Query Modeling for Entity Search Based on Terms, Categories, and Examples

KRISZTIAN BALOG, Norwegian University of Science and Technology
MARC BRON and MAARTEN DE RIJKE, University of Amsterdam

Users often search for entities instead of documents, and in this setting, are willing to provide extra input, in addition to a series of query terms, such as category information and example entities. We propose a general probabilistic framework for entity search to evaluate and provide insights in the many ways of using these types of input for query modeling. We focus on the use of category information and show the advantage of a category-based representation over a term-based representation, and also demonstrate the effectiveness of category-based expansion using example entities. Our best performing model shows very competitive performance on the INEX-XER entity ranking and list completion tasks.

Categories and Subject Descriptors: H.3 [**Information Storage and Retrieval**]: H.3.1 Content Analysis and Indexing; H.3.3 Information Search and Retrieval; H.3.4 Systems and Software; H.4 [**Information Systems Applications**]: H.4.2 Types of Systems; H.4.m Miscellaneous

General Terms: Algorithms, Measurement, Performance, Experimentation

Additional Key Words and Phrases: Entity retrieval, query modeling, query expansion, generative probabilistic model

ACM Reference Format:
Balog, K., Bron, M., and de Rijke, M. 2011. Query modeling for entity search based on terms, categories, and examples. ACM Trans. Inf. Syst. 29, 4, Article 22 (November 2011), 31 pages.
DOI = 10.1145/2037661.2037667 http://doi.acm.org/10.1145/2037661.2037667

1. INTRODUCTION

Users often search for specific entities such as people, products, or locations instead of documents that merely mention them [de Vries et al. 2008; Mishne and de Rijke 2006]. Examples of information needs include “Countries where one can pay with the euro,” “Impressionist art museums in The Netherlands,” or “Experts on authoring tools,” where answers to be returned are countries, museums, or experts; not just articles discussing them. In such scenarios, users may be assumed to be willing to express their information need more elaborately than with a few keywords [Balog et al. 2008].

This is a revised and extended version of Balog et al. [2010].

This research was partially supported by the European Union’s ICT Policy Support Programme as part of the Competitiveness and Innovation Framework Programme, CIP ICT-PSP under grant agreement nr 250430, the PROMISE Network of Excellence co-funded by the 7th Framework Programme of the European Commission, grant agreement no. 258191, the Netherlands Organisation for Scientific Research (NWO) under project nrs 612.061.814, 612.061.815, 640.004.802, 380-70-011, the Center for Creation, Content and Technology (CCCT), the Hyperlocal Service Platform project funded by the Service Innovation & ICT program, and WAHSP project funded by the CLARIN-nl program, and under COMMIT project Infiniti.

Authors’ addresses: K. Balog, Department of Computer and Information Science, Norwegian University of Science and Technology; email: krisztian.balog@idi.ntnu.no; M. Bron and M. de Rijke, ISLA, University of Amsterdam; email: m.m.bron@uva.nl and maarten.de.rijke@gmail.com.

DOI 10.1145/2037661.2037667 http://doi.acm.org/10.1145/2037661.2037667

These additional means may include categories to which the target entities should belong or example entities. We focus on abstractions of these scenarios as they are evaluated in the context of INEX, the INitiative for the Evaluation of XML retrieval. In 2007, INEX launched an *entity ranking* track [de Vries et al. 2008], which also ran in 2008 [Demartini et al. 2009]. Here, entities are represented by their Wikipedia page, and queries asking for an entity are typed (that is, asking for entities belonging to certain categories) and may come with examples. Two tasks are being considered at INEX: (1) *entity ranking*, where query and target categories are given, and (2) *list completion*, where textual query, example entities, and (optionally) target categories are given.

Given that information needs involving entities can be formulated in so many ways, with so many ingredients (textual query, categories, examples), the natural system oriented question to ask is how to map these ingredients into the query component of a retrieval model. In this article, we focus on effectively capturing and exploiting category-based information. Several approaches to incorporating such information have been proposed (see Section 2 below), but there is no systematic account of approaches yet. One of the contributions of the article is a comprehensive overview of related work on this topic, entity retrieval in Wikipedia.

We introduce a probabilistic framework for entity retrieval that explicitly models category information in a theoretically transparent manner. Information needs and entities are represented as a tuple consisting of a term-based model plus a category-based model, both characterized by probability distributions. Ranking of entities is then based on similarity to the query, measured in terms of similarities between probability distributions. Our framework is capable of synthesizing all previous approaches proposed for exploiting category information in the context of the INEX Entity Ranking task. In this article, our focus is on two core steps: query modeling and query model expansion.

We seek to answer the following research questions. First, does our two-component query model improve over single component approaches, either term-based or category-based? Second, what are effective ways of modeling (blind) relevance feedback in this setting, using either or both of the term-based and category-based components?

Our main contribution in this paper is the introduction of a probabilistic retrieval model for entity search, in which we are able to effectively integrate term-based and category-based representations of queries and entities. We provide extensive evaluations and analyses of our query models and approaches to query expansion. Category-based feedback is found to be more beneficial than term-based feedback, and category-based feedback using example entities brings in the biggest improvements, bigger than combinations of blind feedback and information derived from example entities.

In Section 2 we discuss related work. We introduce our retrieval model in Section 3. In Section 4 we zoom in on query modeling and query model expansion; Section 5 is devoted to an experimental evaluation. We discuss and analyze our findings in Section 6 and conclude in Section 7.

2. RELATED WORK

We first describe previous work on query modeling, then review related work on entity-oriented search tasks, and finally consider related work on entity ranking tasks in the context of INEX.

2.1. Query Modeling

The main focus of this article—using information derived from terms, categories, and examples for capturing the information need underlying a query—is a clear example of query modeling, the process of representing a user's query to best capture the underlying information need. As the information need is often expressed using very few terms, query modeling tends to involve query expansion. In the language modeling approach to document retrieval, relevance models have been particularly influential examples of query expansion [Lavrenko and Croft 2001].

In the setting of semi-structured documents, query modeling has often been aproached using mixtures of multiple sources of evidence, possibly biased through the use of priors. For example, in web retrieval, anchor text and fielded queries, have proved effective [Kraaij et al. 2002], as has the combination of query operations, fielded queries, and fielded indexes [Mishne and de Rijke 2005; Metzler and Croft 2005]. In the setting of XML retrieval, layout structure can make a significant difference [Kamps et al. 2006]. Recent work has shown that automatically inferring structural information from keyword queries and incorporating this into a query model may lead to significant improvements [Kim et al. 2009].

Variations of query models, in the setting of standard document retrieval, include alternative ways of estimating the revised query model [Tao and Zhai 2006], using semantic information such as thesauri [Meij and de Rijke 2007] or ontologies [Järvelin et al. 2001] to inform the query model or using so-called example documents that inform the search engine about the type of document the user would like the search engine to retrieve [Balog et al. 2008].

As described in the following subsections, much of the work on query modeling in the setting of entity retrieval revolves around the use of category-based information. The models that we consider in Section 4 gauge these proposals in a general probabilistic framework.

2.2. Entity Retrieval

A range of commercial providers now support entity-oriented search, dealing with a broad range of entity types: people, companies, services, and locations. Examples include TextMap,[1] ZoomInfo,[2] Evri,[3] and the Yahoo! correlator demo.[4] They differ in their data sources, in the entity types they support, functionality, and user interface. Common to them, however, is their ability to rank entities with respect to a topic or to another entity. Little is known, however, about the algorithms underlying these applications.

Conrad and Utt [1994] introduce techniques for extracting entities and identifying relationships between entities in large, free-text databases. The degree of association between entities is based on the number of co-occurrences within a fixed window size. A more general approach is also proposed, where all paragraphs containing a mention of an entity are collapsed into a single pseudo document. Raghavan et al. [2004] re-state this approach in a language modeling framework and use the contextual language around entities to create a document-style representation, that is, entity language model, for each entity. This representation is then used for a variety of tasks: fact-based question answering, classification into predefined categories, and clustering and selecting keywords to describe the relationship between similar entities.

Sayyadian et al. [2004] introduce the problem of finding missing information about a real-world entity from text and structured data. Results show that entity retrieval over text documents can be significantly aided by the availability of structured data.

The TREC Question Answering track recognized the importance of search focused on entities with factoid questions and list questions (asking for entities that meet certain

[1] http://www.textmap.com/.

[2] http://www.zoominfo.com/.

[3] http://www.evri.com/.

[4] http://sandbox.yahoo.com/correlator.

constraints) [Voorhees 2005]. To answer list questions, systems have to return instances of the class of entities that match the description in the question. List questions are often treated as (repeated) factoids, but special strategies are called for as answers may need to be collected from multiple documents [Chu-Carroll et al. 2004]. Recognizing the importance of list queries [Rose and Levinson 2004], Google Sets allows users to enter some instances of a concept and retrieve others that closely match the examples provided [GoogleSets 2009]. Ghahramani and Heller [2006] develop an algorithm for completing a list based on examples using machine learning techniques.

The TREC 2005–2008 Enterprise track [Balog et al. 2009] featured an *expert finding* task: given a topic, return a ranked list of experts on the topic. Lessons learned involve models, algorithms, and evaluation methodology [Balog 2008; Balog et al. 2006]. Two important families of retrieval models for expert finding have emerged: *candidate-centric* models that first compile a textual representation of candidate experts by aggregating the documents associated with them and then rank these representations with respect to the topic for which experts are being sought; and *document-centric* models that start by ranking documents with respect to their relevance to the query and then rank candidate experts depending on the strength of their association with the top ranked documents. Many variations on these models have been examined, for a range of expertise retrieval tasks, exploring such features as proximity [Balog et al. 2009; Petkova and Croft 2007], document priors [Zhu et al. 2009], expert-document associations [Balog and de Rijke 2008], and external evidence [Serdyukov and Hiemstra 2008]. While expert finding focuses on a single entity type ("person") and a specific relation ("being an expert in"), the proposed methods typically do not model the concept of expertise; therefore, most of the approaches devised for expert finding can also be applied to the more general task of entity search.

Zaragoza et al. [2007] consider the case of retrieving entities in Wikipedia where instances are not necessarily represented by textual content other than their descriptive label. In 2007, the INEX Entity Ranking track (INEX-XER) [de Vries et al. 2008; Demartini et al. 2009] introduced tasks where candidate items are restricted to having their own Wikipedia page. Both Fissaha Adafre et al. [2007] and Vercoustre et al. [2007] addressed an early version of these tasks (entity ranking and list completion), inspired by the 2006 INEX pilot; other early work on the topic is due to Vercoustre et al. [2008].

2.3. Entity Ranking at INEX

Launched in 2002, INEX has been focused on the use of XML and other types of document structure to improve retrieval effectiveness. While the initial focus was on document and element retrieval, over the years, INEX has expanded to consider multimedia tasks as well as various mining tasks. In recent years, INEX has mainly been using Wikipedia as its document collection. An important lesson learned at INEX-XER is that exploiting the rich structure of the collection (text plus category information, associations between entities, and query-dependent link structure) may help improve retrieval performance over plain document retrieval [de Vries et al. 2008].

Nearly all INEX participants have used category information; many of them made this explicit in a separate category component in the overall ranking formula [Vercoustre et al. 2008; Weerkamp et al. 2009; Zhu et al. 2008; Jiang et al. 2009; Kaptein and Kamps 2009; Vercoustre et al. 2009]. A standard way of combining the category and term-based components was to use a language modeling approach and to estimate the probability of an entity given the query and category information [Jiang et al. 2009; Weerkamp et al. 2009; Zhu et al. 2008]. Calculating the similarity between the categories of answer entities and the target categories or between the categories of answer entities and the set of categories attached to example entities is sometimes based on lexical similarity [Vercoustre et al. 2008], on the content of categories (concatenating all text that belongs to that category) [Kaptein and Kamps 2009], or on the overlap ratio between sets of categories [Weerkamp et al. 2009]. Another popular solution was to add categories as a separate metadata field to the content of documents and apply a multi-field retrieval model (e.g., Zhu et al. [2008] and Demartini et al. [2008]).

Target category information provided as part of the query is not necessarily complete, as the assignment to categories by human annotators is far from perfect. Some teams have experimented with expanding the target categories, for example, using the category structure to expand with categories up to a certain level [Jämsen et al. 2008; Weerkamp et al. 2009; Tsikrika et al. 2008]. Others expand the target categories using lexical similarity between category labels and query terms [Vercoustre et al. 2008; Kaptein and Kamps 2009]. The categorization in Wikipedia is not a well-defined "is-a" hierarchy; that is, members of a subcategory are not necessarily members of its supercategory. To make up for this, Demartini et al. [2008] introduce a filtering method that adds only subcategories which are of the same type, according to the YAGO ontology [Suchanek et al. 2007]. Craswell et al. [2009] calculate a specificity score to filter out categories that are too general. Mixed results are reported with respect to the usefulness of category expansion: according to Thom et al. [2007], adding sub-categories and parent categories does not improve performance. Zhu et al. [2008] report that expanding the list of predefined categories with children and grandchildren categories is helpful, while the inclusion of parent categories hurts performance. Results in Demartini et al. [2008] show that expansion does not improve overall performance, but minor improvements are observable in early precision. Finally, Jämsen et al. [2008] report that expansion (using a different decay coefficient for the up and down directions) improves and, moreover, that it works better for the list completion task than for entity ranking; their explanation is that example entities provide a more extensive and fine-grained set of target categories.

