Probabilistic rule induction for explainable CBR under uncertainty

AI-2021 41st SGAI International Conference on AI

Martin Jedwabny, Pierre Bisquert and Madalina Croitoru

14-16 December 2021

LIRMM, Inria GraphIK, Univ Montpellier, CNRS, Montpellier, France

Outline

Introduction

Introduction

- **Context**: implementation of ethical automated agents [Tolmeijer et al., 2020].
- **Problem**: ethical values are often hard to elicit as people have different views on morality and the opinion of domain experts (philosophers in our case) is expensive to elicit.
- **Example** (taking control of a autonomous vehicle):
  An AI-equipped vehicle can spontaneously take control from the human driver under dangerous circumstances. Each situation revolves around either taking control of the vehicle, or doing nothing.

Introduction

- **Question**: how can we find this answer in different situations? Should we ask experts? Should we survey people? How can we cover unseen scenarios?
- **Idea**: imagine we have a knowledge base with plenty of cases, we could get the most similar/relevant ones and look at their answers to produce a decision for new cases. E.g:
  1. The driver is speeding past the legal limit to take a passenger to a hospital.
  2. The driver has been going in and out of his/her lane with no objects discernible ahead.
  3. etc...
- In other words, we can apply **case-based reasoning**.

Introduction

- **How**: developing a framework with which we:
  1. Crowd-source answers to different ethically-nuanced scenarios related to a specific problem at hand,
  2. Retrieve the most similar past cases to this problem, and
  3. Generalize these past cases on ethical terms.

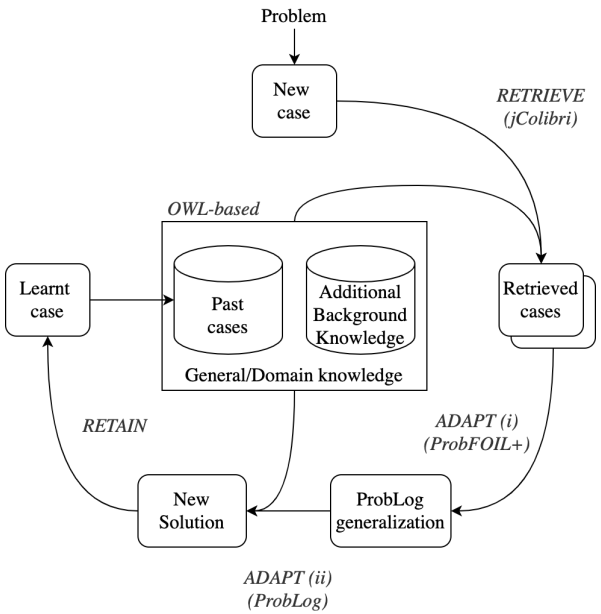

Figure 1: Overall architecture

Case-based reasoning

Case-based reasoning

- Case-based reasoning (CBR) is a methodology which consists of finding and reusing past problems to solve new ones.
- A case base is a collection $\mathcal{CB} = (x_i, y_i)_i$ of (possibly non-unique) problem/solution pairs.
- Given a case base and a novel problem x_{new}, case-based reasoners suppose that $\mathcal{CB}$ follows an unknown relation $Sols \subseteq \mathcal{P} \times \mathcal{S}$ and its objective is to replicate its behaviour such that $(x_{new}, y_{new}) \in Sols$.

Case-based reasoning

- The CBR process typically consists of [Aamodt and Plaza, 1994]:
  1. *Retrieve* the k cases that maximize a similarity function $sim : \mathcal{P} \times \mathcal{P} \mapsto [0, 1]$ between cases and x^{new}.
  2. *Adapt* the retrieved cases from the previous step by integrating their solutions into a solution for x^{new}.
  3. *Retain* the new solution after validation into $\mathcal{CB}$ if necessary.

