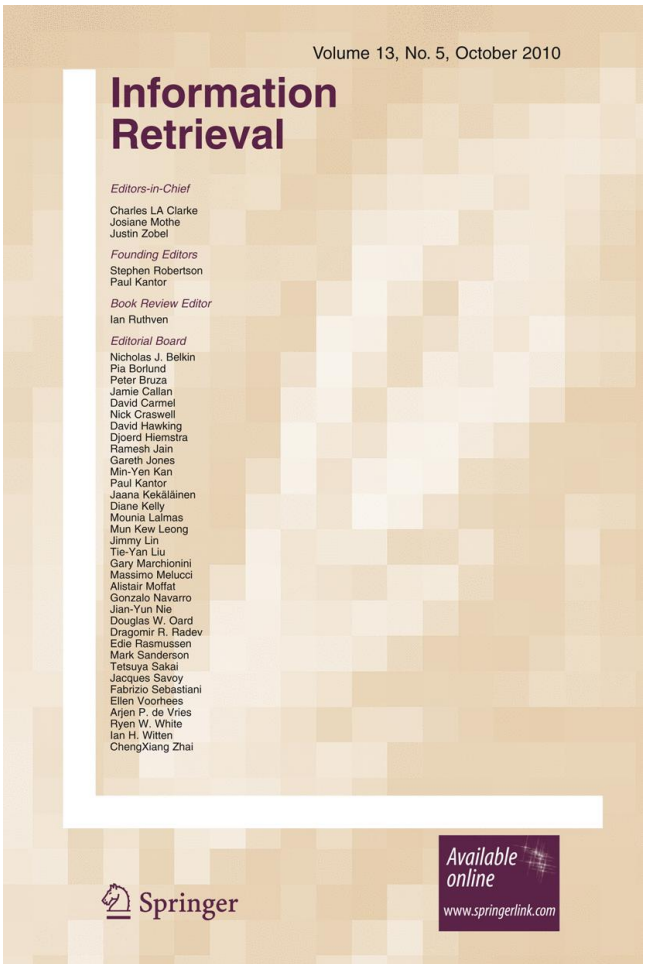
ISSN 1386-4564, Volume 13, Number 5



Inf Retrieval (2010) 13:568–600
DOI 10.1007/s10791-009-9125-9

Entity ranking in Wikipedia: utilising categories, links and topic difficulty prediction

Jovan Pehcevski · James A. Thom · Anne-Marie Vercoustre · Vladimir Naumovski

Received: 8 May 2009 / Accepted: 22 December 2009 / Published online: 9 January 2010

Abstract Entity ranking has recently emerged as a research field that aims at retrieving entities as answers to a query. Unlike entity extraction where the goal is to tag names of entities in documents, entity ranking is primarily focused on returning a ranked list of relevant entity names for the query. Many approaches to entity ranking have been proposed, and most of them were evaluated on the INEX Wikipedia test collection. In this paper, we describe a system we developed for ranking Wikipedia entities in answer to a query. The entity ranking approach implemented in our system utilises the known categories, the link structure of Wikipedia, as well as the link co-occurrences with the entity examples (when provided) to retrieve relevant entities as answers to the query. We also extend our entity ranking approach by utilising the knowledge of predicted classes of topic difficulty. To predict the topic difficulty, we generate a classifier that uses features extracted from an INEX topic definition to classify the topic into an experimentally predetermined class. This knowledge is then utilised to dynamically set the optimal values for the retrieval parameters of our entity ranking system. Our experiments demonstrate that the use of categories and the link structure of Wikipedia can significantly improve entity

This paper incorporates work published in several conferences by Pehcevski et al. (2008), Thom et al. (2007) and Vercoustre et al. (2008a, b, 2009).

J. Pehcevski (✉)
Faculty of Informatics, European University, Skopje, Macedonia
e-mail: jovan.pehcevski@eurm.edu.mk; jovan.pehcevski@acm.org

J. A. Thom
School of Computer Science, RMIT University, Melbourne, VIC, Australia
e-mail: james.thom@rmit.edu.au

A.-M. Vercoustre
INRIA, Rocquencourt, France
e-mail: anne-marie.vercoustre@inria.fr

V. Naumovski
T-Mobile, Skopje, Macedonia
e-mail: vladimir.naumovski@t-mobile.com.mk

ranking effectiveness, and that topic difficulty prediction is a promising approach that could also be exploited to further improve the entity ranking performance.

Keywords Entity ranking · INEX · XML Retrieval · Wikipedia

1 Introduction

The traditional entity extraction problem is to extract named entities from plain text using natural language processing techniques or statistical methods and intensive training from large collections. The primary goal is to tag those entities and use the tag names to support future information retrieval. *Entity ranking* has recently emerged as a research field that aims at retrieving entities as answers to a query. Here the goal is not to tag the names of the entities in documents but rather to get back a list of the relevant entity names. It is a generalisation of the expert search task explored by the TREC Enterprise track (Soboroff et al. 2006), except that instead of ranking people who are experts in a given topic, other types of entities such as organisations, countries, or locations can also be retrieved and ranked. For example, the query "European countries where I can pay with Euros" (de Vries et al. 2008) should return a list of entities representing relevant countries, and not a list of entities about the Euro and similar currencies.

The INitiative for Evaluation of XML retrieval (INEX) started the XML Entity Ranking (XER) track in 2007 (de Vries et al. 2008) with the goal of creating a test collection for entity ranking using the Wikipedia XML document collection (Denoyer and Gallinari 2006). The XER track was run again in 2008, introducing new tasks, topics and pooling strategies aiming at expanding and improving the XER test collection (Demartini et al. 2009).

There have been two main tasks in the INEX XER track:

task 1 (*entity ranking*), with the aim of retrieving entities of a given category that satisfy a topic described in natural language text; and
task 2 (*list completion*), where given a topic text and a small number of entity examples, the aim was to complete this partial list of answers.

The inclusion of the target category (in the first task) and entity examples (in the second task) makes these quite different from the task of full-text retrieval, and the combination of the query and entity examples (in the second task) makes it somewhat different from the task addressed by an application such as Google Sets[1] where only entity examples are provided.

In this paper, we describe our approach to ranking entities from the Wikipedia XML document collection. Our approach is based on the following hypotheses:

1. A good entity page is a page that answers the query, or a query extended with names of target categories (task 1) or entity examples (task 2).
2. A good entity page is a page associated with a category close to the target category (task 1) or to the categories of the entity examples (task 2).
3. A good entity page is referred to by a page answering the query; this is an adaptation of the HITS (Kleinberg 1999) algorithm to the problem of entity ranking.
4. A good entity page is referred to by focused contexts with many occurrences of the entity examples (task 2). A broad context could be the full page that contains the entity

[1] http://www.labs.google.com/sets

examples, while smaller and more narrow focused contexts could be XML elements such as paragraphs, lists, or tables.

Different approaches to entity ranking have been proposed and evaluated on the INEX Wikipedia XER test collection, which resulted in many advances to this research field (de Vries et al. 2008; Demartini et al. 2009). However, little attention has been put on the impact of the different types (or classes) of topics on the entity ranking performance.

Predicting query difficulty in information retrieval (IR) has been the subject of a SIGIR workshop in 2005 that focused both on prediction methods for query difficulty and on the potential applications of those predicted methods (Carmel et al. 2005). The applications included re-ranking answers to a query, selective relevance feedback, and query rewriting. On the other hand, the distinction between easy and difficult queries in IR evaluation is relatively recent but offers important insights and new perspectives (He and Ounis 2006; Zhou and Croft 2007). The difficult queries are defined as the ones on which the evaluated systems are the less successful. The initial motivation for query difficulty prediction was related to the need for evaluation measures that could be applied for robust evaluation of systems across different collections (Voorhees 2004; Webber et al. 2008). Recently Mizzaro (2008), also advocated that the evaluation measures should reward systems that return good results on difficult queries more than on easy ones, and penalise systems that perform poorly on easy queries more than on difficult ones.

By taking into account our hypotheses and the above arguments, the main research question we address in this paper is: How can the knowledge of Wikipedia categories, link structure and topic difficulty prediction be utilised to improve the effectiveness of entity ranking?

The remainder of this paper is organised as follows. We begin with a description of the INEX XER track in Sect. 2, which also includes properties of the various Wikipedia topic data sets: the two sets of INEX 2007 XER topics used as training and testing data sets, and the set of INEX 2008 XER topics used as a testing data set. The architecture of our entity ranking system, which uses Wikipedia categories and links to improve entity ranking effectiveness, is described in Sect. 3. Using the INEX 2007 XER training data set, we select the optimal system retrieval parameters when using categories (Sect. 4) and links (Sect. 5). These parameters are then re-evaluated in Sect. 6 using the INEX 2007 XER testing data set. In Sect. 7, we first investigate topic difficulty prediction by using the INEX 2007 XER testing data set (for training) and then evaluate the effectiveness of this approach on the INEX 2008 XER testing data set (for testing). We review related work in Sect. 8 and conclude the paper in Sect. 9.

2 Entity ranking at INEX

In this section, we provide a detailed description of the INEX XML entity ranking (XER) track. We first describe some properties of the INEX Wikipedia XML document collection, and then discuss the various INEX XER training and testing data sets.

2.1 INEX Wikipedia XML document collection

Wikipedia is a well known web-based, multilingual, free content encyclopedia written collaboratively by contributors from around the world. Denoyer and Gallinari (2006) have developed an XML-based corpus based on a snapshot of the Wikipedia, which was used by various INEX tracks from 2006 until 2008. This is the collection we use in this paper, and

although it differs from the real Wikipedia in some respects (size, document format, category tables), it is nevertheless a very realistic approximation. From 2009 onwards a new XML snapshot of the Wikipedia is being used by INEX.

2.1.1 Entities in Wikipedia

The entities have a name (the name of the corresponding page) and a unique ID in the collection. When mentioning such an entity in a new Wikipedia article, authors are encouraged to link occurrences of the entity name to the page describing this entity. This is an important feature as it makes it easy to locate potential entities, which is a major issue in entity extraction from plain text. However, in this collection, not all potential entities have been associated with corresponding pages.

The INEX XER topics have been carefully designed to make sure there is a sufficient number of answer entities. For example, in the Euro page (see Fig. 1), all the underlined hypertext links can be seen as occurrences of entities that are each linked to their corresponding pages. In this figure, there are 18 entity references of which 15 are country names; specifically, these countries are all "European Union member states", which brings us to the notion of categories in Wikipedia.

2.1.2 Categories in Wikipedia

Wikipedia also offers categories that authors can associate with Wikipedia pages. There are 113,483 categories in the INEX Wikipedia XML collection, which are organised in a graph of categories. Each page can be associated with many categories (2.28 as an average).

Wikipedia categories have unique names (e.g. "France", "European Countries", "Countries"). New categories can also be created by authors, although they have to follow Wikipedia recommendations in both creating new categories and associating them with pages. For example, the Spain page is associated with the following categories: "Spain", "European Union member states", "Spanish-speaking countries", "Constitutional monarchies" (as well as some other Wikipedia administrative categories).

Some properties of Wikipedia categories include:

- a category may have many subcategories and parent categories;
- some categories have many associated pages (a large *extension*), while others have a smaller extension;
- a page that belongs to a given category extension generally does not belong to its ancestors' extension; for example, the page Spain does not belong to the category "European countries";
- the sub-category relation is not always a subsumption relationship; for example, "Maps of Europe" is a sub-category of "Europe", but the two categories are not in an *is-a* relationship; and
- there are cycles in the category graph.

Yu et al. (2007) explore these properties of Wikipedia categories in more detail.

"The **euro** ... is the official currency of the Eurozone (also known as the Euro Area), which consists of the European states of Austria, Belgium, Finland, France, Germany, Greece, Ireland, Italy, Luxembourg, the Netherlands, Portugal, Slovenia and Spain, and will extend to include Cyprus and Malta from 1 January 2008."