As to query formulation for entity retrieval, stemming and stopwording are usually performed. Craswell et al. [2009] go beyond this and modify the query with NLP techniques, removing verbs while focussing on adjectives, nouns, and named entities; Murugeshan and Mukherjee [2008] attempt to improve on query modeling by identifying meaningful n-grams in the keyword query.

Several participants in the list completion task use the categories of example entities for constructing or expanding the set of target categories, using various expansion techniques [Craswell et al. 2009; Weerkamp et al. 2009; Jiang et al. 2009; Vercoustre et al. 2008; Zhu et al. 2008; Jämsen et al. 2008]; some use category information to expand the term-based model, see, for example, Weerkamp et al. [2009] and Kaptein and Kamps [2009]. Similarly to Zhu et al. [2008] and Weerkamp et al. [2009], we use example entities as explicit relevance feedback information, in Section 4.2.

There is a wide range of approaches looking for evidence in other documents, that is, besides the Wikipedia page corresponding to the entity. Both Zhu et al. [2008] and Jiang et al. [2009] employ a co-occurrence model, which takes into account the co-occurrence of the entity and query terms (or example entities) in other documents, by borrowing methods from the expert finding domain ([Zhu et al. 2006] and [Balog et al. 2006], respectively). Many entity ranking approaches utilize the link structure of Wikipedia, for example, as link priors [Kaptein and Kamps 2009] or using random walks to model multi-step relevance propagation between linked entities [Tsikrika et al. 2008]. Fissaha Adafre et al. [2007] use a co-citation based approach; independently, Pehcevski et al. [2008] expand upon a co-citation approach and exploit link co-occurrences to improve the effectiveness of entity ranking. Likewise, Kamps and Koolen [2008] show that if link-based evidence is made sensitive to local contexts, retrieval effectiveness can be improved significantly.

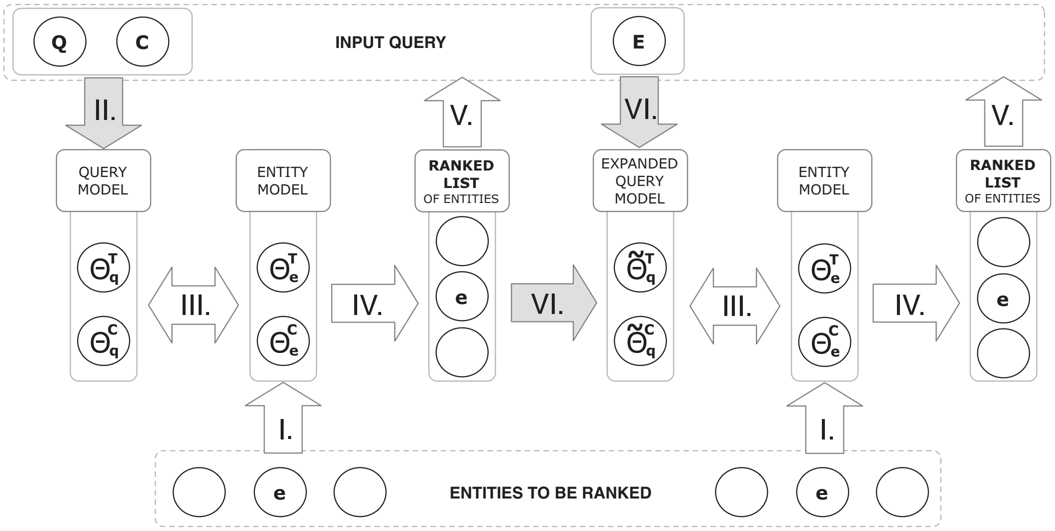

Fig. 1. A general scheme for entity ranking. The steps on which we focus in this article are indicated with grey arrows (Steps II and VI); all steps are explained in Section 3.1.

3. MODELING ENTITY RANKING

Now that we have reviewed related work, we are ready to present a general retrieval scheme for two entity ranking tasks, as they have been formulated within the INEX Entity Ranking track. In the *entity ranking* task, one is given a sequence of terms (Q) and a set of target categories (C), and has to return entities. For the *list completion* task we need to return entities given a sequence of query terms (Q), a set of target categories (C), and a set of similar entities (E).[5] For both tasks, we use q to denote the query that consists of all of the “user input,” that is, $q = \{Q, C, E\}$, where only the keyword part Q is compulsory.

3.1. A General Scheme for Entity Ranking

Figure 1 depicts our general scheme for ranking entities. The process goes as follows. We are given the user’s input, consisting of a query (Q), a set of input categories (C), and optionally a number of example entities (E). This input is translated into a query model, with a term-based and/or a category-based component (Step II); in our approach these components are characterized by probability distributions (denoted θ_q^T and θ_q^C). During retrieval this model is compared (in Step III) against models created for indexed entities (that have been derived in Step I; we also represent entities in terms of probability distributions, θ_e^T and θ_e^C). In Step IV a ranked list of entities is produced (based on Step III), which, in turn may (optionally) lead to a set of feedback entities (Step V), either explicitly provided by the user or obtained “blindly” based on the initial ranking. This feedback entity set (E) may (optionally) give rise to an expanded query model (Step VI); from that point onwards, we can repeat Steps III, IV, V, VI one or more times.

[5] Note that according to the original INEX-XER task definitions, list completion is the task of ranking entities given the free text query (Q) and the set of example entities (E), while target categories (C) are optional. Since our focus in this article is on modeling category information, we take example entities to be optional and consider scenarios both with and without using this piece of input data.

Our focus in this article is centered around the problem of modeling the query: (1) How can the user's input be translated into an initial query model (Step II)? And (2) How can this often sparse representation be refined or extended to better express the underlying information need (Step VI)? Specifically, we are interested in sources and components that play a role in estimating the term- and category-based representations of query models. Some entity ranking models may involve additional components or steps for entity modeling (e.g., priors) or for ranking (e.g., links between entities); this does not affect our general query modeling approach (Steps II and VI), and could straightforwardly be incorporated into the ranking part (Steps I, III, and IV) of our general framework, assuming that there are no dependencies on the query side that would need to be considered.

Below, we follow the roadmap outlined just now; the reader may find it helpful to return to Figure 1 and to the roadmap.

3.2. A Probabilistic Model for Entity Ranking (Steps III and IV)

We introduce a generative probabilistic framework that implements the entity ranking approach depicted in Figure 1. We rank entities e according to their probability of being relevant to a given information need (q): $P(e|q)$. That is, the probability to sample e from the model estimated by the query q. Instead of estimating this probability directly, we apply Bayes' rule and rewrite it:

$$P(e|q) \propto P(q|e) \cdot P(e), \tag{1}$$

where $P(e)$ is the prior probability of choosing a particular entity e, that we subsequently attempt to draw the query q from, with probability $P(q|e)$. To remain focused, we assume that $P(e)$ is uniform, thus, does not affect the ranking.

Each entity is represented as a pair: $e = (\theta_e^T, \theta_e^C)$, where θ_e^T is a probability distribution over terms, and θ_e^C is a probability distribution over categories. Similarly, the query is also represented as a pair: $q = (\theta_q^T, \theta_q^C)$, which is then optionally further refined, resulting in an expanded query model $\tilde{q} = (\tilde{\theta}_q^T, \tilde{\theta}_q^C)$ that is used for ranking entities. In the remainder of this section, equations are provided for q; the very same methods are used for ranking using the expanded query: $\tilde{q}$ simply needs to be substituted for q.

The probability of an entity generating the query is estimated using a mixture model:

$$P(q|e) = \lambda \cdot P\big(\theta_q^T|\theta_e^T\big) + (1-\lambda) \cdot P\big(\theta_q^C|\theta_e^C\big), \tag{2}$$

where λ controls the interpolation between the term-based and category-based representations. The estimation of $P(\theta_q^T|\theta_e^T)$ and $P(\theta_q^C|\theta_e^C)$ requires a measure of the difference between the query and entity models, both represented by (empirical) probability distributions. Here, we opt for the Kullback-Leibler (KL) divergence; before we delve into the specifics, we discuss the motivation behind this choice.

A straightforward way of estimating the probability $P(\theta_q^T|\theta_e^T)$ (or $P(\theta_q^C|\theta_e^C)$) would be to take the product of the individual term (category) probabilities in the entity model, raised to the power of their probability in the query model; formally, for the term-based case:

$$P\big(\theta_q^T|\theta_e^T\big) = \prod_{t \in \theta_q^T} P\big(t|\theta_e^T\big)^{P(t|\theta_q^T)}. \tag{3}$$

There are two pragmatic problems with computing this probability directly. First, the multiplication of very small probabilities would result in a floating-point underflow; a solution for dealing with that is to move computations to the log domain (where, subsequently, multiplications become additions), and calculate $\log P(\theta_q^T|\theta_e^T)$ ($\log P(\theta_q^C|\theta_e^C)$);

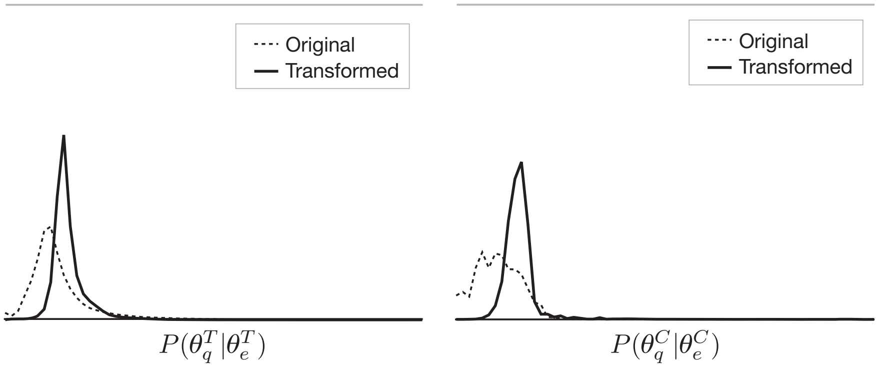

Fig. 2. Distributions of probabilities averaged over all queries in our test sets; (left): term-based, (right): category-based similarity. Original refers to the direct calculation of the probabilities (that is, Eq. (3) for the term-based case); transformed is based on KL-divergence (that is, Eq. (5) for the term-based case).

note that the exponential function of this expression needs to be taken, before combining the term- and category-based components (according to Eq. (2)). This leads us to the second issue; looking at the distributions of these probabilities averaged over all queries in our test sets (Figure 2, dashed lines), we find that they are skewed to the left. Moreover, the shapes of the term-based and category-based components are quite different; a better empirical performance can be achieved by making these distributions symmetrical using log transformed ratios (Figure 2, solid lines). Note that this transformation does not make a difference in the ranking generated by a single component (either term- or category-based), but it has a positive effect when the two need to be combined.

We, therefore, use KL divergence scores, instead of calculating the probabilities "directly" (as formulated in Eq. (3)), and estimate term-based similarity

$$KL\big(\theta_q^T||\theta_e^T\big) = \sum_t P\big(t|\theta_q^T\big) \cdot \log \frac{P\big(t|\theta_q^T\big)}{P\big(t|\theta_e^T\big)}, \tag{4}$$

where the probability of term t given the term-based model of entity e, $P(t|\theta_e^T)$ and given the term-based model of query q, $P(t|\theta_q^T)$, remain to be defined. To ensure that KL divergence is always well-defined, we require $P(t|\theta_e^T) > 0$ for any term t such that $P(t|\theta_q^T) > 0$; this is ensured by smoothing the entity model (see Section 3.3). Moreover, if the quantity $0 \cdot \log 0$ appears in the formula, it is interpreted as zero. (Note that calculating $KL(\theta_q^T||\theta_e^T)$ this way differs from $\log P(\theta_q^T|\theta_e^T)$ only in a query-dependent constant.) Since KL divergence is a score (which is lower when two distributions are more similar), we turn it into a probability using Eq. (5):

$$P\big(\theta_q^T|\theta_e^T\big) = z_T \cdot \big(\max KL\big(\theta_q^T||\cdot\big) - KL\big(\theta_q^T||\theta_e^T\big)\big), \tag{5}$$

where z_T is a normalization factor set as follows:

$$z_T = 1 \Big/ \sum_e \max\big(KL\big(\theta_q^T||\cdot\big) - KL\big(\theta_q^T||\theta_e^T\big)\big). \tag{6}$$

The category-based component of the mixture in Eq. (2) is calculated analogously to the term-based case:

$$P\big(\theta_q^C|\theta_e^C\big) = z_C \cdot \big(\max KL\big(\theta_q^C||\cdot\big) - KL\big(\theta_q^C||\theta_e^C\big)\big), \tag{7}$$

where

$$z_C = 1 \Big/ \sum_e \max\left(KL(\theta_q^C || \cdot) - KL(\theta_q^C || \theta_e^C)\right), \tag{8}$$

and

$$KL(\theta_q^C || \theta_e^C) = \sum_c P(c|\theta_q^C) \cdot \log \frac{P(c|\theta_q^C)}{P(c|\theta_e^C)}. \tag{9}$$

The probability of a category according to an entity's category model ($P(c|\theta_e^C)$) and the probability of a category according to the query's category model ($P(c|\theta_q^C)$) remain to be defined. Again, we require $P(c|\theta_e^C) > 0$ for any category c that might appear in the query's category model.