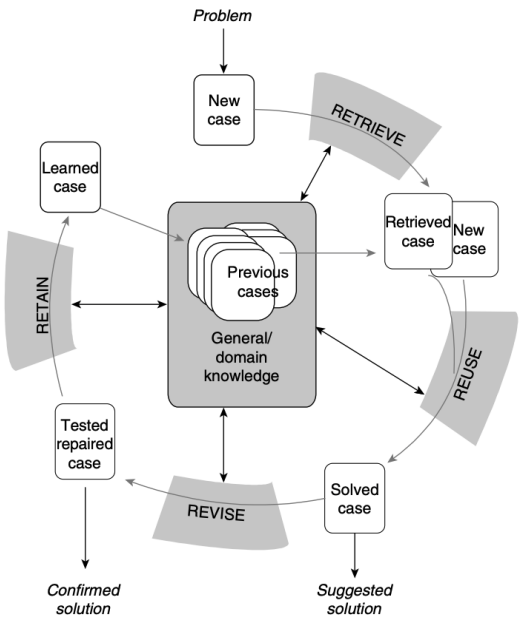

Figure 2: Case-based reasoning cycle.

Crowd-sourcing ethical preferences

Crowd-sourcing ethical preferences

- **Situation**: in ethical dilemmas, no choice is perfectly ethical, still, AI systems have to make choices, for example, by using preferences.
- **Problem**: preferences are largely subjective in ethics, people contradict each other and sometimes people make inconsistent choices in very similar situations.
- **Idea**: infer ethical preferences from surveys given to a large number of people and aggregate their decisions.

Crowd-sourcing ethical preferences

Example (autonomous vehicle continued): We can model the case base such that:

- A **case** x_i is represented by a set of predicates of the form $Duty(Situation, Option, Value)$ where $Duty \in \{preventCollision, respectAutonomy, withinLimit, preventHarm\}$, $Option \in \{takeControl, doNothing\}$, $Value \in \{true, false\}$.
- A **solution** $y_i \in \{takeControl, doNothing\}$.

Crowd-sourcing ethical preferences

- **Sample duty**: $preventCollision(s, takeControl, true)$ means taking control of the car in situation 's' guarantees preventing a collision.
- **Sample case**: driving alone, there is a bale of hay ahead in the driver's lane. There is a vehicle close behind that will run the driver's vehicle upon sudden braking and he/she can't change lanes, all of which can be determined by the system. The driver starts to brake.

$$\begin{aligned}\Sigma_s = \{&preventCollision(s, takeControl, false), respectAutonomy(s, takeControl, false),\\ &preventHarm(s, takeControl, true), preventCollision(s, doNothing, false),\\ &respectAutonomy(s, doNothing, true), preventHarm(s, doNothing, false)\}\end{aligned}$$

- **In practice**: users receive a detailed description of several cases and decide whether to *takeControl*, or *doNothing*.

Crowd-sourcing ethical preferences

- **Problem**: infer a set of logic-based rules that can take decisions using ethical preferences. In this case, determine whether to take control of the vehicle or not.
- **Approach 1**: we tried to infer a set of rules using first-order logic, but we got very poor results because:
  1. People have different ethical values and the case base is largely inconsistent.
  2. Even a single user can contradict him/her-self by selecting different answers for two situations that are almost identical.
  3. First-order logic inference has very poor results for noisy data.
- **Approach 2**: extend the logic rules with probabilistic annotations and infer a set of this type of rules.

Probabilistic logic induction

Probabilistic logic induction

- A **probabilistic rule**:

$$p :: H \leftarrow B_1, \ldots, B_k$$

  is composed of a probabilistic annotation $p \in [0, 1]$, a head atom H and a finite list of body atoms $B_1, \ldots, B_k$.
- We consider **atoms** of the form $P(t_1, \ldots, t_n)$ is a predicate P of arity $n \in \mathbb{N}_0$ applied to terms $t_1, \ldots, t_n$ (constants, variables or functors).
- The rules we consider don't contain **negation**.
- **Intuition**: $p :: H \leftarrow B_1, \ldots, B_k$ represents that there is a p chance of $H \leftarrow B_1, \ldots, B_k$ holding, **NOT** that, given $B_1, \ldots, B_k$, there is a p chance that H holds.