Fig. 1 Extract from the Euro Wikipedia page

```
<?xml version="1.0" encoding="utf-8"?>
<inex_topic topic_id="0" query_type="XER" ct_no="0">
<title>European countries where I can pay with Euros</title>
<description>I want a list of European countries where
I can pay with Euros.</description>
<narrative>Each answer should be the article about a
specific European country that uses the
Euro as currency.</narrative>
<categories>
   <category id="185">"european countries"</category>
</categories>
<entities>
   <entity id="10581">"France"</entity>
   <entity id="11867">"Germany"</entity>
   <entity id="26667">"Spain"</entity>
</entities>
</inex_topic>
```

Fig. 2 Example of an INEX XML entity ranking topic (taken from the initial INEX 2007 XER training set)

When searching for entities it is natural to take advantage of the Wikipedia categories since they give a hint on whether the retrieved entities are of the expected type. For example, when looking for entities of type "authors", pages associated with the category "Novelist" are more likely to be relevant than pages associated with the category "Book".

2.2 INEX XER Training and testing data sets

An example of an INEX XML entity ranking topic is shown in Fig. 2. In this example, the `title` field contains the plain content only query, the `description` provides a natural language description of the information need, and the `narrative` provides a detailed explanation of what makes an entity answer relevant. In addition to these fields, the `categories` field provides the category of the expected entity answers (task 1: entity ranking), while the `entities` field provides a few of the expected entity answers for the topic (task 2: list completion).

Since there was no existing set of topics with corresponding relevance assessments for entity ranking prior to 2007, we developed our own data set that we made available as a training set for other participants in the INEX 2007 XER track. This training data set is based on a selection of topics from the INEX 2006 ad hoc track, since most of these ad hoc topics reflect real-life tasks represented by queries very similar to short Web queries (Kamps and Larsen 2006). We chose 27 topics that we considered were of an "entity ranking" nature, where for each page that had been assessed as containing relevant information, we reassessed whether or not it was an entity answer, and whether it *loosely* belonged to a category of entities we had *loosely* identified as being the target of the topic. If there were entity examples mentioned in the original topic these were usually used as entity examples in the entity topic. Otherwise, a selected number (typically 2 or 3) of entity examples were chosen somewhat arbitrarily from the relevance assessments. To this set of 27 topics we also added the Euro topic example (shown in Fig. 2) that we had created by hand from the original INEX description of the entity ranking track (de Vries et al. 2008), resulting in total of 28 entity ranking topics. This became the INEX 2007 XER training data set.

The final INEX XER test collection includes three sets of topics—28 topics for training (2007), 46 topics for testing (2007),[2] and 35 more topics for testing (2008), all with

[2] These 46 topics in the INEX 2007 XER testing data set can also be used for training when testing using the INEX 2008 XER testing data set.

corresponding relevance assessments available. Most of these topics were proposed and assessed by the track participants (de Vries et al. 2008; Demartini et al. 2009).

Eight participating groups submitted in total 35 XER runs in 2007, while six participants submitted another 33 XER runs in 2008. The top 100 entity answers from these runs were pooled which resulted in about 500 entities by topic that were then assessed by the track participants. Relevance judging is simpler than in other INEX tasks, as assessors only need to determine whether an article (representing an entity) is a correct answer or not.

3 Our entity ranking system

In this section, we describe our approach to ranking entities from the Wikipedia XML document collection.

We have built a system that implements an entity ranking approach where candidate entity pages are ranked by combining three different scores: a category score, a linkrank score, and the initial full-text retrieval score. We use Zettair,[3] a full-text search engine developed at RMIT University, which returns pages ranked by their similarity score to the query. We use the Okapi BM25 similarity measure in Zettair as it has been shown to be effective on the INEX Wikipedia collection (Awang Iskandar et al. 2007).

As shown in Fig. 3, our system involves several modules for processing a query, submitting it to the search engine, applying our entity ranking algorithms, and finally returning a ranked list of entities.

The architecture provides a general framework for evaluating entity ranking which allows for replacing some modules by more advanced modules, or by providing a more efficient implementation of a module. It also uses an evaluation module (not shown in the figure) to assist in tuning the system by varying the parameters and to globally evaluate the entity ranking approach. The architecture involves the following modules.

1. The *topic module* takes an INEX topic as input (as the topic example shown in Fig. 2) and generates the corresponding Zettair query and the list of target categories and entity examples (as one option, the names of target categories or example entities may be added to the query).
2. The *search module* sends the query to Zettair and returns a list of scored Wikipedia pages (typically 1,500). The assumption is that a good entity answer can be a page that matches the query.
3. The *link extractor module* extracts the links from an experimentally predetermined number (N) of highly ranked pages,[4] together with the information about the paths of the links (XML XPaths). This results in total of N1 + (the initial) N candidate pages for entity answers. The assumption is that a good entity answer can be a page that is referred to by a highly ranked page matching the query; this is an adaptation of the Google PageRank (Brin and Page 1998) and HITS (Kleinberg 1999) algorithms to the problem of entity ranking.

[3] http://www.seg.rmit.edu.au/zettair/

[4] We discarded external links and some internal collection links that do not refer to existing pages in the INEX Wikipedia collection. The number N has been kept to a relatively small value mainly for performance purposes, since Wikipedia pages contain many links that would need to be extracted. We carried out some preliminary experiments with different values of the parameter N, by varying it between 5 and 100 with a step of 5, and found that N = 20 was a good compromise between maintaining satisfactory performance and discovering more potentially good entities.

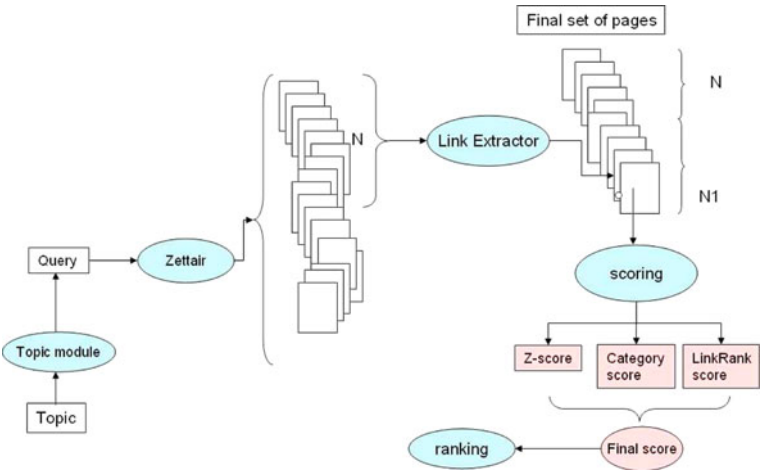


Fig. 3 Our system architecture for XML entity ranking

4. The *category similarity module* calculates a score $S_C(t)$ for each candidate page t based on the assumption that a good entity answer is a page associated with a category close to the target categories (task 1) or to categories attached to the entity examples (task 2). We explore alternatives for this score in Sect. 4.
5. The *linkrank module* calculates a score $S_L(t)$ for each candidate page t based on the assumption that a good entity answer is a page referred to from contexts with many occurrences of the entity examples. A coarse context would be the full page that contains the entity examples. Smaller and better contexts may be elements such as paragraphs, lists, or tables. In Sect. 5 we explore variations to exploit locality of links around the entity examples. For our initial experiments on category scores in Sect. 4, we use a very basic linkrank function that combines the Zettair score $z(p_r)$ of each page p_r (in the top N pages returned by the search engine) with the number (possibly zero) of reference links $\#links(p_r, t)$ from the page p_r to the candidate entity answer page t:

$$S_L(t) = \sum_{r=1}^{N} z(p_r) * \#links(p_r, t) \tag{1}$$

6. The *full-text retrieval module* calculates a score $S_Z(t)$ for each candidate page t based on its initial Zettair score. The Z score assigns the initial Zettair full-text score to an answer entity page. If the answer page does not appear in the final list of 1,500 ranked pages returned by Zettair, then its Z score is zero:

$$S_Z(t) = \begin{cases} z(t) & \text{if page } t \text{ was returned by Zettair} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

The Z score is not the same as the plain Zettair score, since our system extracts new N1 entities (pages) from the links contained in the highest N pages returned by Zettair; these new pages may or may not be included in the initial 1,500 pages returned by Zettair.

The final global score $S(t)$ for an answer entity page is calculated as a linear combination of three normalised scores, the linkrank score $S_L(t)$, the category similarity score $S_C(t)$, and the Z score $S_Z(t)$:

$$S(t) = \alpha S_L(t) + \beta S_C(t) + (1 - \alpha - \beta) S_Z(t) \tag{3}$$

where α and β are two parameters that can be tuned differently depending on the entity retrieval task. Combinations for these two parameters are explored in the training phases in the next two sections. The most effective combination is then used in the final evaluation on the testing data set in Sect. 6.

4 Using Wikipedia categories

To make use of the Wikipedia categories in entity ranking, in this section we define similarity functions between:

- categories of answer entities and target categories (for task 1), and
- categories of answer entities and the set of categories attached to the entity examples (for task 2).

We present results that investigate the effectiveness of our entity ranking approach for the two entity ranking tasks using variations of the category score combined with the basic linkrank score. For these experiments we use the INEX 2007 XER training data set comprising 28 entity ranking topics. We use mean average precision (MAP) as our primary method of evaluation, but also report results using several alternative information retrieval measures: mean of P[5] and P[10] (mean precision at top 5 or 10 entities returned), and mean R-precision (R-precision for a topic is the P[R], where R is the number of entities that have been judged relevant for the topic). When dealing with entity ranking, the ultimate goal is to retrieve all the answer entities at the top of the ranking, and so we believe that MAP (as our primary method of evaluation) may be more suitable than the other measures in capturing these aspects.

4.1 Task 1: entity ranking

For this task we investigated the effectiveness of our category similarity module when varying the extensions of the set of categories attached to both the target categories and the answer entities, and the impact of these variations on the global score produced by our entity ranking system.

4.1.1 Investigating category similarity approaches

We first define a basic similarity function that computes the ratio of common categories between the set of categories $\texttt{cat}(t)$ associated to an answer entity page t and the set $\texttt{cat}(C)$ which is the union of the provided target categories C:

$$S_C(t) = \frac{|\texttt{cat}(t) \cap \texttt{cat}(C)|}{|\texttt{cat}(C)|} \tag{4}$$

The target categories will be generally very broad, so it is to be expected that the answer entities would not generally belong to these broad categories. Accordingly, we defined

several extensions of the set of categories, both for the target categories and the categories attached to answer entities.

The extensions are based on using sub-categories and parent categories in the graph of Wikipedia categories. We define $\texttt{cat}_\texttt{d}(C)$ as the set containing the target category and its sub-categories (one level down) and $\texttt{cat}_\texttt{u}(t)$ as the set containing the categories attached to an answer entity t and their parent categories (one level up). Similarity function can then be defined using the same ratio as above except that $\texttt{cat}(t)$ is replaced with $\texttt{cat}_\texttt{u}(t)$ and $\texttt{cat}(C)$ with $\texttt{cat}_\texttt{d}(C)$. We performed preliminary experiments by adding different ancestor and descendant categories of any levels from the Wikipedia category graph, but found that these extensions were less effective than those we use above.

Another approach is to use lexical similarity between categories. For example, "european countries" is lexically similar to "countries" since they both contain the word "countries" in their names. We use an information retrieval approach to retrieve similar categories: we have indexed all the categories using their names as corresponding documents. By sending the category names C as a query to the search engine, we then retrieve all the categories that are lexically similar to C (we do not use any stemming and/or stopping, just plain name-only search). We keep the top M ranked categories and add them to C to form the set $\texttt{Ccat}(C)$ (in our initial experiments we set $\text{M} = 10$). We then use the same similarity function as before, where $\texttt{cat}(C)$ is replaced with $\texttt{Ccat}(C)$. We also experiment with two alternative approaches: by sending the title of the topic T as a query to the search engine (denoted as $\texttt{Tcat}(C)$); and by sending both the title of the topic T and the category names C as a query to the search engine (denoted as $\texttt{TCcat}(C)$).