3.3. Entity Modeling (Step I)

We have just completed Steps III and IV. Following the roadmap provided at the end of Section 3.1, our next task is to describe the entity model component, that is, Step I. Steps II and VI are discussed in the next section.

3.3.1. Term-Based Representation. To estimate $P(t|\theta_e^T)$ we smooth the empirical entity model with the background collection to prevent zero probabilities. We employ Bayesian smoothing using Dirichlet priors which has been shown to achieve superior performance on a variety of tasks and collections [Zhai and Lafferty 2004; Losada and Azzopardi 2008] and set

$$P(t|\theta_e^T) = \frac{n(t,e) + \mu^T \cdot P(t)}{\sum_t n(t,e) + \mu^T}, \tag{10}$$

where $n(t,e)$ denotes the number of times term t occurs in the document representing entity e, $\sum_t n(t,e)$ is the total number of term occurrences, that is, the document length, $P(t)$ is the background model (the relative frequency of t in the collection), and μ^T is the smoothing parameter. Since entities correspond to Wikipedia articles, this representation of an entity is identical to constructing a smoothed document model for each Wikipedia page in a standard language modeling approach [Song and Croft 1999; Lafferty and Zhai 2001]. Alternatively, the entity model can be expanded with terms from related entities, that is, entities sharing the categories or entities linking to or from the Wikipedia page [Fissaha Adafre et al. 2007]. To remain focused, we do not explore this direction here.

3.3.2. Category-Based Representation. Category assignments in Wikipedia are neither consistent nor complete. This makes it impractical to make a binary decision between matching and non-matching categories, that is, simply filtering on the target categories is not sufficient. Analogously to the term-based case, to define the category-based representation of an entity e (θ_e^C), we smooth the maximum-likelihood estimate with a background model. We employ Dirichlet smoothing, based on the following intuition: entities with a richer category-based representation require less smoothing. Thus, the amount of smoothing applied is dynamically adjusted, depending on how many categories an entity is assigned to. We use the parameter μ^C to avoid confusion with μ^T:

$$P(c|\theta_e^C) = \frac{n(c,e) + \mu^C \cdot P(c)}{\sum_c n(c,e) + \mu^C}. \tag{11}$$

In Eq. (11), $n(c,e)$ is 1 if entity e is assigned to category c, and 0 otherwise; $\sum_c n(c,e)$ is the total number of categories e is assigned to; $P(c)$ is the background category model

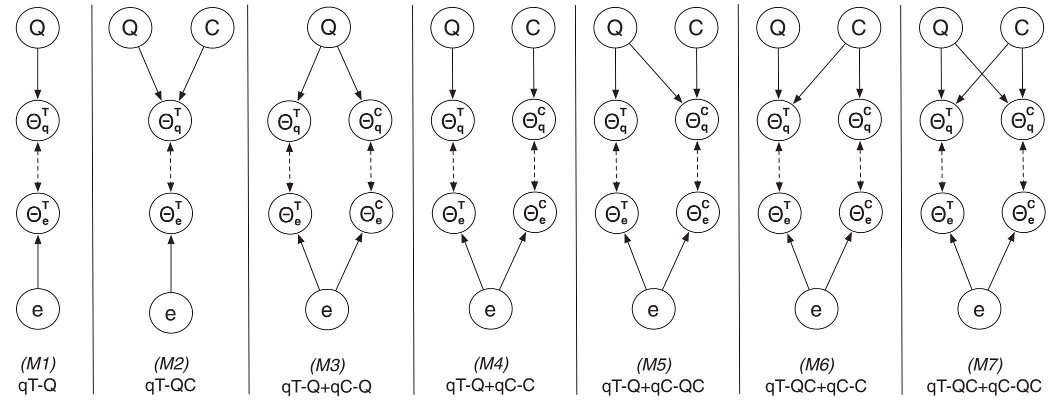

Fig. 3. Query models without expansion; Q stands for the term-based component of the topic, E for example entities, and C for the target categories; solid arrows from input to query model indicate that the input is used to create the model; dashed arrows indicate a comparison between models; acronyms function as labels for our models; acronyms and models are explained in Section 4.1.

and is set using a maximum-likelihood estimate:

$$P(c) = \frac{\sum_e n(c, e)}{\sum_c \sum_e n(c, e)}, \tag{12}$$

where $\sum_c \sum_e n(c, e)$ is the number of entity-category assignments in the collection.

We leave the exploration of other smoothing methods and a sensitivity analysis with respect to smoothing parameters as future work.

4. ESTIMATING AND EXPANDING QUERY MODELS

In this section we introduce methods for estimating and expanding query models: Steps II and VI in Figure 1; confer also the roadmap described at the end of Section 3.1. In particular, this means constructing the initial query model (θ_q) and the expanded query model ($\tilde{\theta}_q$); this, in turn, means estimating the probabilities $P(t|\theta_q^T)$, $P(c|\theta_q^C)$, $P(t|\tilde{\theta}_q^T)$, and $P(c|\tilde{\theta}_q^C)$ as listed in Figure 1 and discussed in Section 3.

4.1. Query Models (Step II)

We define a series of seven query models, M1–M7, each consisting of a term-based component and/or a category-based component; graphical depictions of the models are given in Figure 3. Our naming convention is as follows. A query model is denoted using an expression of the form *qT-X+qC-Y*, where the first half (qT-X) denotes the term-based component of the query model with various options detailed in the X-part, and the second half (qC-Y) denotes the category-based component with various options denoted by Y; for query models that lack a category-based component we omit the corresponding term qC-Y. While this section details the seven query models, Section 4.2 describes expansions of those query models.

(M1) qT-Q. This query model only has a term-based component and uses no category information (that is, it amounts to standard language modeling for document retrieval [Zhai and Lafferty 2004]). Writing $n(t, Q)$ for the number of times term t is present in query Q, we define the baseline term-based query to be

$$P_{bl}(t|\theta_q^T) = \frac{n(t, Q)}{\sum_t n(t, Q)}. \tag{13}$$

(M2) qT-QC. This model, previously described in Craswell et al. [2009], makes use of the possibility to expand the keyword query with terms derived from the names (labels) of input categories to form a term-based query model

$$P_{ct}\big(t|\theta_q^T\big) = \frac{\sum_{c\in C} n(t,c)}{\sum_{c\in C}\sum_t n(t,c)}, \tag{14}$$

where $n(t, c)$ denotes the number of times t occurs in the name of category c.
By using categories in this way, we remain in the term-based component of the query model; for the sake of simplicity we combine original terms and terms from category names with equal weight (by setting the mixture weight α^T to 0.5):

$$P\big(t|\theta_q^T\big) = (1-\alpha^T)\cdot P_{bl}\big(t|\theta_q^T\big) + \alpha^T \cdot P_{ct}\big(t|\theta_q^T\big). \tag{15}$$

(M3) qT-Q+qC-Q. From this point onwards, we distinguish categories from text in the query model; this is possible even if categories are not provided explicitly by the user. Our third model uses the keyword query (Q) to infer the category-component of the query model (θ_q^C), by considering the top N_c most relevant categories given the query. Relevance of a category is estimated based on matching between the name of the category and the query, that is, a standard language modeling approach on top of an index of category names, where $P(Q|c)$ is the probability of category c generating query Q:

$$P_q\big(c|\theta_q^C\big) = \begin{cases} P(Q|c)/\sum_{c\in N_c} P(Q|c), & \text{if } c \in \text{top } N_c \\ 0, & \text{otherwise.} \end{cases} \tag{16}$$

For the term-based query component, we use the baseline model (that is, set θ_q^T using Eq. (13)). The idea of using the keyword query to find relevant categories was first proposed in Thom et al. [2007] and was also used at INEX (e.g., [Vercoustre et al. 2008]).
(M4) qT-Q+qC-C. This model, previously described in Weerkamp et al. [2009], also employs a baseline term-based query component (that is, setting θ_q^T according to Eq. (13)), but it uses the input categories to form a category-based query model. Setting $n(c, q)$ to 1 if c is a target category, and $\sum_c n(c, q)$ to the total number of target categories provided with the topic statement, we put

$$P_{bl}\big(c|\theta_q^C\big) = \frac{n(c,q)}{\sum_c n(c,q)}. \tag{17}$$

This, in a sense, is the baseline "two-component" model; it uses the keyword query to form the term-based representation and the input categories to set the category-based representation of the query; both are maximum likelihood estimates.
(M5) qT-Q+qC-QC. Since input category information may be very sparse, it makes sense to enrich this component of the query by considering other categories that are relevant to the keyword query. This model uses a baseline term-based query component (Eq. (13)), and employs the mixture model on the category side:

$$P\big(c|\theta_q^C\big) = (1-\alpha^C)\cdot P_{bl}\big(c|\theta_q^C\big) + \alpha^C \cdot P_q\big(c|\theta_q^C\big). \tag{18}$$

Again, to keep things simple we allocate the probability mass equally between the two components (by setting the mixture weight α^C to 0.5).
(M6) qT-QC+qC-C. This model combines qT-Q+qC-C with names of input categories added to (M1) (qT-Q); the names contribute half of the probability mass to the term-based query model ($\alpha^T = 0.5$). The components $P(t|\theta_q^T)$ and $P(c|\theta_q^C)$ are estimated

as in Eq. (15) and (17), respectively.

$$\begin{aligned} P\big(t|\theta_q^T\big) &= (1-\alpha^T)\cdot P_{bl}\big(t|\theta_q^T\big) + \alpha^T \cdot P_{ct}\big(t|\theta_q^T\big), \\ P\big(c|\theta_q^C\big) &= P_{bl}\big(c|\theta_q^C\big). \end{aligned}$$

(M7) qT-QC+qC-QC. This model combines (M5) (qT-Q+qC-QC) and (M6) (qT-QC+qC-C); input category labels are added to the term-based query model and query terms are used to add relevant categories to the category-based model. For the components $P(t|\theta_q^T)$ and $P(c|\theta_q^C)$, Eq. (15) and (18) are used, respectively. Again we allocate probability mass by setting $\alpha^T = 0.5$ and $\alpha^C = 0.5$.

$$\begin{aligned} P\big(t|\theta_q^T\big) &= (1-\alpha^T)\cdot P_{bl}\big(t|\theta_q^T\big) + \alpha^T \cdot P_{ct}\big(t|\theta_q^T\big), \\ P\big(c|\theta_q^C\big) &= (1-\alpha^C)\cdot P_{bl}\big(c|\theta_q^C\big) + \alpha^C \cdot P_{q}\big(c|\theta_q^C\big). \end{aligned}$$

4.2. Expanded Query Models (Step VI)

Expansions of a basic query model can take place on either (or both) of the two components: term-based and category-based. The general form we use for expansion is a mixture of the baselines defined in Section 4.1 (subscripted with *bl*) and an expansion (subscripted with *ex*). For the term-based component we set

$$P\big(t|\tilde{\theta}_q^T\big) = (1-\lambda^T)\cdot P_{bl}\big(t|\theta_q^T\big) + \lambda^T \cdot P_{ex}\big(t|\theta_q^T\big), \tag{19}$$

and for the category-based component we set

$$P\big(c|\tilde{\theta}_q^C\big) = (1-\lambda^C)\cdot P_{bl}\big(c|\theta_q^C\big) + \lambda^C \cdot P_{ex}\big(c|\theta_q^C\big), \tag{20}$$

where λ^T and λ^C are the weights on the expanded query models in Equations (19) and (20), respectively.

We present a general method for estimating the expansions $P_{ex}(t|\theta_q^T)$ and $P_{ex}(c|\theta_q^C)$, using a set of feedback entities, FB. This feedback set may be obtained by taking the top N relevant entities according to a ranking obtained using the initial query. We use E' to denote this set of blind feedback entities. Alternatively, one might assume explicit feedback, such as the example entities (denoted by E) in our scenario. Constructing the feedback set by using either blind feedback ($FB = E'$), example entities ($FB = E$), or a combination of both ($FB = E' \cup E$), yields three query expansion methods. Depending on where feedback takes place we have nine variations in total, as shown in Figure 4. Next, we define our methods for constructing the expanded query models from a set of feedback entities.

A word on notation: we use variations on the schema qT-FB+qC-FB' to denote expanded query models; here, the term-based or category-based component may be missing and FB and FB' are (possibly empty) sets of feedback entities.

