception d'algorithmes de génération de concepts formels. *Revue des sciences et technologies de l'information, série ingénierie des systèmes d'information (RSTI-ISI)*, **9**, 109–132.

GEERTS F., GOETHALS B. & VAN DEN BUSSCHE J. (2001). A tight upper bound on the number of candidate patterns. In *Proceedings of ICDM'01*, p. 155–162.

GUNOPULOS D., MANNILA H., KHARDON R. & TOIVONEN H. (1997a). Data mining, hypergraph transversals, and machine learning. In *PODS 1997*, p. 209–216.

GUNOPULOS D., MANNILA H. & SALUJA S. (1997b). Discovering all most specific sentences by randomized algorithms. In *ICDT*, p. 215–229.

KUZNETSOV S. O. & OBIEDKOV S. A. (2002). Comparing performance of algorithms for generating concept lattices. *J. Exp. Theor. Artif. Intell.*, **14(2-3)**, 189–216.

PASQUIER N., BASTIDE Y., TAOUIL R. & LAKHAL L. (1999). Efficient mining of association rules using closed itemset lattices. *Information Systems*, **24**(1), 25–46.

PURDOM P. W., VAN GUCHT D. & GROTH D. P. (2004). Average-case performance of the apriori algorithm. *SIAM Journal on Computing*, **33**(5), 1223–1260.

SRIKANT R. & AGRAWAL R. (1996). Mining quantitative association rules in large relational tables. In *Proceedings of the 1996 ACM SIGMOD international conference on Management of data*, p. 1–12 : ACM Press.

UNO T. & SATOH K. (2003). LCM : An efficient algorithm for enumerating frequent closed item sets. In *Workshop on Frequent Itemset Mining Implementations (ICDM'03)*.

WILLE R. (1992). Concept lattices and conceptual knowledge systems. In *Computer mathematic applied, 23(6-9) :493-515*.

ZAKI M. J. (2000). Generating non-redundant association rules. In *SIGKDD'00, Boston*, p. 34–43.

Fouille de données biomédicales : apports des arbres de décision et des règles d'association à l'étude du syndrome métabolique dans la cohorte STANISLAS

Sandy Maumus[1-2], Amedeo Napoli[2], Laszlo Szathmary[2], Sophie Visvikis-Siest[1]

[1] INSERM U525, 54000 Nancy
{sandy.maumus, sophie.visvikis-siest}@nancy.inserm.fr

[2] LORIA, 54506 Vandoeuvre-Lès-Nancy
{maumus, napoli, szathmar}@loria.fr

Résumé : Nous présentons deux études de fouille de données, l'une s'appuyant sur les arbres de décision et l'autre sur les motifs fréquents et les règles d'association. Les résultats obtenus sont encourageants car nous réussissons à extraire des connaissances utiles et nouvelles pour l'expert. De plus, ces travaux nous ont conduit à proposer les premiers éléments d'une méthodologie globale de fouille de règles en bioinformatique. Au cœur de ces réflexions, nous soulignons le rôle majeur de l'expert dans le processus de fouille de données.

Mots-clés : Arbre de décision, Motif fréquent, Règle d'association, Syndrome métabolique.

Dans l'équipe 4 de l'unité INSERM 525, nous disposons d'une base de données contribuant à la compréhension des mécanismes entraînant l'athérosclérose : la cohorte STANISLAS (Siest *et al.*, 1998). C'est une étude familiale sur dix ans dont l'objectif principal est d'étudier le rôle et la contribution de facteurs génétiques et environnementaux sur la fonction cardiovasculaire. Des familles de la Meurthe-et-Moselle et des Vosges ont été invitées à venir passer un examen de santé tous les cinq ans. Lors du recrutement initial (1993-1995, t_0), les critères d'inclusion étaient les suivants : familles supposées saines, exemptes de maladies aiguës et/ou chroniques, composées de deux parents et de deux enfants de plus de six ans. 1006 familles (4295 sujets) ont ainsi pu être recrutées. Les données recueillies peuvent se diviser en trois catégories : (1) cliniques et environnementales, (2) biologiques et (3) génétiques.