An alternative approach of using lexical similarity between categories is to index the categories using their names and the names of all their attached entities as corresponding documents. For example, if C = "countries", the retrieved set of categories $\texttt{Ccat}(C)$ may contain not only the categories that contain "countries" in their names, but also categories attached to entities whose names are lexically similar to "countries".

The results of these investigations are shown in Table 1(C and CE).[5] Several observations can be drawn from these results.

First, the choice of using the Zettair category index can dramatically influence the entity ranking performance. When cross-comparing the results in the two tables, we observe that the three lexical similarity runs using the Zettair index of category names substantially outperform the corresponding runs using the Zettair index of category and entity names. The differences in performance are all statistically significant ($p < 0.05$).

Second, the run that uses the query that combines the terms from the title and the category fields of an INEX topic ($\texttt{TCcat}(C)$-$\texttt{cat}(t)$) performs the best among the three runs using lexical similarity, and overall it also performs the best among the five runs when using the Zettair index of category names. However, the differences in performance between this and the other four runs are not statistically significant.

Third, extending the set of categories attached to both the target categories and the answer entities overall does not result in an improved performance over the non-extended sets, although there are some (non-significant) early precision improvements. Tsikrika et al. (2008) also found that extending the set of target categories with ancestors of any levels decreases the performance, although contrary to the results from our preliminary

[5] The first two runs do not use any of Zettair's category indexes and are included for comparison.

Table 1 Performance scores for runs using different retrieval strategies in our category similarity module (α0.0–β1.0), obtained for task 1 by different evaluation measures

Run	P[r]		R-prec	MAP
	5	10		
(C) Index of category names				
$\texttt{cat}(C)$-$\texttt{cat}(t)$	0.229	0.250	0.215	0.196
$\texttt{cat}_\texttt{d}(C)$-$\texttt{cat}_\texttt{u}(t)$	0.243	0.246	0.209	0.185
$\texttt{Ccat}(C)$-$\texttt{cat}(t)$	0.214	0.250	0.214	0.197
$\texttt{Tcat}(C)$-$\texttt{cat}(t)$	**0.264**	0.261	0.239	0.216
$\texttt{TCcat}(C)$-$\texttt{cat}(t)$	**0.264**	**0.286**	**0.247**	**0.226**
(CE) Index of category and entity names				
$\texttt{cat}(C)$-$\texttt{cat}(t)$	0.229	**0.250**	**0.215**	**0.196**
$\texttt{cat}_\texttt{d}(C)$-$\texttt{cat}_\texttt{u}(t)$	**0.243**	0.246	0.209	0.185
$\texttt{Ccat}(C)$-$\texttt{cat}(t)$	0.157	0.171	0.149	0.148
$\texttt{Tcat}(C)$-$\texttt{cat}(t)$	0.171	0.182	0.170	0.157
$\texttt{TCcat}(C)$-$\texttt{cat}(t)$	0.207	0.214	0.175	0.173

For the three runs using lexical similarity, the Zettair index comprises documents containing category names (C), or documents containing category names and names of entities associated with the category (CE). The number of category answers retrieved by Zettair is M = 10. For each measure, the best performing score is shown in bold

experiments they observed performance improvements when using descendants of target categories of up to third level in the Wikipedia category graph. The different (language modelling) retrieval method they use might be a reason for this, although we plan to re-investigate this behaviour on larger test collections in the future.

4.1.2 Investigating the parameter M

The above results show that the best effectiveness for our category similarity module (α0.0–β1.0) is achieved when using the Zettair index of category names, together with the query strategy that combines the terms from the title and the category fields of an INEX topic. For these experiments we used a fixed value M = 10 for the parameter M that represents the number of category answers retrieved by Zettair. However, since this was an arbitrary choice we also investigated whether a different value of M could also have a positive impact on the retrieval effectiveness. We therefore varied M from 1 to 20 in steps of 1, and measured the MAP scores achieved by our best performing $\texttt{TCcat}(C)$-$\texttt{cat}(t)$ run using the Zettair index of category names.

Figure 4 shows the results of this investigation. We observe that a value of 5 for the parameter M yields the highest MAP score (0.242) for our category similarity module, which is a 7% relative performance improvement over the MAP score obtained with M = 10. This performance improvement is statistically significant ($p < 0.05$).

For completeness, for each of the 20 values for the parameter M we also compared the performances of the two runs that do not use lexical similarity to performances of the three runs that do use it. The same performance trend between the runs was also observed as that shown in Table 1(C and CE).

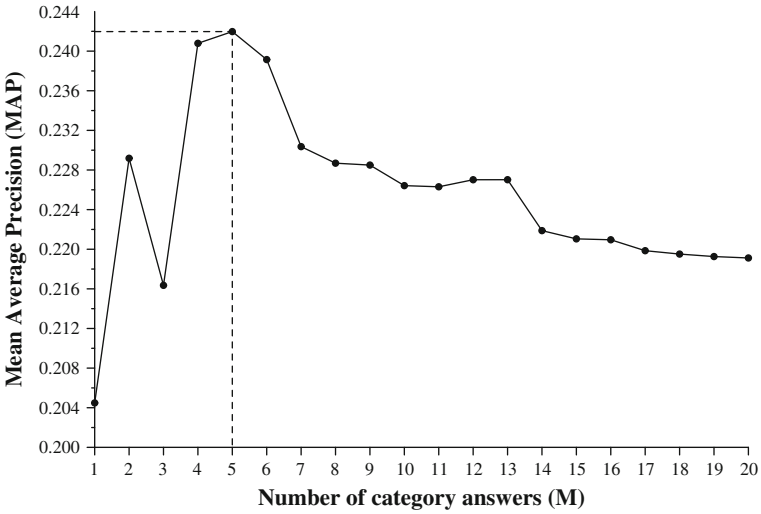


Fig. 4 Investigating the optimal value for the number of category answers retrieved by Zettair, when using the run TCcat(C)-cat(t)

4.1.3 Investigating the combining parameters α and β

To find the best score that could be achieved by our entity ranking approach for task 1, we used the run TCcat(C)-cat(t) with the optimal value M = 5 and investigated various combinations of scores obtained from the three modules. We calculated MAP over the 28 topics in our training collection, as we varied α from 0 to 1 in steps of 0.1. For each value of α, we also varied β from 0 to $(1 - \alpha)$ in steps of 0.1. We found that the highest MAP score (0.287) is achieved for $\alpha = 0.1$ and $\beta = 0.8$. This is a 19% relative performance improvement over the best score achieved by using only the category module (α0.0–β1.0). This performance improvement is statistically significant ($p < 0.05$). We also calculated the scores using mean R-precision instead of MAP as our evaluation measure, and again observed the same performance behaviour and optimal values for the two parameters.

4.2 Task 2: list completion

For this task we carried out two separate investigations. First, as with task 1 we wanted to investigate the effectiveness of our category similarity module when varying the extensions of the set of categories attached to both the example and the answer entities. Second, for the best category similarity approach we investigated the optimal values for the α and β parameters, with the aim of finding the best score that could be achieved by our entity ranking approach for task 2.

4.2.1 Investigating category similarity approaches

In task 2, the categories attached to entity examples are likely to correspond to very specific categories, just like those attached to the answer entities. We define a similarity function that computes the ratio of common categories between the set of categories

Table 2 Performance scores for runs using different retrieval strategies in our category similarity module (α0.0–β1.0), obtained for task 2 by different evaluation measures

Run	P[r]		R-prec	MAP
	5	10		
$\texttt{cat}(E)$-$\texttt{cat}(t)$	**0.536**	**0.393**	**0.332**	**0.338**
$\texttt{cat}(E)$-$\texttt{cat}_\texttt{u}(t)$	0.493	0.361	0.294	0.313
$\texttt{cat}_\texttt{u}(E)$-$\texttt{cat}(t)$	0.407	0.336	0.275	0.255
$\texttt{cat}_\texttt{u}(E)$-$\texttt{cat}_\texttt{u}(t)$	0.357	0.332	0.269	0.261

For each measure, the best performing score is shown in bold

attached to an answer entity page $\texttt{cat}(t)$ and the set of the union of the categories attached to entity examples $\texttt{cat}(E)$:

$$S_C(t) = \frac{|\texttt{cat}(t) \cap \texttt{cat}(E)|}{|\texttt{cat}(E)|} \tag{5}$$

We also expand the two sets of categories by adding the parent categories to calculate $\texttt{cat}_\texttt{u}(t)$ and $\texttt{cat}_\texttt{u}(E)$ and apply the same similarity function as above.

The results of these investigations are shown in Table 2. We observe that, as with task 1, extending the set of categories attached to either (or both) of the example and answer entities does not result in an improved performance. The differences in performance between the best performing run that does not use the extended category sets and the other three runs that use any (or both) of these sets are all statistically significant ($p < 0.05$). We also found that adding different ancestor and descendant categories of any levels from the Wikipedia category graph further decreases the entity ranking performance.

4.2.2 Investigating the combining parameters α and β

To find the best score that could be achieved by our entity ranking approach for task 2, we used the run $\texttt{cat}(E)$-$\texttt{cat}(t)$ and investigated various combinations of scores obtained from the three modules. We calculated MAP over the 28 topics in our training collection, as we used all the 66 combined values for parameters α and β. We found that the highest MAP score (0.396) was again achieved for $\alpha = 0.1$ and $\beta = 0.8$. This score is a 17% relative performance improvement over the best score achieved by using only the category module (α0.0–β1.0). The performance improvement is statistically significant ($p < 0.05$).

4.3 Comparing task 1 and task 2

To investigate which of the two query strategies (target categories or example entities) is more effective for entity ranking, we compared the scores of the best performing runs across the two tasks. Table 3 shows the results of this comparison, when separately taking into account two distinct cases: a case when using scores coming out of the category module only (α0.0–β1.0); and a case when using optimal global scores coming out of the three modules (α0.1–β0.8).

We observe that, irrespective of whether category or global scores are used by our entity ranking approach, the run that uses the set of categories attached to example entities (task 2) substantially outperforms the run that uses the set of categories identified by Zettair

Table 3 Comparing best performing runs for task 1 and task 2 for two distinct cases (using either category or global scores)

Run	P[r]		R-prec	MAP
	5	10		
Category score: α0.0–β1.0				
TCcat(C)-cat(t)	0.307	0.318	0.263	0.242
cat(E)-cat(t)	**0.536**	**0.393**	**0.332**	**0.338**
Global score: α0.1–β0.8				
TCcat(C)-cat(t)	0.379	0.361	0.338	0.287
cat(E)-cat(t)	**0.607**	**0.457**	**0.412**	**0.396**

The number of category answers retrieved by Zettair for run TCcat(C)-cat(t) is M = 5. For each case, the best results are shown in bold

using the topic title and the target categories (task 1). The differences in performance between the two runs are statistically significant ($p < 0.05$). This finding shows that using example entities is much more effective query strategy than using the loosely defined target categories, which allows for the answer entities to be identified and ranked more accurately.

5 Exploiting locality of Wikipedia links

The main assumption behind the idea of exploiting locality of links in entity ranking is that references to entities (links) located in close proximity to the entity examples, which typically appear in list-like contexts, are more likely to represent relevant entities than links that appear in other parts of the page. Here, the notion of *list* refers to grouping together objects of the same (or similar) nature. The aim is therefore to assign a bigger weight to links that co-occur with links to entity examples in such list-like contexts.

Consider the example of the Euro page shown in Fig. 1 (Sect. 2). For the topic "European countries where I can pay with Euros" where France, Germany and Spain are the three entity examples (shown in Fig. 2 in Sect. 2), we observe that the 15 countries that are members of the Eurozone are all listed in the same paragraph with the three entity examples. In fact, there are other contexts in this page where those 15 countries also co-occur together. By contrast, although there are a few references to the United Kingdom in the Euro page, it does not occur in the same context as the three examples (except for the page itself).