4.2.1. Term-Based Expansion. Given a set of feedback entities FB, the expanded query model is constructed as

$$P\big(t|\tilde{\theta}_q^T\big) = \frac{P_{K_T}(t|FB)}{\sum_{t'} P_{K_T}(t'|FB)}, \tag{21}$$

where t' stands for a term, and $P_{K_T}(t|FB)$ denotes the top K_T terms with the highest $P(t|FB)$ value, calculated according to Eq. (22):

$$P(t|FB) = \frac{1}{|FB|}\sum_{e\in FB}\frac{n(t,e)}{\sum_t n(t,e)}, \tag{22}$$

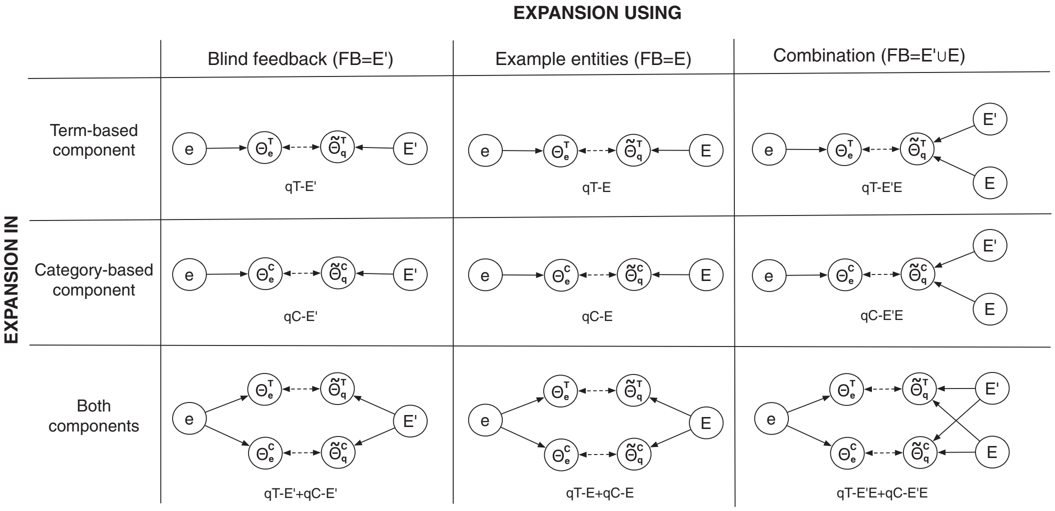

Fig. 4. Models with expansion; same graphical and notational conventions as in Figure 3; acronyms explained in Section 4.2.

where $\sum_t n(t,e)$ is the total number of terms, that is, the length of the document corresponding to entity e.

The approach we propose is adopted from Balog et al. [2008], where query models are estimated from example documents; a simplification we make is that all feedback documents are assumed to be equally important. A fundamental difference between this approach and relevance models by Lavrenko and Croft [2001] is that relevance models assume conditional dependence between the query and the terms selected for expansion. Following Balog et al. [2008], we lift this assumption as we want to avoid biasing the expansion term selection towards the query (and thereby possibly loosing important aspects, not covered by the original keyword query).

4.2.2. Category-Based Expansion. Analogously to the term-based case, we calculate the expanded query model for categories

$$P\big(c|\tilde{\theta}_q^C\big) = \frac{P_{K_C}(c|FB)}{\sum_{c'} P_{K_C}(c'|FB)}, \tag{23}$$

where c' stands for a category, and $P_{K_C}(c|FB)$ denotes the top K_C categories with the highest $P(c|FB)$ value, calculated according to Eq. (24), (where, as before, $n(c,e)$ is 1 if e belongs to c):

$$P(c|FB) = \frac{1}{|FB|}\sum_{e\in FB}\frac{n(c,e)}{\sum_t n(c,e)}. \tag{24}$$

5. EXPERIMENTAL EVALUATION

In order to answer the research questions listed in Section 1, we run a set of experiments. Next we detail our experimental setup, present the results, and formulate answers.

5.1. Experimental Setup

5.1.1. Test Collection. We use the test sets of the 2007 and 2008 editions of the INEX Entity Ranking track (INEX-XER) [de Vries et al. 2008; Demartini et al. 2009], that

Table I. Results on the Entity Ranking Task, No Expansion. Best Results per Test Set in Boldface

				XER2007		XER2008	
Model	λ	θ_q^T	θ_q^C	MAP	MRR	xinfAP	MRR
(M1) qT-Q	1.0	Eq. 13	—	0.1798	0.2906	0.1348	0.2543
(M2) qT-QC	1.0	Eq. 15	—	0.1706	0.3029	0.1259	0.2931
(M3) qT-Q+qC-Q	0.5	Eq. 13	Eq. 16	0.2410	0.3830	0.1977	0.3190
(M4) qT-Q+qC-C	0.5	Eq. 13	Eq. 17	0.2162	0.4168	0.3099	0.4783
(M5) qT-Q+qC-QC	0.5	Eq. 13	Eq. 18	**0.2554**	**0.4531**	**0.3124**	**0.5024**
(M6) qT-QC+qC-C	0.5	Eq. 15	Eq. 17	0.1881	0.2948	0.2911	0.4439
(M7) qT-QC+qC-QC	0.5	Eq. 15	Eq. 18	0.2255	0.3346	0.2950	0.4357

use (a dump of) the English Wikipedia as document collection from which (articles corresponding to) entities are to be returned. The collection consists of over 650,000 documents plus a hierarchy of (over 115,000) categories; this is not a strict hierarchy, but a graph. Wikipedia articles are labeled with categories, but these assignments are not always consistent and far from complete [de Vries et al. 2008].

Tasks. INEX-XER has two tasks: *entity ranking* and *list completion*. An entity ranking topic specifies a free-text query Q and target categories C. In the (original) list completion task, the topic statement consists of a free-text query Q and examples entities E, without knowledge of C; the task is to complete the lists of entities. As our focus in this article is on using categories, we consider a variation of the list completion task where this information (C) is always provided by the user.

5.1.2. Topics and Judgments. Two sets of topics are available for INEX-XER. For 2007 a test set (XER2007) of 46 topics was created for the entity ranking track, 25 of which were specifically developed and assessed by track participants. On average, pools contained about 500 entities per topic [de Vries et al. 2008]. For 2008, a test set (XER2008) was compiled that contains 35 topics, developed and assessed by track participants [de Vries et al. 2008]. The metrics we use are Mean Average Precision (MAP) and Mean Reciprocal Rank (MRR) for the XER2007 topic set. For XER2008, xinfAP replaces MAP [Yilmaz et al. 2008] as it is a better estimate of Average Precision (AP) in the case of incomplete assessments [Demartini et al. 2009].

5.1.3. Parameter Settings. Our models involve a number of parameters. In this section we apply baseline settings for these parameters, and we use the values for all models. Specifically, we use the average document length for the term-smoothing parameter ($\mu^T = 411$) and the average number of categories assigned to an entity for the category-based smoothing parameter ($\mu^C = 2.2$). Our mixture models involve two components to which we assign equal importance, that is, $\lambda = \lambda^T = \lambda^C = 0.5$. In Section 6.2 we investigate the sensitivity of our models with respect to these parameters.

5.2. The Performance of Our Query Models

We examine the effectiveness of our query models and, in particular, of the use of two components—for terms and categories—on the *entity ranking* task. In the experiments that involve the keyword query in the construction of the category-component of the query model, we consider the top 10 categories relevant to the query, that is, set $N_c = 10$ (see Eq. (16)). Table I lists the results for the query models defined in Section 4.1, using the default parameter settings detailed in Section 5.1. In Section 6 we report on optimized runs and compare them against the best scores obtained at INEX-XER.

We compare the performance of the models using a two-tailed t-test at a significance level of $p = 0.05$. In Table I, we see that simply flattening the target category information and adding category names as terms to the term component is not an effective strategy; see (M1) vs. (M2), (M4) vs. (M6), and (M5) vs. (M7). When we consider

Table II. Results on the Entity Ranking and List Completion Tasks, with Expansion

				XER2007		XER2008	
Model	*FB*	$\tilde{\theta}_q^T$	$\tilde{\theta}_q^C$	MAP	MRR	xinfAP	MRR
Entity ranking (blind feedback only)							
BASELINE (no expansion)				0.2554	**0.4531**	0.3124	0.5024
qT-E'	$\{E'\}$	Eq. 21	—	0.2511	0.3654$^{\triangledown}$	0.3214	0.4694
qC-E'	$\{E'\}$	—	Eq. 23	**0.2601**	0.4516	0.3315$^{\triangle}$	**0.5042**
qT-E'+qC-E'	$\{E'\}$	Eq. 21	Eq. 23	0.2541	0.4090	**0.3365**$^{\triangle}$	0.4984
List completion (blind feedback and/or examples)							
BASELINE (no expansion)				0.2202	0.4042	0.2729	0.4339
Blind feedback							
qT-E'	$\{E'\}$	Eq. 21	—	0.2138	0.3235$^{\triangledown}$	0.2814	0.4139
qC-E'	$\{E'\}$	—	Eq. 23	0.2258	0.3858	0.2968$^{\triangle}$	0.4777
qT-E'+qC-E'	$\{E'\}$	Eq. 21	Eq. 23	0.2197	0.3576	0.3017$^{\triangle}$	0.4768
Examples							
qT-E	$\{E\}$	Eq. 21	—	0.2376$^{\triangle}$	0.3875	0.2886	0.4274
qC-E	$\{E\}$	—	Eq. 23	0.3141$^{\triangle}$	**0.5380**$^{\triangle}$	0.3873$^{\triangle}$	0.6123$^{\triangle}$
qT-E+qC-E	$\{E\}$	Eq. 21	Eq. 23	**0.3267**$^{\triangle}$	0.5357$^{\triangle}$	**0.3926**$^{\triangle}$	**0.6353**$^{\triangle}$
Blind feedback plus examples							
qT-E'E	$\{E', E\}$	Eq. 21	—	0.2200	0.3193$^{\triangledown}$	0.2843	0.4036
qC-E'E	$\{E', E\}$	—	Eq. 23	0.2565$^{\triangle}$	0.4416	0.3286$^{\triangle}$	0.4999
qT-E'E+qC-E'E	$\{E', E\}$	Eq. 21	Eq. 23	0.2475$^{\triangle}$	0.3854	0.3315$^{\triangle}$	0.4678

Best results in boldface. Baseline corresponds to model (M5) in Table I. Significant differences with baseline denoted with $^{\triangle}$ and $^{\triangledown}$.

category-based information provided with the input query as a separate component, we do see improvements across the test sets: see (M1) vs. (M3) (significant for 2007 and 2008) and (M2) vs. (M6) (significant for 2008). As to the category-component, the switch from using only the keyword query for its construction to using target categories (C) defined explicitly by the user ((M3) vs. (M4)) does not lead to consistent improvements (although the improvement is significant on the 2008 set); the move from the latter to a combination of both ((M4) vs. (M5) (significant on the 2007 set) and (M6) vs. (M7)) leads to consistent improvements for all tasks and measures.

5.3. The Performance of Our Expanded Query Models

Next, we report on the effectiveness of the expansion models depicted in Figure 4. A quick note before we start: when we only use Q and C, results are evaluated on the *entity ranking* task. When E is also used we evaluate results on the *list completion* task. Some notation: E' denotes a pseudo-relevant set of entities, that is, the top N obtained using methods of the previous subsection; and E denotes a set of example entities. When we use example entities for expansion, we need to remove them from the runs, that is, use the list completion relevance judgments. In order to have a fair comparison between approaches reported in this subsection, we need to do that for the pseudo-feedback runs as well, that is, when we only use E'. We use the following settings.

—Number of feedback entities (N): 5.
—Number of feedback categories (K_C): 10.
—Number of feedback terms (K_T): 15.
—Default values for λ, λ^T, and λ^C: 0.5.

Table II presents the results of query expansion, applied on top of the best performing run from the previous subsection (M5). We find that category-based feedback always outperforms term-based feedback. In case of blind feedback, category-based expansion brings in improvements (significant for 2008, MAP), while term-based expansion mostly hurts (significantly so on 2007, MRR). Example-based feedback leads to

Table III. Description of the Topics Highlighted in the Topic-Level Analysis

ID	Query Part	Category Part
#30	Space history astronaut cosmonaut engineer	astronauts
#31	Film starring Steven Seagal	(movies), (films)
#45	Dutch artists paris	Dutch painters
#64	Alan Moore graphic novels adapted to film	graphic novels
#79	Works by Charles Rennie Mackintosh	buildings and structures
#91	Paul Auster novels	novels
#132	Living nordic classical composers	(21st century classical composers), (living classical composers), (Finnish composers)
#141	Universities in Catalunya	Catalan universities
#144	Chess world champions	(chess grandmasters), (world chess champions)

improvements for both term-based and category-based expansion, but the latter is far more substantial; relative improvements can be up to +42% in MAP (2007 and 2008 topics) and +41% in MRR (2008 topics). The combination of blind feedback and examples improves over blind feedback, but is outperformed by category-based feedback using examples. An interesting observation is that the combination of category-based and term-based feedback is shown to not perform considerably better (in any of the settings) than category-based feedback alone.

6. DISCUSSION

In this section we analyze the experimental results obtained in Section 5. We start with a topic-level analysis and follow with an analysis of the sensitivity of our models to their parameters. Finally, we compare the performance of our approaches to published results obtained with similar methods.

6.1. Topic-Level Analysis

In this section we analyze the performance of our query models by offering a topic-level analysis, contrasting the performance of models on a topic-by-topic basis. Table III lists the topics highlighted in the analysis. We refer to the topic components as follows: to the topic ID as #N, to the query part as #N_Q, and the category part as #N_C. For our comparisons, we focus on Average Precision (AP) scores and use the $\triangle$AP notation to denote changes.

6.1.1. Query Models. We compare the addition of a number of features to our query models. We only consider models without feedback and use the numbering introduced in Table I to refer to our models (e.g., (M1), (M2), ...).

Flattening category information. Figure 5 shows the difference in AP per topic between query models that include the category labels to the query in the term-based component ((M2), (M6) and (M7)) and models that do not ((M1), (M4) and (M5)). We see that for (M1) vs. (M2), roughly as many topics are helped as hurt by including category labels in the term-based component; for (M4) vs. (M6) and (M5) vs. (M7), more topics are hurt than helped when category information is added to the term-based component.