Un des thèmes auxquels nous nous intéressons plus particulièrement est le syndrome métabolique (SM), une affection regroupant des facteurs de risque cardiovasculaire. Notre objectif est d'expérimenter sur les données de la cohorte différentes techniques de fouille afin d'étudier les mécanismes impliqués dans le SM.

Pour nos expérimentations, nous avons utilisé d'une part l'algorithme C4.5 proposé dans le système Weka (Witten & Frank, 2002) qui permet la construction d'arbres de décision. D'autre part, nous avons testé CORON, une plate-forme développée dans l'équipe Orpailleur qui extrait les motifs fréquents (Szathmary &

Napoli, 2005). Un autre module intégré dans CORON, ASSRULEX (Association Rule eXtractor), permet la génération de règles d'association à la fois à partir des motifs fréquents et des motifs fermés fréquents.

Les expérimentations avec C4.5 ont été menées sur deux sous-populations de la cohorte STANISLAS. Nous avons créé une nouvelle variable nominale, « SM », qui décrit pour chaque individu s'il est atteint ou non par le syndrome métabolique. Pour cela, nous avons utilisé une définition standard où un individu est atteint par le SM s'il a au moins trois des cinq critères suivants : hyperglycémie, hypertriglycéridémie, HDL-cholestérol bas ("bon" cholestérol), hypertension, obésité. Ainsi, SM est la classe cible sur laquelle nous cherchons à classifier les individus. Le rôle de l'expert est ici très important pour le choix des paramètres et l'interprétation de résultats.

Pour l'extraction de motifs et de règles, l'implication de l'expert dans le processus de fouille de données est également très importante et s'étend du pré-traitement au post-traitement des données. Nous proposons une méthodologie globale pour la fouille de règles en cinq étapes : (1) Etapes de pré-traitement, (2) Utilisation d'un logiciel de fouille, (3) Etapes de post-traitement : fouille de règles, (4) Visualisation des résultats, (5) Interprétation et validation des résultats. Ce schéma global a été appliqué à des données discrètes de la cohorte STANISLAS. La conclusion finale de cette étude après validation statistique a apporté une connaissance nouvelle à l'expert ayant une réelle valeur en biologie.

En conclusion, nos expérimentations soulignent le rôle primordial de l'expert dans le processus de fouille de données. Les arbres de décision permettent de déterminer les paramètres qui discriminent au mieux les individus selon la classe SM. Quant aux motifs et aux règles, en suivant une méthodologie globale, ils permettent de mettre en évidence des profils et de générer de nouvelles hypothèses en biologie qui peuvent ensuite être validées, par les statistiques par exemple. Alors que ces deux méthodes de fouille ont ici été évaluées séparément, pour la suite de notre travail, nous souhaitons réaliser leur combinaison et étudier l'intérêt des résultats qui seront générés.

Références

SIEST G., VISVIKIS S., HERBETH B., GUEGUEN R., VINCENT-VIRY, M., SASS C. , BEAUD B., LECOMTE E., STEINMETZ J., LOCUTY J. & CHEVRIER P. (1998). Objectives, design and recruitment of a familial and longitudinal cohort for studying gene-environment interactions in the field of cardiovascular risk: the Stanislas cohort. *Clin. Chem. and Lab. Med.* 36, p. 35-42.

SZATHMARY L. & NAPOLI A. (2005). CORON: A Framework for Levelwise Itemset Mining Algorithms. In *Supplementary Proceedings of the Third International Conference on Formal Concept Analysis (ICFCA '05)*, p. 110-113, Lens, France.

WITTEN I.H. & FRANK E. (2000). Data Mining: Practical machine learning tools with Java implementations. Morgan Kaufmann, San Francisco.

Index des auteurs