Statically defined contexts We have identified three types of elements that correspond to list-like contexts in the Wikipedia XML document collection: paragraphs (tag `p`); lists (tags `normallist`, `numberlist`, and `definitionlist`); and tables (tag `table`). We design two algorithms for identifying the static contexts: one that identifies the context on the basis of the leftmost occurrence of the pre-defined tags in the XPath of an extracted link (`StatL`), and another that uses the rightmost occurrence of the pre-defined tags in the XPath to identify the context (`StatR`). We do this to investigate whether the recursive occurrences of the same tag, as often found in many XML documents in the INEX Wikipedia collection, has an impact on the ability to better identify relevant entities.

Consider Table 4, where the links to entity examples are identified by their absolute XPath notations. The three non-overlapping elements that will be identified by the `StatL`

algorithm are the elements `p[1]`, `p[3]`, and `normallist[1]`, while with the `StatR` algorithm `p[5]` will be identified instead of `p[3]` in addition to also identifying the other two elements.

The main drawback of the static approach is that it requires a pre-defined list of element contexts which is totally dependent on the document collection. The advantage is that, once defined, the list-like contexts are easy to identify.

Dynamically defined contexts To determine the contexts dynamically, we adapted the concept of *coherent retrieval elements* (Pehcevski et al. 2005) initially used to identify the appropriate granularity of elements to return as answers in XML retrieval.

For the list of extracted entities corresponding to entity examples, a *Coherent Retrieval Element* (CRE) is defined as an element that represents the *lowest common ancestor* (LCA) of at least two entity examples. To identify the CREs, we sequentially process the list of extracted entity examples by considering every pair of elements, starting from the first element down to the element preceding the last element in the list. For each pair of elements, their LCA is chosen to represent a dynamic context (a CRE). After processing the last pair of elements, the resulting list of CREs is sorted in a descending order according to the XPath length of the CRE (the longer the length, the higher the rank). If two CREs have the same XPath length, the one that contains more distinct entity examples is ranked higher. Last, starting from the highest ranked CRE, we filter all the CREs that either contain or are contained by that element. We end up with a final list of (one or more) non-overlapping CREs that represent the dynamically defined contexts for the page.

For example, the two dynamic contexts that will be identified for the list of extracted entity examples shown in Table 4 are `p[1]` and `normallist[1]`. Although `body[1]` was also initially identified as a CRE, it was subsequently filtered from the final list since it overlaps with `p[1]` (the first identified CRE).

The main advantage of the dynamic approach is that it is independent of the document collection, and it does not require a pre-defined list of contexts. The possible drawback is that narrow contexts containing only one entity example (such as `p[5]` in Table 4) are never identified.

Extending the linkrank function In order to take the contexts into account, we have extended (and re-implemented) the linkrank function so that, for an answer entity page t, takes into account the Zettair score of the referring page $z(p_r)$, the number of distinct entity

Table 4 List of links referring to entity examples (France, Germany, and Spain), extracted from the Wikipedia page 9272.xml

Page		Links		
ID	Name	XPath	ID	Name
9472	Euro	/article[1]/body[1]/p[1]/collectionlink[7]	10581	France
9472	Euro	/article[1]/body[1]/p[1]/collectionlink[8]	11867	Germany
9472	Euro	/article[1]/body[1]/p[1]/collectionlink[15]	26667	Spain
9472	Euro	/article[1]/body[1]/p[3]/p[5]/collectionlink[6]	11867	Germany
9472	Euro	/article[1]/body[1]/normallist[1]/item[4]/collectionlink[1]	10581	France
9472	Euro	/article[1]/body[1]/normallist[1]/item[5]/collectionlink[2]	11867	Germany
9472	Euro	/article[1]/body[1]/normallist[1]/item[7]/collectionlink[1]	26667	Spain
9472	Euro	/article[1]/body[1]/normallist[1]/item[8]/collectionlink[1]	26667	Spain

examples in the referring page $\#ent(p_r)$, and the locality of links around the entity examples:

$$S_L(t) = \sum_{r=1}^{N} \left(z(p_r) \cdot (\#ent(p_r) + 0.5) \cdot \sum_{l_t \in L(p_r,t)} f(l_t, c_r | c_r \in C(p_r)) \right) \quad (6)$$

where l_t is a link that belongs to the set of links $L(p_r, t)$ that point from the page p_r to the answer entity t; c_r belongs to the set of contexts $C(p_r)$ around entity examples found for the page p_r; and $f(l_t, c_r)$ represents the weight associated to the link l_t that belongs to the context c_r. We add 0.5 to $\#ent(p_r)$ to allow for cases where there are no entity examples in the referring page.

The weighting function $f(l_t, c_r)$ is represented as follows:

$$f(l_t, c_r) = \begin{cases} 1 & \text{if } c_r = p_r \text{ (the context is the full page)} \\ 1 + \#ent(c_r) & \text{if } c_r = e_r \text{ (the context is an XML element)} \end{cases}$$

We now present results that investigate the effectiveness of our entity ranking approach when using different types of contexts around the entity examples. For these initial experiments we also use the INEX 2007 XER training data set comprising 28 entity ranking topics.

5.1 Full page context

We used the context of the full page to determine suitable values for the parameters α and β, and also to try out some minor variations to our entity ranking approach (such as whether or not to include the names of the entity examples in the query sent to Zettair).

We calculated MAP over the 28 topics in our test collection, as we varied α from 0 to 1 in steps of 0.1. For each value of α, we also varied β from 0 to $(1 - \alpha)$ in steps of 0.1. Table 5 shows these results, where we can also observe the benefit of combining the Z

Table 5 Mean average precision scores for runs using 66 possible α–β combinations, obtained on the 28 topics of our training collection

Alpha (α)	Beta (β)										
	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
0.0	*0.1780*	0.2040	0.2347	0.2673	0.3022	0.3139	0.3269	0.3346	0.3446	0.3491	*0.3089*
0.1	0.1886	0.2122	0.2548	0.2851	0.3121	0.3278	0.3418	0.3516	**0.3570**	0.3161	
0.2	0.1949	0.2187	0.2514	0.2911	0.3244	0.3313	0.3453	0.3481	0.3090		
0.3	0.2011	0.2273	0.2480	0.2730	0.3039	0.3219	0.3248	0.2911			
0.4	0.2092	0.2337	0.2537	0.2758	0.2859	0.2871	0.2622				
0.5	0.2173	0.2365	0.2495	0.2614	0.2656	0.2442					
0.6	0.2122	0.2271	0.2431	0.2401	0.2277						
0.7	0.2052	0.2154	0.2181	0.2020							
0.8	0.1886	0.1864	0.1750								
0.9	0.1644	0.1536									
1.0	*0.1314*										

Queries sent to Zettair include only terms from the topic title (Q). The MAP score of the plain *Zettair* run is *0.1718*. The numbers in italics show the scores obtained for each of the three individual modules. The best performing MAP score among the 66 scores in the table is shown in bold

Table 6 Performance scores for runs using the context of the full page, obtained by different evaluation measures

Run	P[r]		R-prec	MAP
	5	10		
(Q) Topic title				
Zettair	0.2286	0.2321	0.2078	0.1718
α0.0–β0.0	0.2286	0.2321	0.2135	0.1780
α0.0–β1.0	0.3643	0.3071	0.3151	0.3089
α1.0–β0.0	0.1571	0.1571	0.1385	0.1314
α0.1–β0.8	**0.4714**	**0.3857**	**0.3902**	**0.3570**
α0.2–β0.6	0.4357	0.3786	0.3751	0.3453
(QE) Topic title and entity examples				
Zettair	0.2000	0.1714	0.1574	0.1427
α0.0–β0.0	0.2000	0.1714	0.1775	0.1533
α0.0–β1.0	0.3357	0.2821	0.2749	0.2674
α1.0–β0.0	0.1857	0.1750	0.1587	0.1520
α0.1–β0.8	0.3357	0.3286	0.3109	0.3140
α0.2–β0.6	**0.3429**	**0.3357**	**0.3362**	**0.3242**

Queries sent to Zettair include only terms from the topic title (Q), or terms from the topic title and the names of entity examples (QE). For each measure, the best performing score is shown in bold

score when using only the linkrank score ($\beta = 0.0$) or when using only the category score ($\alpha = 0.0$). We found that the highest MAP (0.3570) on this data set is achieved for $\alpha = 0.1$ and $\beta = 0.8$. We also trained using mean R-precision instead of MAP as our evaluation measure, but we also observed the same optimal values for the two parameters.

We used a selected number of runs to carry out a more detailed investigation of the performance achieved by each independent module and by the optimal module combination. We also investigated whether adding names of the entity examples to the query sent to Zettair would have a positive performance impact. The results of these investigations are shown in Table 6(Q and QE).

Several observations can be drawn from these results. First, adding names of the entity examples to the query sent to Zettair generally performs worse for all but the linkrank module, for which we see a consistent performance improvement. Second, different optimal values are observed for the two parameters in the two tables, which suggest that adding the entity examples to the query can dramatically influence the retrieval performance. Third, we observe that the best entity ranking approaches are those that combine the ranking evidence from the three modules (runs α0.1–β0.8 for Q and α0.2–β0.6 for QE). With MAP, these two runs perform significantly better ($p < 0.05$) than the plain Zettair full-text retrieval run, and they are also significantly better than any of the three runs representing each individual module in our entity ranking approach.

These results therefore show that the global score (the combination of the three individual scores), optimised in a way to give more weight on the category score, brings the best value in retrieving the relevant entities for the INEX Wikipedia document collection. However, the results also show that using only the linkrank module and the context of the full page results in a very poor entity ranking strategy, which is why below we also experiment with narrow contexts.

5.2 Static and dynamic contexts

We now investigate whether using smaller and more narrow contexts has a positive impact on the effectiveness of entity ranking. Table 7(Q and QE) show the results of this investigation. These results reflect the case when only the linkrank module (α1.0–β0.0) is used by our entity ranking approach.

As in the case with using the full page context, for all the four runs we observe a consistent performance improvement when names of the entity examples are added to the query sent to Zettair. Importantly, when compared to the baseline (the full page context), we observe a substantial increase in performance for the three runs that use smaller and more narrow contexts, irrespective of the type of query used. These increases in performance are all statistically significant ($p < 0.05$). However, the type of query sent to Zettair (Q or QE) seems to have an impact on the best performance that could be achieved by these three runs. Specifically, with MAP the `StatL` run performs best among the three runs when only the topic title is used as an input query (Q), while the `StatR` run is best when using terms from the topic title and the names of entity examples (QE). In both cases the `DynCRE` run achieves the best early precision but overall performs worst among the three runs, although the differences in performance between each of the three run pairs are not statistically significant.

Implementing narrow contexts in our linkrank module allows for the locality of links to be exploited in entity ranking. By changing the context around entity examples, we would also expect the optimal values for the two combining parameters to change. We therefore varied the values for α and β and re-calculated MAP over the 28 topics in our training collection. For the three runs using narrow contexts we found that the optimal value for α has shifted from 0.1 to 0.2 (in the case of Q), while for the two static runs the optimal α value was 0.3 (in the case of QE). In both cases, the optimal value for β was found to be 0.6. The performances of the three optimal runs were very similar, and all of them substantially outperformed the optimal run using the full page context.

Table 7 Performance scores for runs using different types of contexts in the linkrank module (α1.0–β0.0), obtained by different evaluation measures

Run	P[r]		R-prec	MAP
	5	10		
(Q) Topic title				
`FullPage`	0.1571	0.1571	0.1385	0.1314
`StatL`	0.2143	**0.2250**	**0.2285**	**0.1902**
`StatR`	**0.2214**	0.2143	0.2191	0.1853
`DynCRE`	**0.2214**	0.2107	0.2152	0.1828
(QE) Topic title and entity examples				
`FullPage`	0.1857	0.1750	0.1587	0.1520
`StatL`	0.2429	0.2179	**0.2256**	0.2033
`StatR`	0.2429	**0.2214**	0.2248	**0.2042**
`DynCRE`	**0.2571**	0.2107	0.2207	0.1938

Queries sent to Zettair include only terms from the topic title (Q), or terms from the topic title and the names of entity examples (QE). For each measure, the best performing score is shown in bold

6 System evaluation using the INEX 2007 testing data set

In this section, we present results that investigate the effectiveness of our entity ranking approach when using the INEX 2007 XER testing data set. With these experiments, we aim at confirming the impact of using various category and linkrank similarity techniques on the entity ranking performance; we also compare the run performances with a full-text retrieval run as a baseline.