We take a closer look at topic #31 (XER2007) for which scores decrease most in all comparisons. Table IV shows the top 10 results returned for each model, with relevant entities indicated in bold. We observe that some non-relevant entities are only returned for models (M2), (M6), and (M7) (e.g., “Action movie” and “Indiana Jones 4”). These non-relevant entities are returned because of the category information; the category labels (#31_C) are terms associated with movies in general. Adding them shifts the model from a specific type of movie (that is, #31_Q) to any type of movie. This example illustrates how general categories disrupt this type of modeling strategy.

Table IV. The Top 10 Ranked Entities for Topic #31 (XER2007)

rank	(M1) qT-Q	(M4) qT-Q+qC-C	(M5) qT-Q+qC-QC
1	Steven Seagal	Rock-A-Doodle	**Shadows of the Past**
2	Martial arts film	**Shadows of the Past**	Martial arts film
3	**On Deadly Ground**	Martial arts film	**On Deadly Ground**
4	**Shadows of the Past**	**On Deadly Ground**	**Submerged**
5	**Under Siege**	**Submerged**	**Out for Justice**
6	**Fire Down Below**	**Out for Justice**	**Under Siege**
7	List of films set in Japan	**Under Siege**	Steven Seagal
8	**Submerged**	Steven Seagal	**The Glimmer Man**
9	**Out for Justice**	**The Glimmer Man**	**Hard to Kill**
10	**Into the Sun (film)**	**Hard to Kill**	**Fire Down Below**
rank	(M2) qT-QC	(M6) qT-QC+qC-C	(M7) qT-QC+qC-QC
1	Steven Seagal	Rock-A-Doodle	Rock-A-Doodle
2	Martial arts film	Martial arts film	Martial arts film
3	Action movie	**Shadows of the Past**	**Shadows of the Past**
4	**The Patriot**	Action movie	Action movie
5	**On Deadly Ground**	**On Deadly Ground**	**On Deadly Ground**
6	**Shadows of the Past**	History of science fiction films	The Sugarland Express
7	**Fire Down Below**	**Above the Law (film)**	History of science fiction films
8	**Above the Law (film)**	**Submerged**	Poltergeist film series
9	Jean Claude Van Damme	Sex in film	Indiana Jones 4
10	Katherine Heigl	Cult film	**Above the Law (film)**

Boldface indicates relevant entities.

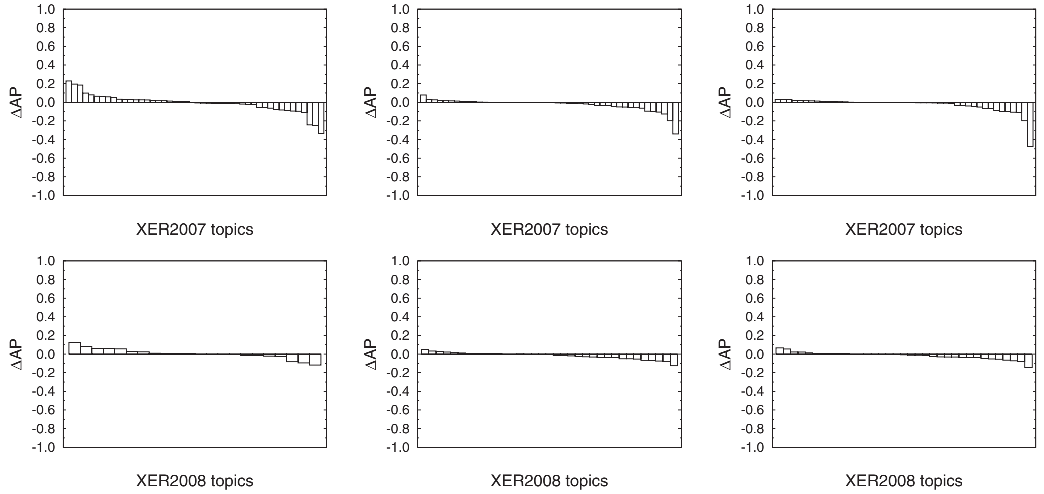

Fig. 5. Flattening the target category information and adding category names as terms to the term-based query component: (left): (M1) = Baseline vs. (M2); (center): (M4) = Baseline vs. (M6); (right): (M5) = Baseline vs. (M7) for the XER2007 (top) and XER2008 data sets (bottom).

Combination with the category component. Figure 6 shows the per topic differences in AP between query models that only use the term-based component ((M1) and (M3)) and models that combine the term-based component with the category-based component ((M2) and (M6)). For XER2008 (bottom) the MAP score increases when the components are combined. MAP also improves on the XER2007 topics (top), although AP scores of some topics show a sharp decrease, for example, topic #79. We take a closer look at this topic.

Table V shows the top 10 entities for each model. (M3) only finds a single relevant entity and (M6) finds none: both models demonstrate a shift in topic. (M6) finds any

Table V. The Top 10 Ranked Entities for Topic #79 (XER2007)

rank	(M1) qT-Q	(M2) qT-QC
1	Charles Rennie Mackintosh	Charles Rennie Mackintosh
2	**Queen's Cross Church**	Liverpool Cathedral
3	**Hunterian Museum and Art Gallery**	Architecture of the United Kingdom
4	**Glasgow School of Art**	**Hunterian Museum and Art Gallery**
5	**78 Derngate**	1928 in architecture
6	Margaret MacDonald (Artist)	International style (architecture)
7	Liverpool Cathedral	**Queen's Cross Church**
8	Glasgow School	**Glasgow School of Art**
9	Charles Macintosh	Modern architecture
10	Design classic	Josef Hoffmann
rank	(M3) qT-Q+qC-Q	(M6) qT-QC+qC-C
1	Tiu Keng Leng	World's largest buildings
2	Design classic	Pavilion (structure)
3	**Queen's Cross Church**	Rafter
4	Architecture of the United Kingdom	Rotunda (architecture)
5	Jeremy Broun	Charnel house
6	Walberswick	Yard (land)
7	Bellahouston	Geodesic dome
8	Collioure	Shed
9	International style (architecture)	Multi-storey car park
10	Kilmacolm	Wickiup

Boldface indicates relevant entities.

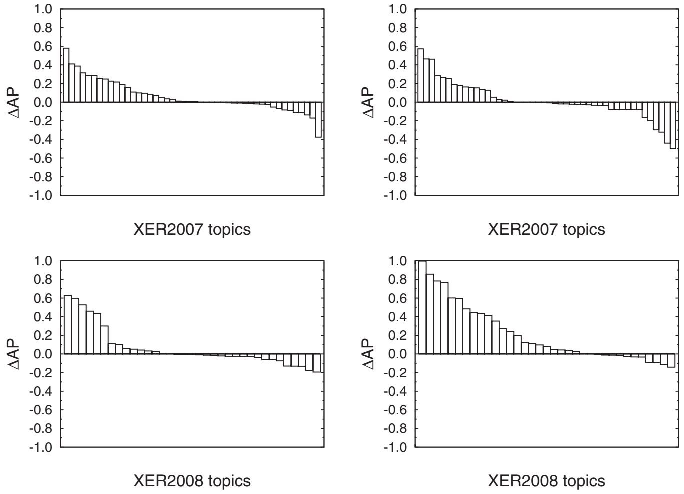

Fig. 6. Adding category-based information as a separate component: (left): (M1) = Baseline vs. (M3); (right): (M2) = Baseline vs. (M6) for the XER2007 (top) and XER2008 (bottom) data sets.

entity related to buildings and structures, caused by the target category (#79_C) being too general. (M3) finds entities that belong to target categories found relevant to the query; in this case categories with "charles" in their label, but unrelated to the topic (e.g., "Charles Mingus albums" and "Charles Dickens novels"). This example illustrates that combinations with the category-based component fail when categories have one of the following characteristics: (1) the category is too general and the relevant entities form only a small proportion of the category, or (2) the category is off topic; none of the relevant entities belong to the category. In most other cases AP scores do increase when a category-based component is added. An extreme case is topic #144 that achieves a perfect score; here, the target categories (#144_C) contain only the relevant entities. We

Table VI. The Top 10 Ranked Entities for Topic #91

rank	M3 qT-Q	M4 qT-QC
1	**The New York Trilogy**	**In the Country of Last Things**
2	**Moon Palace**	The Winthrop Woman (novel)
3	**Brooklyn Follies**	**Extremely Loud and Incredibly Close**
4	**In the Country of Last Things**	**Novel sequence**
5	**The Book of Illusions**	The City and the Pillar
6	**Oracle Night**	Going After Cacciato
7	**The Music of Chance**	**Raj Quartet**
8	Paul Auster	**Paul Clifford**
9	Fanshawe (novel)	**The Age of Reason (Sartre)**
10	French literature of the 19th century	The Notebooks of Malte Laurids Brigge

Boldface Indicates Relevant Entities.

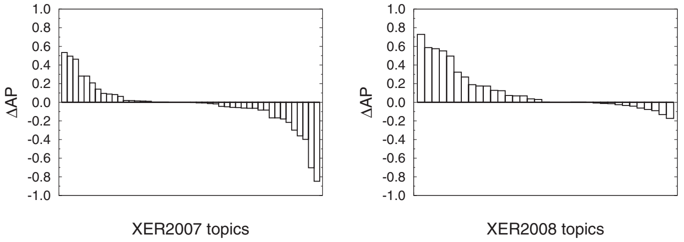

Fig. 7. Using query-based categories vs. target categories for the category component: (M3) = Baseline vs. (M4).

find that for most topics there is an optimal Wikipedia category that contains mostly relevant entities and few others.

Query-based categories versus target categories. Figure 7 shows the per topic differences in AP per topic between a model that uses query-based categories for the category-based component (M3) and a model that uses target category information defined explicitly by the user (M4). On the XER2008 topics (right), the difference in AP is more often positive than negative; (M4) performs better. On the XER2007 topics (left), the opposite occurs; the difference in AP is more often negative and (M3) performs better. We investigate topic #91 as an example where using the query for finding relevant categories is beneficial and topic #141 as an example where using the explicit target category information is advantageous.

Table VI shows the top 10 entities for topic #91. The top 10 entities according to (M3) form a perfect ranking up to rank 7, while the top 10 for (M4) contains noise. It turns out that (M4) uses a general target category—$\#91_C$ ("novels"), while one of the categories found for (M3) is "books by Paul Auster" which contains almost exclusively relevant entities. In the case of topic #141, an optimal category is given ($\#141_C$), so (M4) performs well. This category is not found by using the query, however, as the category label contains the term "Catalan" instead of "Catalunya," which causes poor performance for (M3).

These examples illustrate that using query-based categories performs better when (1) there is an optimal category and (2) the query terms match with this category. In case the optimal category is given, as part of the topic definition, (M4) performs better.

Target categories versus combination of query-based and target categories. The observations in the previous paragraph suggest that a combination of target categories and query-based categories in the query-based component improves over using either one alone. Figure 8 shows the difference in AP score per topic between models that use only the target category in the category-based component ((M4) and (M6)) and models

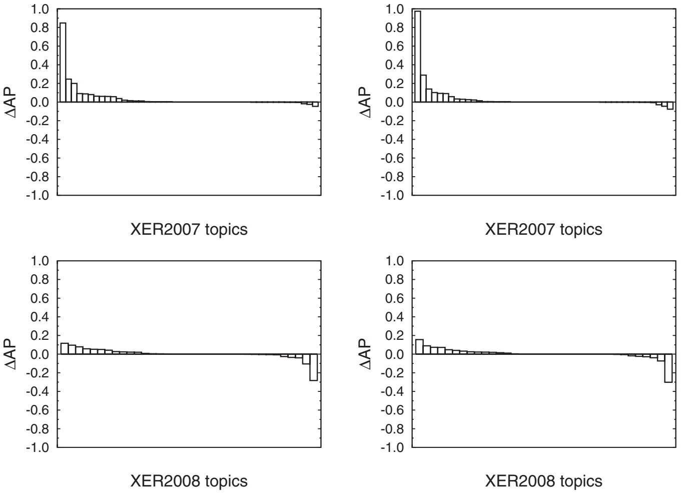

Fig. 8. Using category information vs. using the query and category information for the category component: (left): (M4) = Baseline vs. (M5); (right): (M6) = Baseline vs. (M7) for the XER2007 (top) and XER2008 (bottom) data sets.

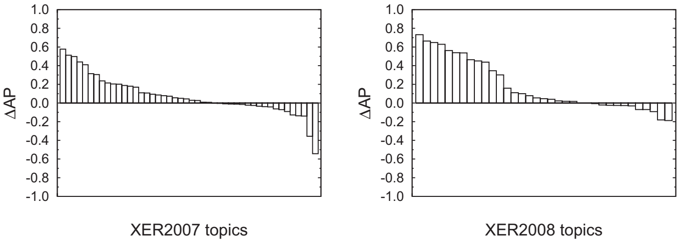

Fig. 9. Comparison of two extremes; using only the query vs. best performing model: (M1) = Baseline vs. (M5).

that use both ((M5) and (M7)). On the XER2007 data combining the target category with query-based categories almost never decreases AP, when it does it is only slightly ($\leq$0.10). On the XER2008 data we see the same trend with the exception of topic #132. In this case the categories matching the query do not add any relevant categories, while the given categories $\#132_C$ are optimal categories.