Probabilistic logic induction

- A ProbLog [De Raedt and Kimmig, 2015] *program* $T = \{p_1 :: r_1, \ldots, p_n :: r_n\}$ is a finite set of probabilistic rules.
- Given a finite set of possible grounding substitutions $\{\theta_{i,1}, \ldots, \theta_{i,m_i}\}$ for each probabilistic rule $p_i :: r_i \in T$, a ProbLog program T defines a probability distribution over the subsets $L \subseteq L_T$ of possible groundings $L_T = \{r_1\theta_{1,1}, \ldots, r_1\theta_{1,m_1}, \ldots, r_n\theta_{n,1}, \ldots, r_n\theta_{n,m_n}\}$ of (non-annotated) rules in T as:

$$P(L \mid T) = \prod_{r_i\theta_j \in L} p_i \prod_{r_i\theta_j \in L_T \setminus L} (1 - p_i)$$

- The *success probability* of a query q (i.e. finite conjunction of atoms) is the overall probability that a random subset $L \subseteq L_T$ entails q:

$$P_s(T \models q) = \sum_{\substack{L \subseteq L_T \\ L \cup T \models q}} P(L \mid T)$$

Probabilistic logic induction

- An toy example of querying (surfing):

```
sunshine(t).
windStrong(t).
0.5::goSurfing(X)←sunshine(X).
0.7::goSurfing(X)←windStrong(X).
q1←goSurfing(t).
```

- There is only one grounding substitution $\theta = \{X \leftarrow t\}$.

$$L_T = \{r_1 = goSurfing(t) \leftarrow sunshine(t).,$$
$$r_2 = goSurfing(t) \leftarrow windStrong(t).\}$$

- We calculate:
  $P(q1) = P(\{r_1, \neg r_2\}) + P(\{r_1, r_2\}) + P(\{\neg r_1, r_2\}) =$
  0.5*0.3+0.5*0.7+0.5*0.7=**0.85**.

Probabilistic logic induction

- As far as complexity results go, probabilistic logic querying has been shown to be polynomial on the size of the knowledge base for some very restrictive classes of rules [De Raedt and Kimmig, 2015].
- In general, querying can easily turn intractable. Therefore many algorithms and heuristics are being developed.