6.1 Runs description

Table 8 lists the six XER runs we evaluate on the INEX 2007 XER testing data set. With the exception of the plain Zettair run, all the runs were created by using our entity ranking system. However, as seen in the table the runs use various parameters whose values are mainly dependent on the task. Specifically, runs differ depending on whether (or which) Zettair category index is used, which of the two types of link contexts is used, whether categories or example entities are used from the topic, and which combination of values is assigned to the α and β parameters.

For example, the run "run 3" for XER task 1 can be interpreted as follows: the Zettair index of category names is used to extract the top five ranked categories, using both the title and the category names (TC) from the INEX topic as a query. This set of five categories is used as an input in the category similarity function ($\texttt{TCcat}(C)$). The full page context (FC) is used to calculate the scores in the linkrank module. The final scores for answer entities are calculated by combining the scores coming out of the three modules ($\alpha = 0.1$, $\beta = 0.8$).

6.2 INEX 2007 Testing data set

The INEX 2007 testing data set comprises two subsets: a subset of topics based on a selection of topics from the INEX 2007 ad hoc track, and a subset of topics specifically developed by participants for the purposes of the XER track. The complete testing data set

Table 8 List of six XER runs submitted for evaluation

Run ID	cat-sim	α	β	Category index			Ctx	Topic	
				Query	Type	M		Cat	Ent
Zettair		–	–	–	–	–	–	–	–
XER task 1									
run 1	$\texttt{cat}(C)$-$\texttt{cat}(t)$	0.0	1.0	–	–	–	FC	Yes	No
run 2	$\texttt{TCcat}(C)$-$\texttt{cat}(t)$	0.0	1.0	TC	C	5	FC	Yes	No
run 3	$\texttt{TCcat}(C)$-$\texttt{cat}(t)$	0.1	0.8	TC	C	5	FC	Yes	No
XER task 2									
run 1	$\texttt{cat}(E)$-$\texttt{cat}(t)$	1.0	0.0	–	–	–	EC	No	Yes
run 2	$\texttt{cat}(E)$-$\texttt{cat}(t)$	0.0	1.0	–	–	–	EC	No	Yes
run 3	$\texttt{cat}(E)$-$\texttt{cat}(t)$	0.2	0.6	–	–	–	EC	No	Yes

"Cat-sim" stands for category similarity, "Ctx" for context, "Cat" for categories, "Ent" for entities, "T" for title, "TC" for title and category names, "C" for category names, "CE" for category and entity names, "FC" for full page context, and "EC" for element context

results in total of 46 topics with corresponding relevance assessments. For task 1 all the relevant entities in the relevance assessments are used to generate the scores, while for task 2 we remove the entity examples both from the list of returned answers and from the relevance assessments, as the task is to find entities other than the provided examples.

6.3 Task 1: entity ranking

Table 9 shows the performance scores on both the training and the testing data sets for task 1, obtained for Zettair and our three XER runs. Runs 1 and 2 use scores coming out from the category module only ($\alpha = 0.0$, $\beta = 1.0$) while run 3 uses a combination of linkrank, category, and Z scores ($\alpha = 0.1$, $\beta = 0.8$). Runs 2 and 3 use lexical similarity for extending the set of target categories.

When comparing the performances of runs that use the category module only (runs 1 and 2) on the testing data set, we observe that run 2 that uses lexical similarity between category names ($\texttt{TCcat}(C)$) is again more effective than the run that uses the topic-provided target categories ($\texttt{cat}(C)$). With MAP, the difference in performance between the two runs is statistically significant ($p < 0.05$). We also observe that the third run, which uses combined scores from the three modules, performs the best among the three. This is a statistically significant 19% relative performance improvement over the best score achieved by using only the category module ($\alpha 0.0$–$\beta 1.0$).

From Table 9 we also observe that, irrespective of the data set used, the three entity ranking runs outperform the plain Zettair run. This suggests that using full-text retrieval alone is not an effective retrieval strategy for this task. The differences in performance between each of the three runs and Zettair are statistically significant ($p < 0.05$) only for the two entity ranking runs that use lexical similarity between category names (runs 2 and 3 in Table 9).

When comparing the MAP scores obtained for runs submitted by all XER track participants, our run 3 was ranked as the third best performing run among the 20 submitted runs for INEX 2007 XER task 1.

Table 9 Performance scores for Zettair and our three XER runs on the training data set (28 topics) and the testing data set (46 topics), obtained with different evaluation measures for INEX 2007 XER task 1

Run ID	cat-sim	α	β	P[r]		R-prec	MAP
				5	10		
Training data set							
Zettair		–	–	0.229	0.232	0.208	0.172
run 1	$\texttt{cat}(C)$-$\texttt{cat}(t)$	0.0	1.0	0.229	0.250	0.215	0.196
run 2	$\texttt{TCcat}(C)$-$\texttt{cat}(t)$	0.0	1.0	0.307	0.318	0.263	0.242
run 3	$\texttt{TCcat}(C)$-$\texttt{cat}(t)$	0.1	0.8	**0.379**	**0.361**	**0.338**	**0.287**
Testing data set							
Zettair		–	–	0.230	0.211	0.208	0.186
run 1	$\texttt{cat}(C)$-$\texttt{cat}(t)$	0.0	1.0	0.283	0.243	0.235	0.199
run 2	$\texttt{TCcat}(C)$-$\texttt{cat}(t)$	0.0	1.0	0.322	0.296	0.300	0.243
run 3	$\texttt{TCcat}(C)$-$\texttt{cat}(t)$	0.1	0.8	**0.378**	**0.339**	**0.346**	**0.294**

For each data set, the best performing score under each measure is shown in bold

6.4 Task 2: list completion

Table 10 shows the performance scores on both the training and testing data sets for task 2, obtained for Zettair and our three XER runs. The three runs use the `StatL` context algorithm in the linkrank module. With the first two runs, we want to compare two entity ranking approaches: the first that uses scores from the linkrank module only (run 1), and the second that uses scores from the category module only (run 2). We observe that using categories is substantially more effective than using the linkrank scores. With MAP, the difference in performance between the two runs is statistically significant ($p < 0.05$) on both data sets.

Run 3 combines the scores from the three modules. To find the optimal values for the two combining parameters for this run, we used the training data set and varied the values for parameters α and β. We found that the highest MAP score (0.377) was achieved for $\alpha = 0.2$ and $\beta = 0.6$. This is a statistically significant 19% relative performance improvement over the best score achieved by using only the category module. From Table 10 we see that the same performance behaviour among the three XER runs is also observed on both (training and testing) data sets.

When the three XER runs are compared with the plain Zettair run, we observe a slightly different performance behaviour depending on the data set used. Specifically, on the training data set the three XER runs outperform the plain Zettair run, while on the testing data set only runs 2 and 3 outperform Zettair which in turn outperforms run 1 (the run that uses linkrank scores only). A more detailed per-topic analysis of this behaviour revealed that this is a result of the different "nature" of the two subsets used in the testing data set. Specifically, Zettair outperformed run 1 only on the 21 topics comprising the ad hoc testing topic subset, while run 1 outperformed Zettair on the 25 topics comprising the testing topic subset developed by the XER participants. This indicates that the ad hoc topic subset may need to be further revised and adapted if it is to be reliably used for XER-specific retrieval tasks.

When comparing the MAP scores obtained for runs submitted by all XER track participants, our run 3 was ranked as the best performing run among the 10 submitted runs for INEX 2007 XER task 2.

Table 10 Performance scores for Zettair and our three XER runs on the training data set (28 topics) and testing data set (46 topics), obtained with different evaluation measures for INEX 2007 XER task 2

Run ID	cat-sim	α	β	P[r]		R-prec	MAP
				5	10		
Training data set							
Zettair	–	–	–	0.229	0.232	0.208	0.172
run 1	cat(E)-cat(t)	1.0	0.0	0.214	0.225	0.229	0.190
run 2	cat(E)-cat(t)	0.0	1.0	0.371	0.325	0.319	0.318
run 3	cat(E)-cat(t)	0.2	0.6	**0.500**	**0.404**	**0.397**	**0.377**
Testing data set							
Zettair	–	–	–	0.183	0.170	0.173	0.155
run 1	cat(E)-cat(t)	1.0	0.0	0.157	0.150	0.163	0.141
run 2	cat(E)-cat(t)	0.0	1.0	0.370	0.298	0.292	0.263
run 3	cat(E)-cat(t)	0.2	0.6	**0.409**	**0.330**	**0.336**	**0.309**

For each data set, the best performing score under each measure is shown in bold

7 Topic difficulty prediction

In this section, we incorporate a method for query (topic) difficulty prediction in our XML entity ranking system. Our approach is based on the generation of a topic classifier that can classify INEX XER topics into two or four classes from a number of features extracted from the topics themselves (also called static or a-priori features) and possibly from a number of other features calculated at run time (also called dynamic or a-posteriori features). We use the open source data mining software Weka (Witten and Frank 2005) developed by the University of Waikato. Weka is a collection of machine learning algorithms for data mining tasks that, given a training set of topics, can generate a classifier from the topics and their associated features. The main goal is to apply the topic difficulty prediction to improve the effectiveness of our entity ranking system without using prediction.

7.1 Identifying classes of topics

The classification of topics into groups is based on how well the INEX participant systems answered to the topics. For each topic, we calculate the topic difficulty using the Average Average Precision (AAP) measure (Mizzaro and Robertson 2007). AAP is the average AP of all the system scores for a given topic: the higher the AAP, the easier the topic.

We define two methods for grouping topics into classes depending on the number of groups we want to build, either two or four classes to experiment with two different types of classes. For grouping the topics into two classes (Easy and Difficult), we use the mean AAP metric as a splitting condition: if AAP for a given topic is superior to the mean AAP (calculated across all topics) then the topic is classified as Easy otherwise it is classified as Difficult. For grouping the topics into four classes (Easy, Moderately_Easy, Moderately_Difficult, and Difficult), we use the mean AAP and the standard deviation around the mean as a splitting condition:

if AAP >= (mean AAP + stDev) **then** Easy topic
if AAP >= (mean AAP) **and** AAP < (mean AAP + stDev) **then** Moderately_Easy topic
if AAP >= (mean AAP − stDev) **and** AAP < (mean AAP) **then** Moderately_Difficult topic
if AAP < (mean AAP − stDev) **then** Difficult topic

The above two or four classes of INEX XER topics are then used as a basis for evaluating our automatic feature-based topic classification algorithm.

7.2 Topic features

From the specific structure of the INEX 2007 XER topics we developed 32 different a-priori features. We call these *a-priori* (or static) features because each of them can be calculated by using only the topic definition before any processing is made by our system. The features include the number of words (excluding stop-words) found in the topic title, the topic description and the topic narrative, respectively; the number of verbs and sentences found in the description or narrative part, as well as the number of words in the union or intersection of these parts of the topic. Additional features are built using the ratio between previous features, for example the ratio between the number of words in the title and the description, or the ratio between the number of sentences in the description and the narrative. The idea is that if the narrative needs more explanation than the title or the

description, it may be because good answers could be difficult to identify among many irrelevant pages. Counting verbs required some natural language processing. We used the NLTK toolkit software (Loper and Bird 2002) which especially helped with the different features concerning verb forms.

Other a-priori features are related to the target categories and the entity examples listed in the topic. The target categories can be either very broad or very specific and they represent indications about the desired type of answer entities, not hard constraints. There could be a correlation between the number of target categories and the topic performance that we wanted to identify. Other features involve not just the topic description but also the INEX Wikipedia test collection, for example, the number of Wikipedia pages attached to the target categories. We also count the number of different Wikipedia categories attached to the entity examples in the topic. Finally we create features that represent the union or intersection of target categories and categories attached to the entity examples.