Two extremes. Figure 9 shows a comparison between two extremes; using only the query (M1) versus the best performing model (M5), according to Table I. On the XER2007 topics (left), adding extra query and category information decreases performance on 16 topics, is the same on five (difference is $\leq$0.01), and improves on 25 topics. On the XER2008 topics (right) adding information decreases performance on ten topics, is the same on three, and improves on 22 topics. For both topic sets, the gains are bigger than the losses, indicating an obvious benefit in using extra information in the query model, although there are some clear individual exceptions. As we observed in our earlier analysis, performance on topics experiencing a loss is influenced negatively by two facts: the given categories are too general and there is no optimal category. In cases where there is an optimal category, and it can be identified using the query, performance will increase.

Table VII. The Top 10 Terms and Categories Resulting from Blind Feedback, for Topics #30, #45, and #64

Topic #30		Topic #45		Topic #64	
Category	Term	Category	Term	Category	Term
European astronauts	history	Dutch painters	work	anarchism	town
Astronauts	oper	Dutch artists	artist	films based on novels	series
Russian cosmonauts	station	Paris	museum	dystopian fiction	film
Slovenian americans	air	Dutch martial artists	Paris	graphic novels	book
Soviet union cosmonauts	engineer	religion in paris	paint	films based on comics	characater
Polish astronauts	training	Paris culture	draw	Jack the Ripper	small
Russian astronauts	science	Paris metro	Dutch	2001 films	published
Chinese astronauts	space	mixed martial artists	Netherland	Vertigo titles	story
Bulgarian cosmonauts	mission	Paris albums	gallery	Alan Turing	kill
Belarusian cosmonauts	flight	Paris rer	painter	film adap from Vonnegut	novel

Performance of #30 is not influenced by feedback; #45 is influenced negatively; and the performance of #64 is influenced positively.

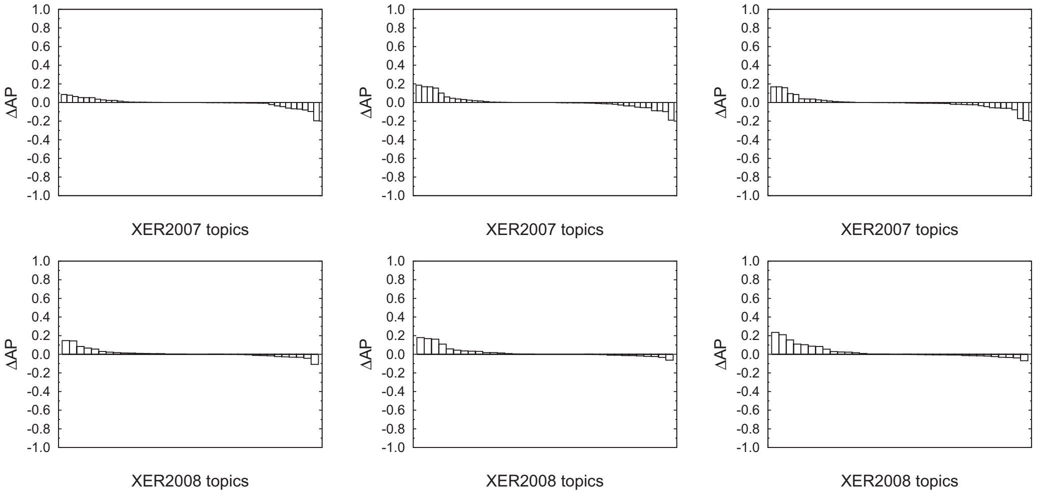

Fig. 10. Baseline vs. blind feedback: (left): term-based; (center): category-based; (right): term- and category-based combined for the XER2007 (top) and XER2008 data sets.

6.1.2. Expanded Query Models. Next we consider the use of feedback for the term and/or category based component (that is, blind feedback, example based feedback, or a combination) on top of the best model from Section 5.2, (M5).

Blind feedback versus no feedback. Figure 10 shows the difference in AP between (M5) and (M5) extended with three types of blind feedback: term-based, category-based, and a combination. On the XER2007 data set (top), the gains and losses are in balance. The AP scores of the XER2008 topics (bottom) are generally positively affected. On most topics, however, the blind feedback methods have no influence. This suggests that the terms and categories added to the model in the feedback step contain relevant information but no new information.

Table VII shows the terms and categories resulting from blind feedback for three topics: #30, where performance is not affected by feedback; #45, where performance is influenced negatively, and #64, where performance is positively influenced. For #45, the feedback categories cause the model to drift towards (martial) artists and culture in general (away from "Dutch artists in Paris"), while the feedback terms are relevant to the topic. For the other two topics, both the feedback terms and categories are relevant. In the case of #30, the optimal category is already found by (M5); feedback categories

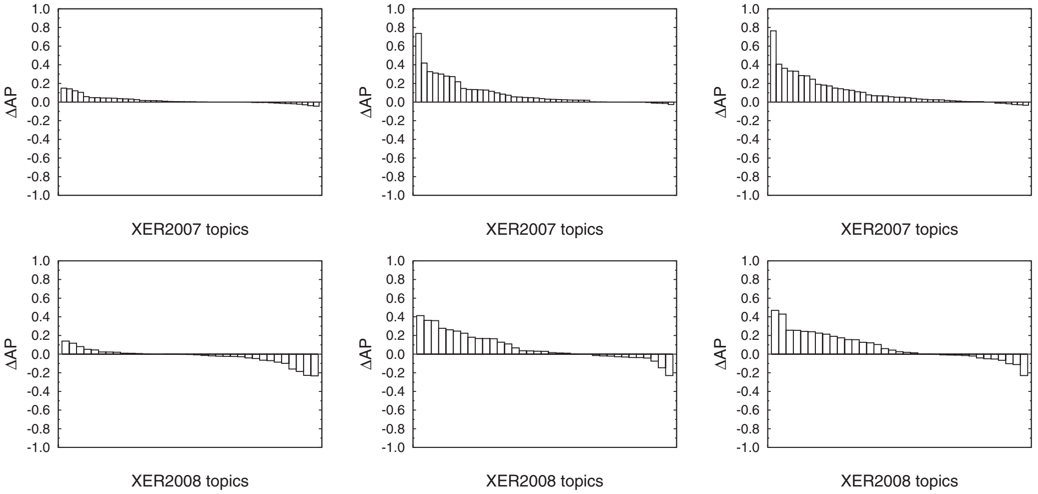

Fig. 11. Baseline (M5) vs. example entities: (left): term-based, (center): category-based, (right): term- and category-based combined for the XER2007 (top) and XER2008 (bottom) data sets .

add only more specific variations (European/Russian astronauts). The same holds for the terms; most of the feedback terms are already present in the query, while the new terms do not shift the model to a new topic. In the case of #64, feedback does introduce relevant terms and categories, for example, “films based on comics” and “Vertigo titles” (Vertigo is a comics publisher). This example illustrates that feedback is only beneficial if the query model is not already optimally defined by the baseline model (M5). In such cases feedback can increase performance, depending on the top ranked entities found by (M5).

Example-based feedback versus no feedback. Figure 11 shows the difference in AP between (M5) and (M5) extended with three types of example-based feedback: term-based, category-based, and a combination. For term-based feedback, results are mixed, with slightly increasing and decreasing AP scores. Most topics, however, are unaffected, suggesting that feedback does not introduce any new terms. For the category-based feedback, as well as for the combination of term- and category-based feedback, we observe that AP scores increase for most topics on both data sets. These results indicate that target categories derived from examples are most effective in building an accurate query (topic) model, while the term-based expansion is only marginally beneficial, as it introduces noise into the model.

6.2. Parameter Sensitivity Analysis

We analyze the sensitivity of our models with respect to their parameters. The strategy we employ is for each parameter we perform a sweep, while using the default settings for all others. The best individually found values are then put together and used in the optimized run, reported in Table IX. This method may not result in the overall best possible parameter settings; however, it is not our aim here to tweak and fine-tune parameters. Table VIII lists the actual parameter values used for our optimized models. All models use λ to determine the mixture of the term-based and category-based components, where a bigger value assigns more importance to the term-based component. The expanded models use λ^T to adjust the weight between the baseline and expanded term-based components, and λ^C to adjust the weight between the baseline and expanded category-based components. Only the blind feedback model uses N, a

Table VIII. Optimal Parameter Settings for the Runs Reported in Table IX

Model	Opt.	Data	λ	λ^T	λ^C	N	K^T	K^C
(M5) No expansion (ER)		XER2007	0.7	-	-	-	-	-
		XER2008	0.6	-	-	-	-	-
(M5-b) Expansion (blind, ER)	Ind.	XER2007	0.8	0.05	0.4	3	25	15
		XER2008	0.7	0.05	0.7	5	25	15
	Seq.	XER2007	0.7	0.5	0.4	3	25	10
		XER2008	0.7	0.5	0.5	5	25	5
(M5-e) Expansion (examples, LC)	Ind.	XER2007	0.6	0.6	0.8	-	15	5
		XER2008	0.5	0.6	0.8	-	25	5
	Seq.	XER2007	0.6	0.6	0.8	-	15	5
		XER2008	0.6	0.8	0.7	-	25	5

Table IX. Results Using Default Parameters vs. Parameters Optimized for MAP

		XER2007		XER2008	
Model	Parameters	MAP	MRR	xinfAP	MRR
Entity ranking					
(M5) No expansion	default	0.2554	0.4531	0.3124	0.5024
(M5) No expansion	optimized	0.2873$^{\triangle}$	**0.4648**	0.3156	0.5023
(M5-b) Expansion (blind)	default	0.2541	0.4090	0.3365	0.4984
(M5-b) Expansion (blind)	optimized (Ind.)	0.2854	0.4573	0.3267	**0.5245**
(M5-b) Expansion (blind)	optimized (Seq.)	0.2767$^{\triangle}$	0.4596	0.3452	0.5029
Best performing INEX run		**0.306**	—	**0.3809**	—
List completion					
(M5) No expansion	default	0.2202	0.4042	0.2729	0.4339
(M5) No expansion	optimized	0.2410	0.3997	0.2784	0.4693
(M5-e) Expansion (examples)	default	0.3267	0.5357	0.3926	0.6353
(M5-e) Expansion (examples)	optimized (Ind.)	**0.3446**	**0.6042**	**0.4072**	0.6835
(M5-e) Expansion (examples)	optimized (Seq.)	**0.3446**	**0.6042**	0.4048	**0.7089**
Best performing INEX run		0.309	—	0.402	—

Significance tested against default parameter setting. Best results for each are in boldface.

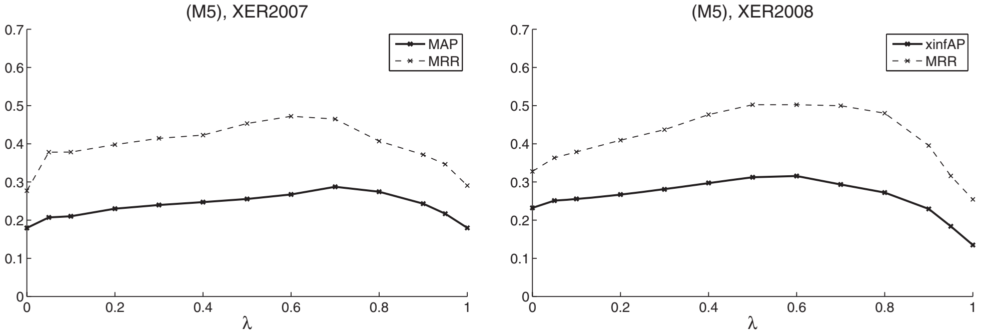

Fig. 12. The effect of varying λ (the weight of the term-based component) on the baseline entity ranking run (M5).

parameter that determines the number of feedback entities to use, as in the example-based case these entities are given. Both expanded models extract a number of terms (K^T) and a number of categories (K^C) from these entities for feedback.

6.2.1. Baseline. Our baseline, again, is produced by (M5). Runs without query expansion involve only one parameter, λ. Figure 12 shows MAP/xinfAP and MRR scores for different values of this parameter. On both data sets we observe a gradual increase in performance as the mixture of the components moves towards a balance ($\lambda = 0.6/0.7$). The fact that a balanced mixture achieves optimal results suggests that

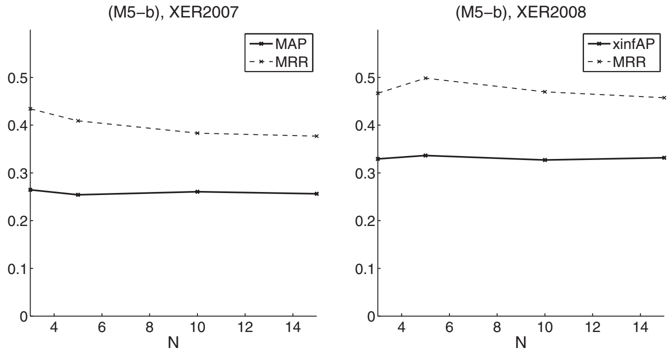

Fig. 13. The effect of varying the number of feedback entities, N, in case of blind feedback (M5-b). Expansion takes place in both the term-based and category-based components.

each component contributes relevant information to the query model. The best empirically found value is 0.7 and 0.6 for 2007 and 2008, respectively; this means that slightly more importance is assigned to the term-based representation over category-based (significant only for MAP, XER2007, when compared against the default setting, that is, 0.5).