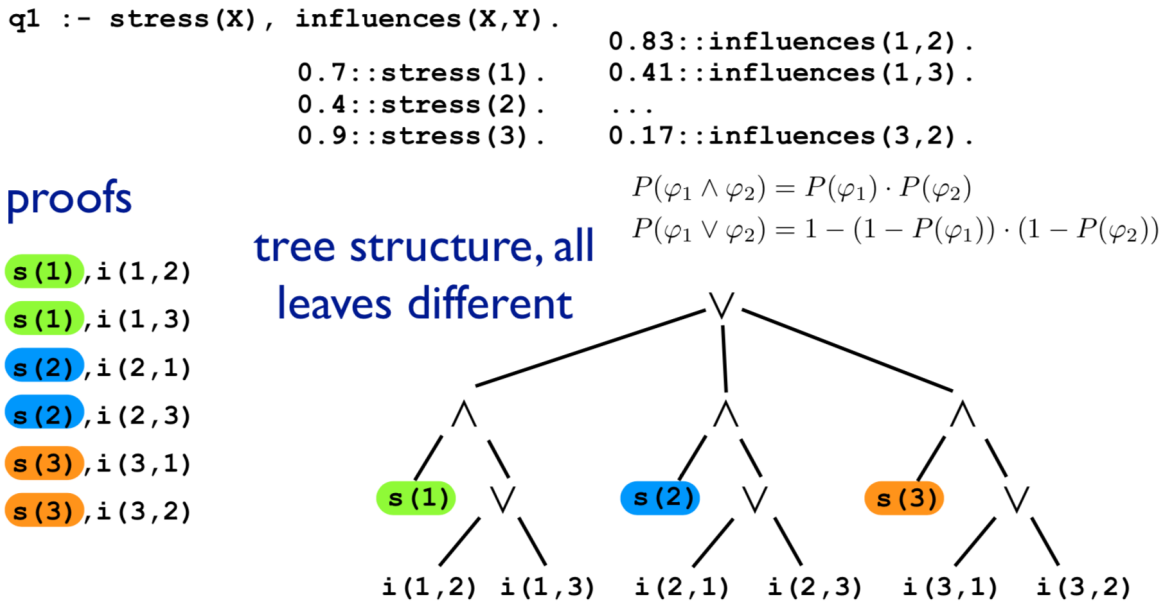

Figure 3: ProbLog querying

Probabilistic logic induction

- **Probabilistic logic induction** [De Raedt et al., 2015] addresses the task of inferring a set of probabilistic rules that justify a target predicate from examples and background knowledge.
- **Problem setting**: given the following:
  1. A set of examples E, composed of pairs (x_i, p_i) where x_i is a grounding for the target predicate t and p_i its probability,
  2. A background theory B containing information related to the examples in the form of a ProbLog program,
  3. A loss function $\text{loss}(H, B, E)$, measuring the error of a hypothesis (set of rules) H w.r.t B and E (in [De Raedt et al., 2015]: $loss(H, B, E) = \sum_{(x_i, p_i) \in E} |P_s(B \cup H \models x_i) - p_i|$), and
  4. A space of possible clauses L_h specified as in [Muggleton, 1995].

  Find a hypothesis $H \subseteq L_h$ such that $H = \underset{H' \subseteq L_h}{\text{argmin}}\ loss(H', B, E)$.

Probabilistic logic induction

- An toy example of induction, given the KB:

```
0.8::windStrong(t1).  0.7::sunshine(t1).
0.8::windStrong(t2).  0.2::sunshine(t2).
0.4::windStrong(t3).  0.1::sunshine(t3).
```

- And the example base:

```
0.7::goSurfing(t1).
0.5::goSurfing(t2).
0.2::goSurfing(t3).
```

- Inference results in the theory:

```
0.625::goSurfing(A)←windStrong(A).
0.95238095::goSurfing(A)←sunshine(A),windStrong(A).
```

Inferring ethical preferences

Inferring ethical preferences

Consider a novel case, characterized by:

$$\Sigma_{s'} = \{respectAutonomy(s', takeControl, false), preventHarm(s', takeControl, false), preventCollision(s', takeControl, true), respectAutonomy(s', doNothing, true), preventHarm(s', doNothing, true), preventCollision(s', doNothing, true)\}$$

We aggregate past related cases using probabilistic logic induction:

$$0.8 :: answer(S, A) \leftarrow preventHarm(S, A, true), preventCollision(S, A, false)$$

$$0.8 :: answer(S, A) \leftarrow respectAutonomy(S, A, true), preventHarm(S, A, true)$$

$$0.2 :: answer(S, A) \leftarrow withinSpeedLimit(S, A, true)$$

$$0.2 :: answer(S, A) \leftarrow preventCollision(S, A, true)$$

Then, we compute the query $q = answer(s', A)$:

$$answer(s', doNothing) : 0.84$$

$$answer(s', takeControl) : 0.2$$

Inferring ethical preferences

- As mentioned before, our approach consists on generating a set of probabilistic rules from similar past cases to answer ethical dilemmas.
- In a nutshell, we:
  1. Select the k most similar ethical dilemmas using all their features.
  2. Generate a set of probabilistic rules that identify which answer to select based on a **subset** of these features.
  3. Query it for each possible solution to the problem.