We also defined a few *a-posteriori* (dynamic) features that could be calculated at run time, that is when sending the topic (query) to our system. These features include the number of links from the highly ranked pages returned by the search engine, the number of contexts identified in those pages and the number of common categories attached to the entity examples and the answer entities.

7.3 Topic classifier

The next step was to identify among the 32 features those that best correlated with the topic difficulty, that is the features that would be usable by a classifier to predict between different classes of topics. We first generated many classifiers, each one associated with a random subset of the 46 INEX 2007 XER topics. The classifiers were generated using the Weka j48 classifier based on the well known Quinlan's C4.5 statistical classifier (Quinlan 1993), with each training topic subset classified using the topic classification. We then manually analysed all the decision trees generated by Weka to identify the features that were actually used by the generated classifiers. As a result, we could extract a small subset of nine features that correlated well with topic difficulty.

Table 11 shows the nine features used to generate the training topic subsets. We discovered that the dynamic (a-posteriori) features had no influence on the generated classifiers, and so we only used the a-priori features. However, in the future we plan to perform a comprehensive leave-one-out analysis to study the importance of each feature in greater detail.

7.4 Training and testing topic sets

The INEX XER track does not offer a large test collection, however, this is currently the only test collection available for the purposes of our topic difficulty classification. We focus on the list completion task (task 2), corresponding to a training data set comprising a total number of 46 topics and 17 runs (from INEX 2007 XER), and a testing data set comprising a total number of 35 topics and 16 runs (from INEX 2008 XER).

For each training subset of INEX 2007 XER topics that we used previously for generating a classifier, we used the testing set comprising all the 35 INEX 2008 XER topics for evaluating the performance of this classifier. We tried many different mostly random combinations of training topic subsets, but because of their relatively small sizes on average the accuracy of the correctly classified instances was around 71%.

Table 11 Nine topic features that correlated well with the topic difficulty prediction

Number	Description	Features
1	Topic definition features	#sent_narr
2–6	Ratio of topic definition features	#w_title/#w_narr, #w_intersec(title,narr)/ #w_union(title,narr)
		#w_intersec(desc,narr)/#w_union(desc,narr),
		#w_intersec(title,desc)/#w_union(title,desc,narr),
		#w_intersec(desc,narr)/#w_union(title,desc,narr)
7–8	Topic definition and Wikipedia features	#pages_per_t_cat, #intersec(e_cat)
9	Ratio of topic definition and Wikipedia features	#intersec(e_cat)/#union(e_cat)

In the table, w stands for words, `narr` for narrative, `desc` for description, `t_cat` for target categories, and `e_cat` for categories attached to entity examples. For example, `#sent_narr` is the number of sentences in the narrative part of the INEX topic. The features are sorted by their descriptions, not by their predictive power

To improve the accuracy, we used a well known approach that combines several decision trees, each generated from slightly different topic sets. This is known as Random Forests (Breiman 2001) and was used in query prediction by Yom-Tov et al. (2004). Before implementing the combined classifier, we carefully built the training topic set for each individual predictor so that the included topics were representative of different features.

We manually divided the training set of 46 INEX 2007 XER topics into four subsets of around 35 topics each. We had to do it manually in order to get as much different and representative topics as possible, especially because of the small topic subset sizes. So those four subsets and the one with all 46 topics made five different training sets from which we built five separate classifiers. For these and the final combined classifier we also had to build a testing topic set that does not include any of the training topics. The INEX 2008 XER topic set was used for this purpose.

7.5 Validation of topic difficulty prediction

The final topic difficulty prediction classifier was built using a simple voting system (which is the reason why we needed an odd number of classifiers). For building a two-class classifier the voting algorithm is trivial: for a topic we get a prediction from the five classifiers and count the number of predictions as Easy and the number of predictions as Difficult; the majority gives the prediction for the final classifier. For example, if the predictions from the five classifiers are [diff, easy, easy, diff, diff], the combined prediction is Difficult.

The combined two-class classifier resulted in a precision of 74% on our testing set which is better than what we could achieve with a single classifier. Table 12 shows the accuracy achieved by each of the six classifiers.

We also considered the possibility of building a four-class classifier (Easy, Moderately_Easy, Moderately_Difficult, and Difficult). A similar voting algorithm is used by simply choosing diff : easy = 0:5 and diff : easy = 5:0 to be predicted as Easy and Difficult topics, respectively, with the diff : easy = (1:4 | 2:3) and diff : easy = (4:1 | 3:2) combinations resulting in Moderately_Easy and Moderately_Difficult topics, respectively. The combined four-class classifier resulted in an accuracy of 31% on our testing set which was much less than that achieved by the two-class classifier.

Table 12 Accuracy achieved by the six two-class classifiers on the 35 INEX 2008 XER topics

Classifier	Class 2 Correct
1	24/35 (68%)
2	25/35 (71%)
3	25/35 (71%)
4	25/35 (71%)
5	24/35 (68%)
Combined	26/35 (74%)

7.6 Applying topic difficulty prediction in entity ranking

Our objective with topic difficulty prediction was to improve the performance of our XER system by dynamically tuning the values for system parameters according to the predicted topic class. Specifically, for each of the 35 INEX 2008 XER topics in the testing set, we adapt the values for the α and β system parameters in accordance with the estimated optimal values observed on the training set. We focus on the list completion task, where a selected number of entity examples are provided in the topic.

7.6.1 Choosing optimal system parameters by topic difficulty

We first estimated the optimal values for the system parameters by using the 46 INEX 2007 XER training topics and by also taking into account their corresponding topic difficulty classes. We generated all the possible 66 runs by respectively varying the values of α and β from 0 to 1 by increment of 0.1. For each topic, we then measured the average precision (AP) for each run and ordered the runs by decreasing value of AP. This way we could identify the values of the two (α, β) parameters that performed *best* for each individual topic. To estimate which (α, β) pair would be *optimal* for a given topic difficulty class, we used the topics that belong to a particular class (such as Easy), and calculated the mean AP (MAP) for each run that appeared at least once among the ten highly ranked runs for a given topic.[6] We then ordered the runs by decreasing scores and identified the highest ranked run as the optimal (α, β) pair for each topic difficulty class. We did this both for the two and the four classes of topic difficulty.

Table 13 shows the estimated optimal values for (α, β) as measured by MAP, when using two or four classes of topic difficulty. Interestingly, with the four-class prediction the optimal parameter values for the Easy topics are ($\alpha = 0.0$, $\beta = 0.7$), that is the link score is ignored. For the Difficult topics the opposite effect is observed with the category score ignored and a high weight spread evenly on the link score α and the Z score ($1 - \alpha - \beta$).

7.6.2 Evaluation of the predicted topic performance

We now use our combined topic difficulty prediction algorithm to tune and evaluate the performance of our XER system on the 35 INEX 2008 XER testing topics. According to

[6] The value ten was determined experimentally on the INEX 2007 XER training topic set.

Table 13 Estimated values for optimal (α, β) system parameters, as measured by MAP using the 46 topics of the INEX 2007 XER training topic set

Measure	Class											
	2				4							
	Easy		Diff		Easy		modEasy		modDiff		Diff	
	α	β	α	β	α	β	α	β	α	β	α	β
MAP	0.2	0.6	0.1	0.8	0.0	0.7	0.2	0.6	0.1	0.8	0.5	0.0

the estimated prediction, we aim at dynamically setting the α and β parameters to their optimal values shown in Table 13. We did two sets of experiments, respectively with two and four classes of topic difficulty. We use MAP as our choice of evaluation measure.

To evaluate the benefits of using topic difficulty prediction, we compare the performances of four different runs:

1. *Baseline* run that does not use topic difficulty prediction with parameter vales set to ($\alpha = 0.2$, $\beta = 0.6$), representing the `cat`(E)-`cat`(t) run using the `StatL` context algorithm in the linkrank module. This was the best performing entity ranking run at INEX 2007 (for task 2: list completion) when using the MAP measure (see Sect. 6.4).
2. *Predicted* run with parameter values set according to the estimated topic difficulty prediction on the training collection. The difficulty of a particular INEX 2008 XER testing topic was first predicted by our topic prediction algorithm, and the system parameter values were then set to the estimated optimal values for that topic difficulty class (as shown in Table 13).
3. *Optimal* run with parameter values set to the estimated optimal values on the training collection. Given the previously determined difficulty of a particular INEX 2008 XER testing topic (by applying the AAP measure on all the INEX 2008 submitted runs), the system parameter values were set to the estimated optimal values for that topic difficulty class. This is the best run we could aim at by using our topic difficulty prediction algorithm.
4. *Perfect* run with parameter values set to the best values (out of all the 66 value combinations) that can be achieved for each topic on the INEX 2008 XER testing collection. This is the run that produces the absolute best performance with our current XER system.

The results are presented in Table 14. The table shows that a two-class prediction of topic difficulty is performing better than the baseline (our last year best run), although the difference in performance is not statistically significant. These two runs were among the top four best performing runs at INEX 2008, all of which were submitted by our participating group. On the other hand, the four-class prediction of topic difficulty resulted in decreased performance compared to the baseline, with the optimal four-class prediction run performing worse than the optimal two-class prediction run. This is mainly due to the fact that the topic prediction algorithm is specifically designed for two-class rather than for four-class prediction. Although the results are promising, we recognise that the small sizes of the training and testing topic sets do not allow for very conclusive evaluation.

Table 14 Evaluation of the predicted topic performance, as measured by MAP using the 35 topics of the INEX 2008 XER testing collection

Run	Class											
	2					4						
	Easy		Diff			Easy		modEasy		modDiff		Diff	
	0.2	0.6	0.1	0.8		0.0	0.7	0.2	0.7	0.1	0.8	0.6	0.0
Baseline	N/A				0.36280	N/A							
Predicted	0.38085					0.30769							
Optimal	0.38705					0.38431							
Perfect	N/A				0.45746†	N/A							

The † symbol shows statistical significance over the Baseline run ($p < 0.05$)

8 Related work

Our entity ranking approach gets its inspiration from wrapping technology, entity extraction and disambiguation, the use of ontologies for entity extraction or entity disambiguation, link analysis and query difficulty prediction.

8.1 Wrappers

A wrapper is a tool that extracts information (entities or values) from a document, or a set of documents, with a purpose of reusing information in another system. A lot of research has been carried out in this field by the database community, mostly in relation to querying heterogeneous databases (Vercoustre and Paradis 1997; Sahuguet and Azavant 1999; Adelberg and Denny 1999; Kushmerick 2000). More recently, wrappers have also been built to extract information from web pages with different applications in mind, such as product comparison, reuse of information in virtual documents, or building experimental data sets. Most web wrappers are either based on scripting languages (Vercoustre and Paradis 1997; Sahuguet and Azavant 1999) that are very close to current XML query languages, or use wrapper induction (Adelberg and Denny 1999; Kushmerick 2000) that learn rules for extracting information.

Wrapper scripting languages (Vercoustre and Paradis 1997; Sahuguet and Azavant 1999) are very close to current XML query languages: they allow for selecting information using XML paths, or more advanced queries, in semi-structured documents (XML-like) and reconstructing results into a new XML document. Such wrappers are easy to write for sets of regular pages, but would break easily even on small changes in the structure of the pages. Inductive wrappers (Adelberg and Denny 1999; Kushmerick 2000) are built using examples from which the wrapper learns the rules for extracting the information and maps it into the appropriate data. Those wrappers need to also be retrained if the pages change.

To prevent wrappers breaking over time without notice when pages change, Lerman et al. (2003) propose using machine learning for wrapper verification and re-induction. Rather than repairing a wrapper over changes in the web data Callan and Mitamura (2002), propose generating the wrapper dynamically—that is at the time of wrapping, using data previously extracted and stored in a database. The extraction rules are based on heuristics around a few pre-defined lexico-syntactic HTML patterns such as lists, tables, and links.

The patterns are weighted according to the number of examples they recognise; the best patterns are used to dynamically extract new data.