6.2.2. Query Expansion. Switching to the feedback runs, we look at two specific types of expansion, both performed on top of the baseline (M5).

—(M5-b) expansion using blind feedback; we evaluate this model on the entity ranking task (cf. qT-E'+qC-E' in Table II).
—(M5-e) expansion using example entities; this model is tested on the list completion task (cf. qT-E+qC-E in Table II).

Note that for both (M5-b) and (M5-e), expansion takes place in both the term-based and category-based components.

First, we optimize λ in a similar fashion as was done for the baseline runs in Section 6.2.1. The best found values are reported in Table VIII; plots are not presented due to space considerations.

Next, we experiment with the number of feedback entities (N). This applies only to the blind feedback model (M5-b); when example entities are provided, we use them all for feedback. Figure 13 shows the results. We find the model to be insensitive to the number of feedback entities. We observe very limited variance in terms of performance when using different N values, and none of those differences are significant.

So far, expansion took place in both components (term- and category-based). To single out the effect of each, in the followings we look at only one component at a time (either term-based or category-based) while leaving the other unchanged (that is, no expansion is performed). We start with the term-based component. Figure 14 shows performance with respect to the number of feedback terms. We find that the models are very insensitive to the choice of this parameter, in terms of MAP/xinfAP, while MRR scores display some fluctuation. Differences between the best and worst performing settings are not significant. Figure 15 plots the effect of varying λ^T, the weight with which expansion terms are taken into account. Again, we see hardly any changes in terms of MAP/xinfAP. In the case of blind feedback (M5-b), highest MRR scores—for both years—are achieved when relying almost exclusively on the baseline model ($\lambda^T = 0.05$); the difference compared with the baseline setting ($\lambda^T = 0.5$) is significant for 2007.

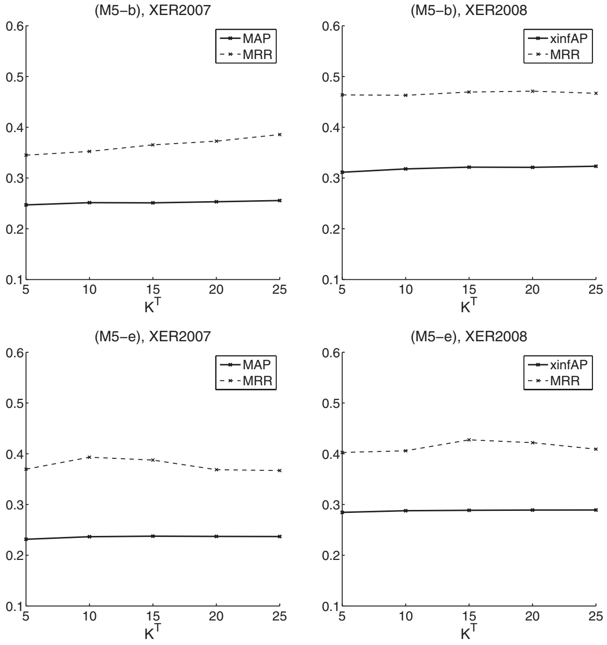

Fig. 14. The effect of varying K^T. Expansion takes place in the term-based component only. (Top) blind feedback (M5-b); (bottom) examples (M5-e).

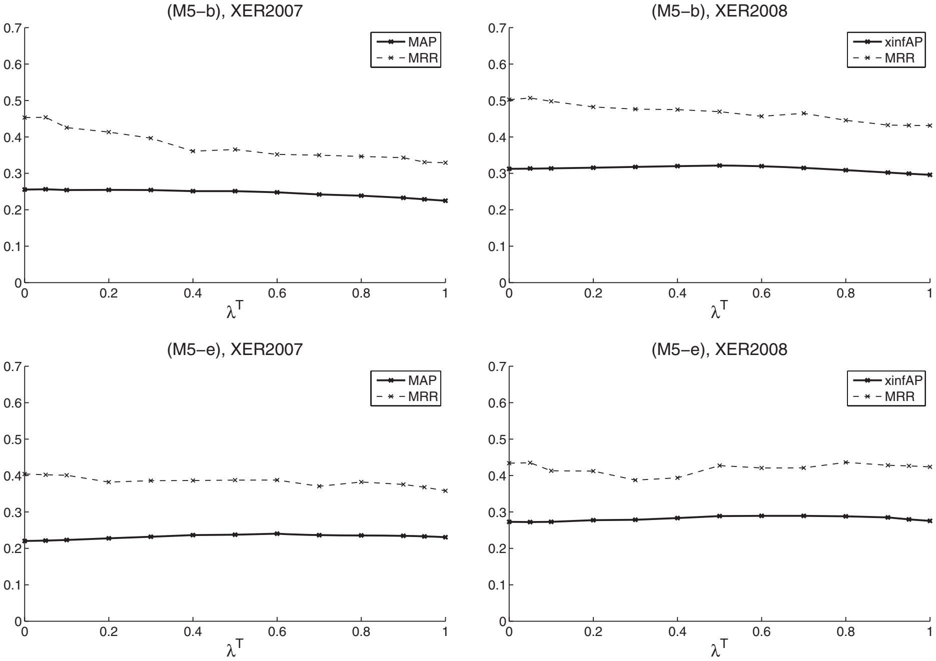

Fig. 15. The effect of varying λ^T. Expansion takes place in the term-based component only. (Top) blind feedback (M5-b); (bottom) examples (M5-e).

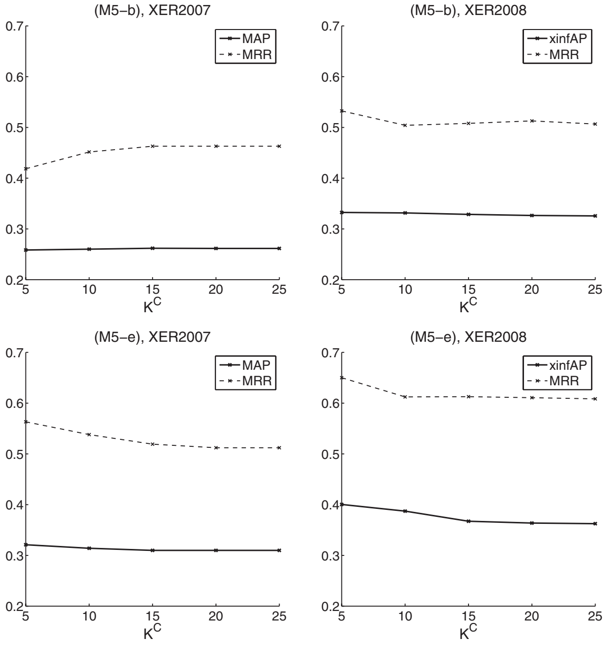

Fig. 16. The effect of varying K^C. Expansion takes place in the category-based component only. (Top) blind feedback (M5-b); (bottom) examples (M5-e).

Next, we perform a similar analysis for the category-based component. Figure 16 shows the number of expansion categories. All MAP/xinfAP curves are relatively flat; the only exception is (M5-e) on the 2008 topics. Here, using a small number of feedback categories ($K^C = 5$) delivers clearly the best performance. However, this is not significantly better than using any other value for this parameter. With one exception ((M5-b), XER2007) the highest MRR scores are achieved using a small number of feedback categories. But, again, none of the differences are significant. Finally, the effect of varying λ^C, the weight put on expansion categories, is shown in Figure 17. The curves are very flat in case of blind feedback (M5-b), with no significant differences. The example-based feedback model (M5-e) is much more interesting; best performance is achieved with an unbalanced mixture, with most of the weight assigned to the expanded component ($\lambda^C = 0.8$). This demonstrates that when feedback entities are indeed relevant ones, category-based feedback is very rewarding, as it contributes new information to the query model.

Overall, we can see that our models are very robust with respect to the number of feedback entities, expansion terms, and expansion categories. We observed only minor differences in performance when changing the weight of mixture models in the term-based case (Figure 15). This was also the case for category-based expansion, in case of blind feedback. When using examples, however, the expanded model did assign a lot of weight to the feedback categories ($\lambda^C = 0.8$). This indicates that the example entities generally introduce "good" target categories, and these are of more value to the model than terms.

6.2.3. Optimized Runs. Given the optimal parameter settings listed in Table VIII, we report on the performance of our models with those optimal parameter settings in Table IX. Since these parameters were obtained independently of each other, we refer to this setting as *Ind*. We also experimented with another optimization method, where

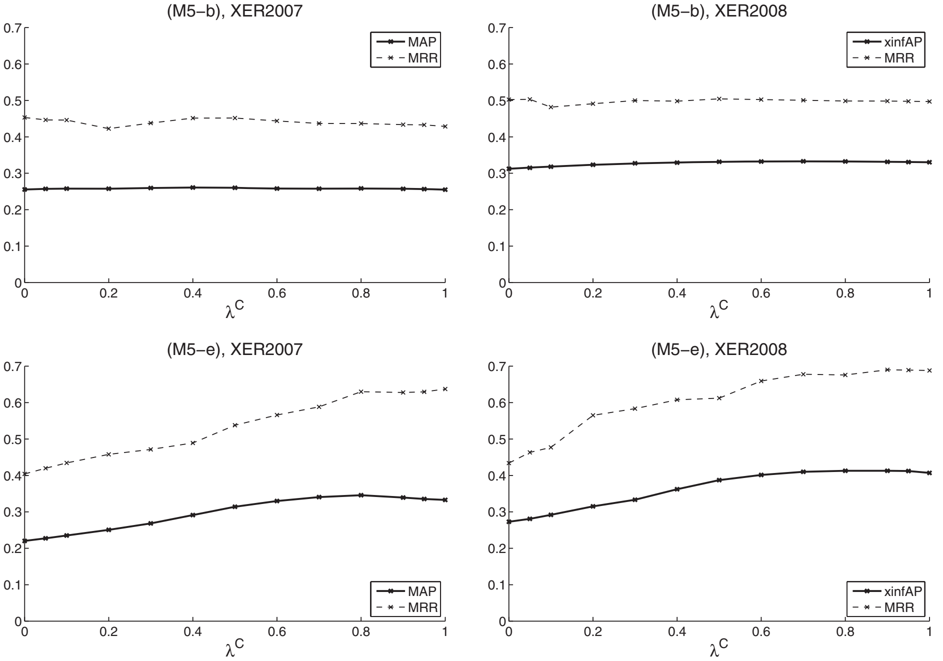

Fig. 17. The effect of varying λ^C. Expansion takes place in the category-based component only. (Top) blind feedback (M5-b); (bottom) examples (M5-e).

parameters were tuned sequentially (*Seq*): first, the number of feedback entities (N, only for (M5-b)); followed by the optimization of the term-based and category-based components separately (first, the number of feedback terms/categories, followed by the mixture weight); and finally the mixture parameter that controls the weight between the term- and category-based representations (λ). In the interest of space we report only the actual values, in Table VIII. While there are clear differences in absolute values, the optimized runs rarely manage to significantly outperform the default settings in terms of MAP.

To calibrate our scores with those obtained by others, we compare our scores against the best sores achieved at the 2007 and 2008 editions of the INEX Entity ranking track (see Table IX). For the entity ranking task, we see that the best performing runs beat our best runs. In 2007, this was a run that used random walks to model multi-step relevance propagation between linked entities [Tsikrika et al. 2008]; in 2008 it was a run that used topic difficulty prediction to dynamically set the values of retrieval parameters (the weight of LinkRank, category similarity, and full text scores) [Vercoustre et al. 2009]. For the list completion task, the outcomes are somewhat different: we match or even outperform the best performing runs in 2007 and 2008. In 2007, the best performing run used a method to exploit the locality of links around example entities, in addition to utilizing link structure and category information [Vercoustre et al. 2008]; in 2008, they expanded their approach with topic difficulty prediction [Vercoustre et al. 2009].

7. CONCLUSIONS

We have introduced a probabilistic framework for entity search. This framework allowed us to systematically explore ways of combining query and category information as well as example entities to create a query model. The framework also allowed us to transparently integrate term-based and category-based feedback information. We explored our models along many dimensions; experimental evaluations were performed using the 2007 and 2008 editions of the INEX Entity Ranking track.

We demonstrated the advantage of a category-based representation over a term-based representation for query modeling. We also showed the effectiveness of category-based feedback, which was found to outperform term-based feedback. The biggest improvements over a competitive baseline based on term- and category-based information were achieved when both term-based and category-based feedback are used with example entities (provided along with the textual keyword query). Our models were able to use the additional information provided by exploiting examples and categories in an effective manner, showing very competitive performance on both the entity ranking and list completion tasks on all available test sets.

In future work we plan to examine ways of automatically estimating parameters that are topic-dependent (that is, dependent on the query terms, and/or target categories, and/or example entities), devise methods for determining whether to apply feedback, and explore the potential of our models for the "related entity finding" task recently launched at TREC [Balog et al. 2010].

More generally, the effectiveness of the methods used in the paper suggests that we branch out in at least two directions. First, what if we try to automatically enrich queries by associating categories with them and then use these for entity retrieval purposes using the models of this paper? Meij et al. [2009] show that this association can be carried out with very high confidence. Second, in addition to example entities and categories, other types of enriched query input may be inferred from the user's interaction with a search engine in the context of entity retrieval, such as clicks and session information [Huurnink et al. 2010]. How can we incorporate these types of more noisy information in the models detailed in this article?