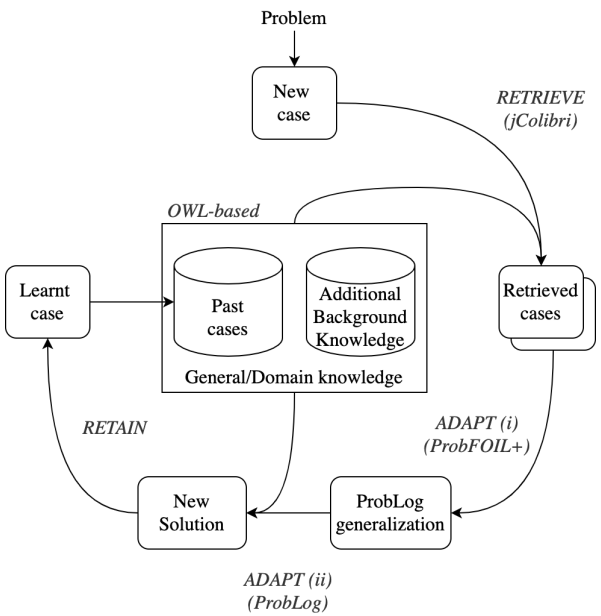

Figure 4: Overall architecture

Implementation

Implementation

- For the representation of our framework, we used a simplified version of CBROnto [Dıaz-Agudo and González-Calero, 2002], an OWL-based domain-agnostic ontology for representing CBR systems.
- The *retrieval* step is performed by using jColibri [Recio-García and Díaz-Agudo, 2006] as a Java library.
- The *adaptation* phase was implemented using a version of the ProbFOIL+ algorithm [De Raedt et al., 2015] which is publicly available [1] with minimal modifications for efficiency.
- The implementation of our system is publicly available at `https://github.com/martinjedwabny/cbr-edm`.

[1] `https://bitbucket.org/problog/prob2foil/`.

Conclusion

Conclusion

- **Contribution 1**: we present a framework for ethical decision-making through crowd-sourcing with probabilistic logic.
- **Contribution 2**: we show how to implement the adaptation phase of a CBR system with noisy data with probabilistic logic induction.
- **Contribution 3**: our framework alleviates the complexity issues of probabilistic logic induction by reducing the amount of cases to generalize (perform induction) and the amount of features to consider using the CBR methodology.
- **Contribution 4**: we provide a publicly available implementation for CBR with uncertainty.

Conclusion

- **For future work**:
  1. Expand our work in the scale of experimentation and compare the quality of our results to other similar CBR systems.
  2. Compare the performance of our reduction of the adaptation step in the CBR cycle to other different encodings.
  3. Test our implementation with other datasets coming from both knowledge-intensive and knowledge-light CBR domains.

Thank you

Thank you for listening!

References i


Aamodt, A. and Plaza, E. (1994).
Case-based reasoning: Foundational issues, methodological variations, and system approaches.
AI communications, 7(1):39–59.


De Raedt, L., Dries, A., Thon, I., Van den Broeck, G., and Verbeke, M. (2015).
Inducing probabilistic relational rules from probabilistic examples.
In *Proceedings of 24th international joint conference on artificial intelligence (IJCAI)*, volume 2015, pages 1835–1842. IJCAI-INT JOINT CONF ARTIF INTELL.


De Raedt, L. and Kimmig, A. (2015).
Probabilistic (logic) programming concepts.
Machine Learning, 100(1):5–47.

References ii


Dıaz-Agudo, B. and González-Calero, P. A. (2002).
Cbronto: a task/method ontology for cbr.
Procs. of the 15th International FLAIRS, 2:101–106.


Muggleton, S. (1995).
Inverse entailment and progol.
New generation computing, 13(3-4):245–286.


Recio-García, J. A. and Díaz-Agudo, B. (2006).
Ontology based cbr with jcolibri.
In *International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pages 149–162. Springer.


Tolmeijer, S., Kneer, M., Sarasua, C., Christen, M., and Bernstein, A. (2020).
Implementations in machine ethics: A survey.
ACM Computing Surveys (CSUR), 53(6):1–38.