Our system for entity ranking also works dynamically, at query time instead of at wrapping time. We also use weighting algorithms based on links that are well represented in web-based collections, as well as utilising categories, a specific Wikipedia feature.

8.2 Entity extraction and disambiguation

Kazama and Torisawa (2007) explore the use of Wikipedia as external knowledge to improve named entity recognition, by using the first sentence of a Wikipedia page to infer the category of the entity attached to that page. These categories are then used as features in their named entity tagger. We do not use inferred categories in our approach; instead, we use categories that were explicitly attached to the entity page by Wikipedia authors. Cucerzan (2007) also uses Wikipedia for entity disambiguation by exploiting (amongst other features) co-references in static contexts such as titles, links, paragraphs and lists. Callan and Mitamura (2002) investigate if the entity extraction rules can be dynamically generated. Their rules are based on heuristics exploiting a few pre-defined HTML contexts such as lists and tables. The contexts are weighted according to the number of contained examples; the best contexts are then used to dynamically extract new data. We use pre-defined contexts in our entity ranking approach; however, we also develop a new algorithm that dynamically determines the contexts around entity examples.

ESTER (Bast et al. 2007) was recently proposed as a system for searching text, entities and relations. ESTER relies on the Wikipedia links to identify the entities and on the context of the links for disambiguation (using 20 words around the anchor text instead of just the anchor text). The system also uses the Wikipedia categories to build a "semantic index" for each entity; for example, the entity "John Lennon" will have an index entry musician:john-lennon, corresponding to the category musician to which the entity belongs. This approach primarily focuses on improving the efficiency of the proposed system, while we are more interested in improving the effectiveness of entity ranking. Hu et al. (2006) propose a linear model that uses a number of features to weight passages containing entity names. They first determine top k passages and extract the top n entities from these passages. Features include term frequency, distance to the entity name and co-occurrences in the same section as the entity. Tsikrika et al. (2008) build a graph of the initial set of documents returned in answer to the query, and for each document use up to k steps to extend the graph with entities from linked documents outside the initial set; this graph is then used to propagate relevance in entity retrieval based on k-step or infinite random walk. Entities are also filtered by using a target category and its descendant categories (up to a third level). We currently use a one-step relevance propagation to retrieve relevant entities in our system, and plan to extend this with a k-step random walk in the future.

Cucerzan and Yarowsky (1999) describe and evaluate a language-independent bootstrapping algorithm based on iterative learning and re-estimation of contextual and morphological patterns. It achieves competitive performance when trained on a very short labelled name list.

8.3 Using ontology for entity extraction and disambiguation

Since Wikipedia has some but not all characteristics associated with an ontology, one could think of adapting some of the similarity measures proposed for comparing concepts in an ontology and use those for comparing categories in Wikipedia. Ehrig et al. (2005)

and Blanchard et al. (2005) have surveyed various such similarity measures. These measures are mostly reflexive and symmetric (Ehrig et al. 2005) and take into account the distance (in the path) between the concepts, the depth from the root of the ontology and the common ancestor of the concepts, and the density of concepts on the paths between the concepts and from the root of the ontology (Blanchard et al. 2005).

All these measures rely on a strong hierarchy of the ontology concepts and a subsumption hypothesis in the parent–child relationship. Since those hypothesis are nor verified in Wikipedia (see Sect. 2), we could not use those similarity functions. Instead we experimented with similarities between sets of categories and lexical similarities between category names.

Hassell et al. (2006) use a "populated ontology" to assist in disambiguation of entities, for example names of authors using their published papers or domain of interest. They use text proximity between entities to disambiguate names (e.g. organisation name would be close to author's name), text co-occurrence (for example for topics relevant to an author), and the notion of "popular entity" associated to a given relationship (for example, an author is popular if it has many "authored" links). They also use semantic relationships such as co-authoring. So their algorithm is tuned for their actual ontology, while our algorithm is more based on the structural properties of the Wikipedia.

Cucerzan (2007) uses Wikipedia data for named entity disambiguation. They first preprocessed a version of the Wikipedia collection (September 2006), and extracted more than 1.4 million entities with an average of 2.4 surface forms by entities. They also extracted more than 1 million (entities, category) pairs that were further filtered out to 540 thousand pairs. Lexico-syntactic patterns, such as titles, links, paragraphs and lists, are used to build co-references of entities in limited contexts. However, the overwhelming number of contexts that could be extracted this way requires the use of heuristics to limit the context extraction. The knowledge extracted from Wikipedia is then used for improving entity disambiguation in the context of web and news search.

8.4 Link analysis

Most information retrieval systems use statistical information concerning the distribution of the query terms to calculate the query-document similarity. However, when dealing with hyperlinked environments such as the web or Wikipedia, link analysis is also important. PageRank and HITS are two of the most popular algorithms that use link analysis to improve web search performance.

PageRank, an algorithm proposed by Brin and Page (1998), is a link analysis algorithm that assigns a numerical weighting to each page of a hyperlinked set of web pages. The idea of PageRank is that a web page is a good page if it is popular, that is if many other (also preferably popular) web pages are pointing to it.

In HITS (Hyperlink Induced Topic Search), *hubs* are considered to be web pages that have links pointing to many *authority* pages (Kleinberg 1999). However, unlike PageRank where the page scores are calculated independently of the query by using the complete web graph, in HITS the calculation of hub and authority scores is query-dependent; here, the so-called *neighbourhood graph* includes not only the set of top-ranked pages for the query, but it also includes the set of pages that either point to or are pointed to by these pages.

We use the idea behind PageRank and HITS in our system; however, instead of counting every possible link referring to an entity page in the collection (as with PageRank), or building a neighbourhood graph (as with HITS), we only consider pages that are pointed to by a selected number of top-ranked pages for the query. This makes our link

ranking algorithm query-dependent (just like HITS), allowing it to be dynamically calculated at query time.

Cai et al. (2004) recognise that most popular linkrank algorithms treat a web page as a single node, despite the fact that the page may contain multiple semantic blocks. Using the visual presentation of a page to extract the semantic structure, they adapted PageRank and HITS to deal with block nodes rather than full page nodes. Nie et al. (2006) propose a topical link analysis model that formalises the idea of splitting the credit (the authority score) of a source page into different topics based on topical distribution. Our entity ranking approach is based on a similar idea, except that instead of using topics for discrimination we use list-like contexts around the entity examples.

Fissaha Adafre et al. (2007) form entity neighbourhoods for every entity, which are based on clustering of similar Wikipedia pages using a combination of extracts from text content and following both incoming and outgoing page links. These entity neighbourhoods are then used as the basis for retrieval for the two entity ranking tasks. Our approach is similar in that it uses XML structural patterns (links) rather than textual ones to identify potential entities. It also relies on the co-location of entity names with some of the entity examples (when provided). However, we also make use of the category hierarchy to better match the result entities with the expected class of the entities to retrieve. This is in line with the recent finding by Kaptein and Kamps (2009), who observed that using the Wikipedia category information resulted in significant improvements on *both* types of entity ranking and ad hoc topics.

8.5 Query difficulty prediction

The approaches to query prediction can generally be grouped into two types: *static prediction* approaches, based on intrinsic characteristics of the query and possibly the document collection (He and Ounis 2006); and *dynamic prediction* approaches, which use characteristics of the top ranked answers to the query (Zhou and Croft 2007).

Lang et al. (2008) evaluate query performance based on the covering topic score that measures how well the topic of the query is covered by documents retrieved by the system (dynamic prediction). Cronen-Townsend et al. (2002) propose to predict query performance by computing the relative entropy (clarity score) between a query language model and the corresponding collection language model (static prediction).

Mothe and Tanguy (2005) predict query difficulty based on linguistic features, using TreeTagger for part-of-speech tagging and other natural language processing tools. Topic features include morphological features (number of words, average of proper nouns, average number of numeral values), syntactical features (average conjunctions and prepositions, average syntactic depth and link span) or semantic features (average polysemy value). They found that the only positively correlated feature is the number of proper nouns, although the average syntactic link span and the average polysemy value also have some correlation with topic difficulty. We use some morphological or syntactic features in our topic prediction algorithm, but we also take advantage of the structure of the topic (title, description, narrative).

Kwok (2005) uses Support Vector Machine (SVM) regression to predict the weakest and strongest queries in the TREC 2004 topic set (static prediction). Their choice of features include inverse document frequency of query terms and average term frequency in the collection. They found that features based on term frequencies could predict correctly even with short queries.

Yom-Tov et al. (2004) predict query difficulty by measuring the contribution of closely related query terms, using features such as the overlap between the k-top answers to a sub-query (a query based on one query term) and to the full query. They experimented with two different query predictors: an histogram-based predictor and a modified decision tree. The difficulty predictor was used for selective query expansion or reduction.

Grivolla et al. (2005) propose several classifiers to predict easy and difficult queries. They use decision tree and SVM types of classifiers, and select useful features among a set of candidates; the classifiers are trained on the TREC 8 test collection. The features are computed from the query itself (static features), the retrieval results and the knowledge about the retrieval process (dynamic features). They tested many different classifiers but did not combine them. Our approach is very similar, except that we use different topic-specific features, a combined classifier and different test collection (INEX instead of TREC).

9 Conclusions and future work

In this paper, we have presented our system for ranking Wikipedia entities in answer to a query. The system implements an entity ranking approach for the INEX Wikipedia XML document collection that is based on exploiting the interesting structural and semantic properties of the collection. Our experiments have demonstrated that utilising categories and the link structure of Wikipedia can significantly improve entity ranking effectiveness, and that topic difficulty prediction is a promising approach that could also be used to further improve the entity ranking performance.

When utilising Wikipedia categories in our system, we found that using lexical similarity between category names results in an effective entity ranking approach, so long as the category index comprises documents containing only category names. We also found that the best entity ranking approach is the one that uses sets of categories directly attached to both the example and the answer entities, and that using various extensions of these two sets significantly decreases the entity ranking performance. Importantly, when comparing the scores of the best performing approaches across the two entity ranking tasks, we found that the query strategy that uses example entities to identify the set of target categories is significantly more effective than the strategy that uses the set of loosely defined target categories. In the future, we plan to introduce different category weighting rules that we hope would better distinguish the answer entities that are more similar to the entity examples.

When utilising the link structure of Wikipedia in our system, we found that the locality of Wikipedia links can be exploited to significantly improve the effectiveness of entity ranking. Using an approach that takes the broad context of a full Wikipedia page as a baseline, we evaluated two alternative approaches that take narrow contexts around the entity examples: one that uses static (predefined) types of elements such as paragraphs, lists and tables; and another that dynamically identifies the contexts by utilising the underlying XML document structure. Although the entity ranking performances of the two approaches were similar, both of them nevertheless significantly outperformed the approach that takes the broad context of a full Wikipedia page. In the future, we plan to further improve our linkrank algorithm by varying the number of entity examples and incorporating relevance feedback that we expect would reveal other useful entities that could be used to identify better contexts.

When utilising topic difficulty prediction in our system, we found that it is possible to predict accurately a two-class level of topic difficulty with a classifier generated from a

selected number of static features extracted from the INEX topic definition and the Wikipedia document collection. Interestingly, when analysing the impact of four classes of topic difficulty on the optimal parameter values of our system, we found that for the Easy topics the use of Wikipedia categories is very important while for the Difficult topics the link structure plays an important role. The application of topic prediction in tuning our system has shown encouraging improvement over the approach that does not use prediction, but we need a larger test collection to confirm the significance of our findings. The major limitation of our topic prediction approach is that it relies on the INEX topic definition that is much richer than standard Web queries (and so hardly applicable in practice). In the future we plan to develop a dynamic query prediction approach based solely on the query terms and (among other things) on the similarity scores of the relevant entities retrieved by our XER system.

Our entity ranking system was evaluated as one of the best performing systems when compared with other participating systems in both the INEX 2007 and INEX 2008 XER tracks. In the future, we aim at further developing our entity ranking algorithms by incorporating natural language processing techniques that we expect would reveal more potentially relevant entities. We also recognise that the entity ranking techniques presented in this paper have been developed for specific INEX tasks and tested on one (XML) version of the Wikipedia. However, they demonstrate the potential of using Wikipedia pages for assisting in more general focused retrieval tasks performed on even more diverse collections (such as the Web), which we plan to investigate and carry out in the future.