ACKNOWLEDGMENTS

We are extremely grateful to our anonymous reviewers; their comments and feedback really helped to improve the paper.

REFERENCES

Balog, K. 2008. People search in the enterprise. Ph.D. thesis, University of Amsterdam.

Balog, K., Azzopardi, L., and de Rijke, M. 2006. Formal models for expert finding in enterprise corpora. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, USA, 43–50.

Balog, K., Azzopardi, L., and de Rijke, M. 2009. A language modeling framework for expert finding. *Inform. Process. Manag. 45,* 1, 1–19.

Balog, K., Bron, M., and de Rijke, M. 2010. Category-based query modeling for entity search. In *Proceedings of the 32nd European Conference on Information Retrieval (ECIR)*. Springer, Berlin, 319–331.

Balog, K. and de Rijke, M. 2008. Associating people and documents. In *Proceedings of the 30th European Conference on Information Retrieval (ECIR)*. Springer, Berlin, 296–308.

Balog, K., de Vries, A. P., Serdyukov, P., Thomas, P., and Westerveld, T. 2010. Overview of the TREC 2009 entity track. In *Proceedings of the 18th Text REtrieval Conference (TREC)*. NIST.

Balog, K., Soboroff, I., Thomas, P., Craswell, N., de Vries, A. P., and Bailey, P. 2009. Overview of the TREC 2008 enterprise track. In *Proceedings of the 17th Text Retrieval Conference Proceedings (TREC)*. NIST.

Balog, K., Weerkamp, W., and de Rijke, M. 2008. A few examples go a long way: constructing query models from elaborate query formulations. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. ACM, New York, NY, 371–378.

Chu-Carroll, J., Czuba, K., Prager, J., Ittycheriah, A., and Blair-Goldensohn, S. 2004. IBM's PIQUANT II in TREC 2004. In *Proceedings of the 13th Text Retrieval Conference (TREC)*. NIST.

Conrad, J. G. and Utt, M. H. 1994. A system for discovering relationships by feature extraction from text databases. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. Springer Verlag, Berlin, 260–270.

CRASWELL, N., DEMARTINI, G., GAUGAZ, J., AND IOFCIU, T. 2009. L3S at INEX2008: Retrieving entities using structured information. Lecture Notes in Computer Science, vol. 5631, Springer-Verlag, Berlin, 253–263.

DE VRIES, A., VERCOUSTRE, A.-M., THOM, J. A., CRASWELL, N., AND LALMAS, M. 2008. Overview of the INEX 2007 entity ranking track. Lecture Notes in Computer Science, vol. 4862, Springer-Verlag, Berlin, 245–251.

DEMARTINI, G., DE VRIES, A., IOFCIU, T., AND ZHU, J. 2009. Overview of the INEX 2008 entity ranking track. Lecture Notes in Computer Science, vol. 5631, Springer-Verlag, Berlin, 243–252.

DEMARTINI, G., FIRAN, C. S., AND IOFCIU, T. 2008. L3S at INEX 2007: Query expansion for entity ranking using a highly accurate ontology. Lecture Notes in Computer Science, vol. 4862, Springer-Verlag, Berlin, 252–263.

FISSAHA ADAFRE, S., DE RIJKE, M., AND TJONG KIM SANG, E. 2007. Entity retrieval. In *Proceedings of the Conference on Recent Advances in Natural Language Processing (RANLP)*.

FUHR, N., KAMPS, J., LALMAS, M., AND TROTMAN, A., Eds. 2008. *Focused Access to XML documents: 6th International Workshop of the Initiative for the Evaluation of XML Retrieval (INEX)*. Lecture Notes in Computer Science, vol. 4862. Springer Verlag, Berlin.

GEVA, S., KAMPS, J., AND TROTMAN, A., Eds. 2009. *Advances in Focused Retrieval: 7th International Workshop of the Initiative for the Evaluation of XML Retrieval (INEX)*. Lecture Notes in Computer Science, vol. 5631. Springer-Verlag, Berlin.

GHAHRAMANI, Z. AND HELLER, K. 2006. Bayesian sets. In *Advances in Neural Information Processing Systems 18*, Y. Weiss, B. Schölkopf, and J. Platt, Eds. MIT Press, Cambridge, MA, 435–442.

GOOGLESETS. 2009. http://labs.google.com/sets (accessed 1/09.)

HUURNINK, B., HOLLINK, L., VAN DEN HEUVEL, W., AND DE RIJKE, M. 2010. Search behavior of media professionals at an audiovisual archive: A transaction log analysis. *J. Amer. Soc. Infor. Sci. Technol. 61,* 6, 1180–1197.

JÄMSEN, J., NÄPPILÄ, T., AND ARVOLA, P. 2008. Entity ranking based on category expansion. Lecture Notes in Computer Science, vol. 4862, Springer-Verlag, Berlin, 264–278.

JÄRVELIN, K., KEKÄLÄINEN, J., AND NIEMI, T. 2001. Expansiontool: Concept-based query expansion and construction. *Infor. Retrieval 4,* 3-4, 231–255.

JIANG, J., LIU, W., RONG, X., AND GAO, Y. 2009. Adapting language modeling methods for expert search to rank Wikipedia entities. Lecture Notes in Computer Science, vol. 5631, Springer-Verlag, Berlin, 264–272.

KAMPS, J. AND KOOLEN, M. 2008. The importance of link evidence in Wikipedia. In *Proceedings of the 30th European Conference on Information Retrieval (ECIR)*. Springer, 270–282.

KAMPS, J., MARX, M., DE RIJKE, M., AND SIGURBJÖRNSSON, B. 2006. Articulating information needs in XML query languages. *ACM Trans. Inf. Syst. 24,* 4, 407–436.

KAPTEIN, R. AND KAMPS, J. 2009. Finding entities in Wikipedia using links and categories. Lecture Notes in Computer Science, vol. 5631, Springer-Verlag, Berlin, 273–279.

KIM, J., XUE, X., AND CROFT, W. B. 2009. A probabilistic retrieval model for semistructured data. In *Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval*. Springer-Verlag, Berlin, 228–239.

KRAAIJ, W., WESTERVELD, T., AND HIEMSTRA, D. 2002. The importance of prior probabilities for entry page search. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, 27–34.

LAFFERTY, J. AND ZHAI, C. 2001. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, 111–119.

LAVRENKO, V. AND CROFT, W. B. 2001. Relevance based language models. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, 120–127.

LOSADA, D. AND AZZOPARDI, L. 2008. An analysis on document length retrieval trends in language modeling smoothing. *Infor. Retrieval 11,* 2, 109–138.

MEIJ, E., BRON, M., HUURNINK, B., HOLLINK, L., AND DE RIJKE, M. 2009. Learning semantic query suggestions. In *Proceedings of the 8th International Semantic Web Conference (ISWC)*. Springer, Berlin.

MEIJ, E. AND DE RIJKE, M. 2007. Thesaurus-based feedback to support mixed search and browsing environments. In *Proceedings of the 11th European Conference on Research and Advanced Technology for Digital Libraries*. Springer, Berlin.

METZLER, D. AND CROFT, W. B. 2005. A Markov random field model for term dependencies. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, 472–479.

MISHNE, G. AND DE RIJKE, M. 2005. Boosting web retrieval through query operations. In *Proceedings of 27th European Conference on IR Research (ECIR)*. D. Losada and J. Fernández-Luna, Eds. Springer, Berlin, 502–516.

MISHNE, G. AND DE RIJKE, M. 2006. A study of blog search. In *Proceedings of the 28th European Conference on IR Research (ECIR)*. M. Lalmas, A. MacFarlane, S. Rüger, A. Tombros, T. Tsikrika, and A. Yavlinsky, Eds. LNCS Series, vol. 3936. Springer, Berlin, 289–301.

MURUGESHAN, M. S. AND MUKHERJEE, S. 2008. An n-gram and initial description based approach for entity ranking track. Lecture Notes in Computer Science, vol. 4862, Springer-Verlag, Berlin, 293–305.

PEHCEVSKI, J., VERCOUSTRE, A.-M., AND THOM, J. A. 2008. Exploiting locality of Wikipedia links in entity ranking. In *Proceedings of the 30th European Conference on Information Retrieval (ECIR)*. Springer, Berlin, 258–269.

PETKOVA, D. AND CROFT, W. B. 2007. Proximity-based document representation for named entity retrieval. In *Proceedings of the 16th ACM Conference on Information and Knowledge Management*. ACM, New York, NY, 731–740.

RAGHAVAN, H., ALLAN, J., AND MCCALLUM, A. 2004. An exploration of entity models, collective classification and relation description. In *Proceedings of the ACM SIGKDD Workshop on Link Analysis and Group Detection (LinkKDD)*. ACM, New York, NY.

ROSE, D. E. AND LEVINSON, D. 2004. Understanding user goals in web search. In *Proceedings of the 13th International Conference on World Wide Web*. ACM, New York, NY, 13–19.

SAYYADIAN, M., SHAKERY, A., DOAN, A., AND ZHAI, C. 2004. Toward entity retrieval over structured and text data. In *Proceedings of the ACM SIGIR Workshop on the Integration of Information Retrieval and Databases (WIRD)*. ACM, New York, NY.

SERDYUKOV, P. AND HIEMSTRA, D. 2008. Being omnipresent to be almighty: The importance of the global web evidence for organizational expert finding. In *Proceedings of the SIGIR Workshop on Future Challenges in Expertise Retrieval (fCHER)*. ACM, New York, NY, 17–24.

SONG, F. AND CROFT, W. B. 1999. A general language model for information retrieval. In *Proceedings of the 18th International Conference on Information and Knowledge Management*. ACM, New York, NY, 316–321.

SUCHANEK, F. M., KASNECI, G., AND WEIKUM, G. 2007. YAGO: a core of semantic knowledge unifying WordNet and Wikipedia. In *Proceedings of the 16th International World Wide Web Conference*. 697–706.

TAO, T. AND ZHAI, C. 2006. Regularized estimation of mixture models for robust pseudo-relevance feedback. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, 162–169.

THOM, J., PEHCEVSKI, J., AND VERCOUSTRE, A.-M. 2007. Use of Wikipedia categories in entity ranking. In *Proceedings of the 12th Australasian Document Computing Symposium (ADCS)*.

TSIKRIKA, T., SERDYUKOV, P., RODE, H., WESTERVELD, T., ALY, R., HIEMSTRA, D., AND DE VRIES, A. P. 2008. Structured document retrieval, multimedia retrieval, and entity ranking using PF/Tijah. Lecture Notes in Computer Science, vol. 4862, Springer-Verlag, Berlin, 306–320.

VERCOUSTRE, A.-M., PEHCEVSKI, J., AND NAUMOVSKI, V. 2009. Topic difficulty prediction in entity ranking. Lecture Notes in Computer Science, vol. 5631, Springer-Verlag, Berlin, 280–291.

VERCOUSTRE, A.-M., PEHCEVSKI, J., AND THOM, J. A. 2008. Using Wikipedia categories and links in entity ranking. Lecture Notes in Computer Science, vol. 4862, Springer-Verlag, Berlin, 321–335.

VERCOUSTRE, A.-M., THOM, J., AND PEHCEVSKI, J. 2007. Entity ranking in Wikipedia. Res. rep. RR-6294, INRIA.

VERCOUSTRE, A.-M., THOM, J. A., AND PEHCEVSKI, J. 2008. Entity ranking in Wikipedia. In *Proceedings of the ACM Symposium on Applied Computing (SAC)*. ACM, New York, NY, 1101–1106.

VOORHEES, E. 2005. Overview of the TREC 2004 question answering track. In *Proceedings of the 13th Text Retrieval Conference (TREC)*. NIST, Special Publication SP 500-261.

WEERKAMP, W., BALOG, K., AND MEIJ, E. 2009. A generative language modeling approach for ranking entities. Lecture Notes in Computer Science, vol. 5631, Springer-Verlag, Berlin, 292–299.

YILMAZ, E., KANOULAS, E., AND ASLAM, J. A. 2008. A simple and efficient sampling method for estimating AP and NDCG. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, 603–610.

ZARAGOZA, H., RODE, H., MIKA, P., ATSERIAS, J., CIARAMITA, M., AND ATTARDI, G. 2007. Ranking very many typed entities on Wikipedia. In *Proceedings of the 16th ACM Conference on Conference on Information and Knowledge Management*. ACM, New York, NY, 1015–1018.

ZHAI, C. AND LAFFERTY, J. 2004. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Infor. Syst. 22,* 2, 179–214.

ZHU, J., HUANG, X., SONG, D., AND RÜGER, S. 2009. Integrating multiple document features in language models for expert finding. In *Knowledge and Information Systems*. DOI 10.1007/s10115-009-0202-6.

ZHU, J., SONG, D., AND RÜGER, S. 2008. Integrating document features for entity ranking. Lecture Notes in Computer Science, vol. 4862, Springer-Verlag, Berlin, 336–347.

ZHU, J., SONG, D., RÜGER, S. M., EISENSTADT, M., AND MOTTA, E. 2006. The Open University at TREC 2006 Enterprise Track Expert Search Task. In *Proceedings of the 15th Text REtrieval Conference (TREC)*. NIST. Special Publication 500-272.

Received April 2010; revised October 2010, April 2011; accepted September 2011