Acknowledgments Much of this work was carried out at INRIA, including while Jovan Pehcevski held a post doctoral position in 2007, James Thom was a visiting scientist in 2007, and Vladimir Naumovski had an internship in 2008.

References

Adelberg, B., & Denny, M. (1999). Nodose version 2.0. In *Proceedings of the 1999 ACM SIGMOD international conference on management of data (SIGMOD'99)*, Philadelphia, Pennsylvania, pp. 559–561.

Awang Iskandar, D., Pehcevski, J., Thom, J. A., & Tahaghoghi, S. M. M. (2007). Social media retrieval using image features and structured text. In *Comparative evaluation of XML information retrieval systems: Fifth workshop of the INitiative for the evaluation of XML retrieval, INEX 2006*, Lecture notes in computer science, Vol. 4518, pp. 358–372.

Bast, H., Chitea, A., Suchanek, F., & Weber, I. (2007). ESTER: Efficient search on text, entities, and relations. In *Proceedings of the 30th ACM international conference on research and development in information retrieval (SIGIR'07)*, Amsterdam, The Netherlands, pp. 671–678.

Blanchard, E., Harzallah, M., & Henri Briand, P. K. (2005). A typology of ontology-based semantic measures. In *Proceedings of the open interop workshop on enterprise modelling and ontologies for interoperability (EMOI-INTEROP'05)*, Porto, Portugal. http://www.sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-160/paper26.pdf.

Breiman, L. (2001). Random forests. *Machine Learning 45*(1), 5–32

Brin, S., & Page, L. (1998). The anatomy of a large-scale hypertextual Web search engine. In *Proceedings of the 7th international conference on world wide web*, Brisbane, Australia, pp. 107–117.

Cai, D., He, X., Wen, J. R., & Ma, W. Y. (2004). Block-level link analysis. In *Proceedings of the 27th ACM international conference on research and development in information retrieval (SIGIR'04)*, Sheffield, UK, pp. 440–447.

Callan, J., & Mitamura, T. (2002). Knowledge-based extraction of named entities. In *Proceedings of the 11th ACM conference on information and knowledge management (CIKM'02)*, McLean, Virginia, pp. 532–537.

Carmel, D., Yom-Tov, E., & Soboroff, I. (2005). Predicting query difficulty—methods and applications. *SIGIR Forum 39*(2), 25–28.

Cronen-Townsend, S., Zhou, Y., & Croft, W. B. (2002). Predicting query performance. In *Proceedings of the 25th ACM SIGIR conference on research and development in information retrieval (SIGIR'02)*, Tampere, Finland, pp. 299–306.

Cucerzan, S. (2007). Large-scale named entity disambiguation based on Wikipedia data. In *Proceedings of the 2007 joint conference on EMNLP and CoNLL*, Prague, The Czech Republic, pp. 708–716.

Cucerzan, S., & Yarowsky, D. (1999). Language independent named entity recognition combining morphological and contextual evidence. In *Proceedings of the 1999 joint SIGDAT conference on EMNLP and VLC*, Maryland, MD, pp. 90–99.

de Vries A. P., Vercoustre A. M., Thom J. A., Craswell N., & Lalmas M. (2008). Overview of the INEX 2007 entity ranking track. In *Focused access to XML documents: Sixth international workshop of the initiative for the evaluation of XML retrieval, INEX 2007*, Lecture notes in computer science, Vol. 4862, pp. 1–23.

Demartini, G., de Vries, A. P., Iofciu, T., & Zhu, J. (2009). Overview of the INEX 2008 entity ranking track. In *Advances in focused retrieval: Seventh international workshop of the initiative for the evaluation of XML retrieval, INEX 2008*, Lecture notes in computer science, Vol. 5631.

Denoyer, L., & Gallinari, P. (2006). The Wikipedia XML corpus. *SIGIR Forum 40*(1), 64–69

Ehrig, M., Haase, P., Stojanovic, N., & Hefke, M. (2005). Similarity for ontologies—a comprehensive framework. In *Proceedings of the 13th European conference on information systems*.

Fissaha Adafre, S., de Rijke, M., & Sang, E. T. K. (2007). Entity retrieval. In *Proceedings of international conference on recent advances in natural language processing (RANLP—2007)*, September 27–29, Borovets, Bulgaria.

Grivolla, J., Jourlin, P., & de Mori, R. (2005). Automatic classification of queries by expected retrieval performance. In *Proceedings of the SIGIR workshop on predicting query difficulty*, Salvador, Brazil.

Hassell, J., Aleman-Meza, B., & Arpinar, I. B. (2006). Ontology-driven automatic entity disambiguation in unstructured text. In *Proceedings of the 5th international semantic web conference (ISWC)*, Athens, GA, Lecture notes in computer science, Vol. 4273, pp. 44–57.

He, B., & Ounis, I. (2006). Query performance prediction. *Information Systems 31*(7), 585–594.

Hu, G., Liu, J., Li, H., Cao, Y., Nie, J. Y., & Gao, J. (2006). A supervised learning approach to entity search. In *Proceedings of the Asia information retrieval symposium (AIRS 2006)*. Lecture notes in computer science, Vol. 4182, pp. 54–66.

Kamps, J., & Larsen, B. (2006). Understanding differences between search requests in XML element retrieval. In *Proceedings of the SIGIR 2006 workshop on XML element retrieval methodology*, Seattle, Washington, pp. 13–19.

Kaptein, R., & Kamps, J. (2009). Finding entities or information using annotations. In *ECIR workshop on information retrieval over social networks*, pp. 71–78.

Kazama, J., & Torisawa, K. (2007). Exploiting Wikipedia as external knowledge for named entity recognition. In *Proceedings of the 2007 joint conference on EMNLP and CoNLL*, Prague, The Czech Republic, pp. 698–707.

Kleinberg, J. M. (1999). Authoritative sources in hyperlinked environment. *Journal of the ACM 46*(5), 604–632.

Kushmerick, N. (2000). Wrapper induction: Efficiency and expressiveness. *Artificial Intelligence 118*(1–2), 15–68.

Kwok, K. (2005). An attempt to identify weakest and strongest queries. In *Proceedings of the SIGIR workshop on predicting query difficulty*, Salvador, Brazil.

Lang, H., Wang, B., Jones, G., Li, J. T., Ding, F., & Liu, Y. X. (2008). Query performance prediction for information retrieval based on covering topic score. *Journal of Computer Science and technology 23*(4), 590–601.

Lerman, K., Minton, S. N., & Knoblock, C. A. (2003). Wrapper maintenance: A machine learning approach. *Journal of Artificial Intelligence Research 18*, 149–181.

Loper, E., & Bird, S. (2002). NLTK: The natural language toolkit. In *Proceedings of the ACL-02 workshop on effective tools and methodologies for teaching natural language processing and computational linguistics*, Philadelphia, Pennsylvania, pp. 63–70.

Mizzaro, S. (2008). The good, the bad, the difficult, and the easy: Something wrong with information retrieval evaluation? In *Proceedings of the 30th European conference on information retrieval (ECIR'08)*, Lecture Notes in Computer Science, Vol. 4956, pp. 642–646.

Mizzaro, S., & Robertson, S. (2007). HITS hits TREC: Exploring IR evaluation results with network analysis. In *Proceedings of the 30th ACM SIGIR conference on research and development in information retrieval (SIGIR'07)*, Amsterdam, The Netherlands, pp. 479–486.

Mothe, J., & Tanguy, L. (2005). Linguistic features to predict query difficulty. In *Proceedings of the SIGIR workshop on predicting query difficulty*, Salvador, Brazil.

Nie, L., Davison, B. D., & Qi, X. (2006). Topical link analysis for web search. In *Proceedings of the 29th ACM international conference on research and development in information retrieval (SIGIR'06)*, Seattle, Washington, pp. 91–98.

Pehcevski, J., Thom, J. A., & Vercoustre, A. M. (2005). Hybrid XML retrieval: Combining information retrieval and a native XML database. *Information Retrieval 8*(4), 571–600.

Pehcevski, J., Vercoustre, A. M., & Thom, J. A. (2008). Exploiting locality of Wikipedia links in entity ranking. In *Proceedings of the 30th European conference on information retrieval (ECIR'08)*, Lecture notes in computer science, Vol. 4956, pp. 258–269.

Quinlan, J. R. (1993). C4.5: Programs for machine learning. Morgan Kaufmann Publishers, Inc.

Sahuguet, A., & Azavant, F. (1999). Building light-weight wrappers for legacy web data-sources using W4F. In *Proceedings of 25th international conference on very large data bases (VLDB'99)*, Edinburgh, Scotland, UK, pp. 738–741.

Soboroff, I., de Vries, A. P., & Craswell, N. (2006). Overview of the TREC 2006 Enterprise track. In *Proceedings of the fifteenth text retrieval conference (TREC 2006)*, pp. 32–51.

Thom, J. A., Pehcevski, J., & Vercoustre, A. M. (2007). Use of Wikipedia categories in entity ranking. In *Proceedings of 12th Australasian document computing symposium (ADCS'07)*, Melbourne, Australia, pp. 56–63.

Tsikrika, T., Serdyukov, P., Rode, H., Westerveld, T., Aly, R., Hiemstra, D., et al. (2008). Structured document retrieval, multimedia retrieval, and entity ranking using PF/Tijah. In *Focused access to XML documents: Sixth international workshop of the initiative for the evaluation of XML retrieval, INEX 2007*, Lecture notes in computer science, Vol. 4862, pp. 306–320.

Vercoustre, A. M., & Paradis, F. (1997). A descriptive language for information object reuse through virtual documents. In *4th International conference on object-oriented information systems (OOIS'97)*, Brisbane, Australia, pp. 299–311.

Vercoustre, A. M., Pehcevski, J., & Thom, J. A. (2008a). Using Wikipedia categories and links in entity ranking. In *Focused access to XML documents: Sixth international workshop of the initiative for the evaluation of XML retrieval, INEX 2007*, Lecture notes in computer science, vol. 4862, pp. 321–335.

Vercoustre, A. M., Thom, J. A., & Pehcevski, J. (2008b). Entity ranking in Wikipedia. In *Proceedings of the 23rd ACM symposium on applied computing*, Fortaleza, Ceará, Brazil, pp. 1101–1106.

Vercoustre, A. M., Pehcevski, J., & Naumovski, V. (2009). Topic difficulty prediction in entity ranking. In *Advances in focused retrieval: Seventh international workshop of the initiative for the evaluation of XML retrieval, INEX 2008*, Lecture notes in computer science, Vol. 5631.

Voorhees, E. M. (2004). The TREC robust retrieval track. In *Proceedings of the thirteenth text retrieval conference (TREC 2004)*.

Webber, W., Moffat, A., & Zobel, J. (2008). Score standardization for inter-collection comparison of retrieval systems. In *Proceedings of the 31st ACM SIGIR conference on research and development in information retrieval (SIGIR'08)*, Singapore, pp. 51–58.

Witten, I. H., & Frank, E. (2005). *Data mining: Practical machine learning tools and techniques*, second edition. Morgan Kaufmann Publishers, Inc.

Yom-Tov, E., Fine, S., Carmel, D., Darlow, A., & Amitay, E. (2004). Juru at TREC 2004: Experiments with prediction of query difficulty. In *Proceedings of the thirteenth text retrieval conference (TREC 2004)*.

Yu, J., Thom, J. A., & Tam, A. (2007). Ontology evaluation using Wikipedia categories for browsing. In *Proceedings of the 16th ACM conference on information and knowledge management (CIKM'07)*, Lisboa, Portugal, pp. 223–232.

Zhou, Y., & Croft, W. B. (2007). Query performance prediction in web search environments. In *Proceedings of the 30th ACM SIGIR conference on research and development in information retrieval (SIGIR'07)*, Amsterdam, The Netherlands, pp. 543–550.