Category-based Query Modeling for Entity Search

Krisztian Balog, Marc Bron, and Maarten de Rijke

ISLA, University of Amsterdam, Science Park 107, 1098 XG Amsterdam, The Netherlands
k.balog@uva.nl, m.m.bron@uva.nl, derijke@uva.nl

Abstract. Users often search for entities instead of documents and in this setting are willing to provide extra input, in addition to a query, such as category information and example entities. We propose a general probabilistic framework for entity search to evaluate and provide insight 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.

1 Introduction

Users often search for specific entities instead of documents mentioning them [8, 20]. Example information needs include "Impressionist art museums in The Netherlands" or "Experts on authoring tools," where answers to be returned are museums and experts and 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 [3]. These additional means may include categories to which 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 [8, 9]. Here, entities are represented by their Wikipedia page and queries asking for an entity are typed (asking for entities belonging to certain categories) and may come with examples. Two tasks are being considered at INEX: (1) *entity ranking*, where a query and target categories are given, and (2) *list completion*, where a 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 (query, categories, examples), the obvious system-oriented question to ask is how to map these ingredients into the query component of a retrieval model. In this paper, we focus on effectively capturing and exploiting category-based information. Several approaches to incorporating such information have been proposed (see §2 below), but there is no systematic account of approaches yet.

We introduce a probabilistic framework for entity retrieval that models category information in a theoretically transparent manner. Information needs and entities are represented as a tuple: 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. Our focus is on two core steps: query modeling and query model expansion.

We seek to answer the following questions. Does our two-component query model improve over single component approaches (either term-based or category-based)? 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 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 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 §2 we discuss related work. We introduce our retrieval model in §3. In §4 we zoom in on query modeling and query model expansion; §5 is devoted to an experimental evaluation. An analysis and conclusion complete the paper.

2 Related Work

We review work on entity-oriented search tasks and then consider work on entity ranking at INEX.

2.1 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, ZoomInfo, Evri, and the Yahoo! correlator demo.[1] They differ in their data sources, supported entity types, functionality, and user interface. Common to them, however, is their ability to rank entities with respect to a topic or another entity.

Conrad and Utt [6] 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 co-occurrence 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. [21] re-state this approach in a language modeling framework. Sayyadian et al. [23] introduce the problem of finding missing information about a real-world entity from text and structured data; entity retrieval over text documents can be significantly aided by 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 constraints) [29]. To answer list questions [22], 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 [5]. Google Sets allows users to enter some instances of a concept and retrieve others that closely match the examples provided [14]. Ghahramani and Heller [13] developed an algorithm for completing a list based on examples using machine learning techniques.

[1] See http://www.textmap.com/, http://www.zoominfo.com/ http://www.evri.com/, and http://sandbox.yahoo.com/correlator, respectively.

Entity search generalizes *expert finding*: given a topic, return a ranked list of experts on the topic. TREC 2005–2008 featured an expert finding track [4]. Lessons learned involve models, algorithms, and evaluation methodology [1, 2].

Zaragoza et al. [32] retrieve 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) [8, 9] introduced tasks where candidate items are restricted to have their own Wikipedia page. Fissaha Adafre et al. [10] addressed an early version of the entity ranking and list completion tasks and explored different representations of list descriptions and example entities.

2.2 Entity ranking at INEX

INEX has traditionally focused on the use of document structure to improve retrieval effectiveness. While the initial focus was on document and element retrieval, over the years, INEX has expanded. In recent years, INEX has mainly been using Wikipedia as its document collection. The main lesson learned is that exploiting the rich structure of the collection (text plus category information, associations between entities, and query-dependent link structure) helps improve retrieval performance [8].

As to query formulation for entity retrieval, usually stemming and stopwording is performed. Craswell et al. [7] go beyond this and modify the query with NLP techniques, removing verbs while focussing on adjectives, nouns and named entities; others use the query to retrieve a set of documents and use co-occurrence statistics between retrieved article titles and query terms as a component in their ranking score [34].

INEX participants have considered expanding queries using category information. E.g., the latter has been used for expanding the query with labels of the target category [7]. Others have used set-based metrics to measure the overlap between target categories and categories of candidate entities [27, 30]. Target category information provided as part of the query need not be complete, as manual assignment of categories to documents is imperfect. Some teams expand the target categories, e.g., by using the category structure to expand with categories [15, 25, 30]. Others expand the target categories using lexical similarity between category labels and query terms [17, 26].

A standard way of combining category and term-based components is to use a language modeling approach and to estimate the probability of an entity given the query and category information [16, 30, 34]. Vercoustre et al. [28] integrate query difficulty prediction and Zhu et al. [34] treat categories as metadata fields and apply a multi-field retrieval model. Several participants in the list completion task use the categories of example entities for category expansion, using various expansion techniques [7, 16, 26, 30, 34]; some use category information to expand the term-based model [30].

3 Modeling Entity Ranking

We present a general retrieval scheme for two entity ranking tasks. In the *entity ranking* task one is given a "standard" keyword query (Q) along with a set of target categories (C) and has to return entities. For *list completion* we need to return entities given a keyword query (Q), a set of target categories (C) and a set of similar entities (E).

3.1 A general scheme for entity ranking

Fig. 1 depicts our scheme for ranking entities. The process goes as follows. We are given the user's input, consisting of a query, a set of input categories, and, optionally, example

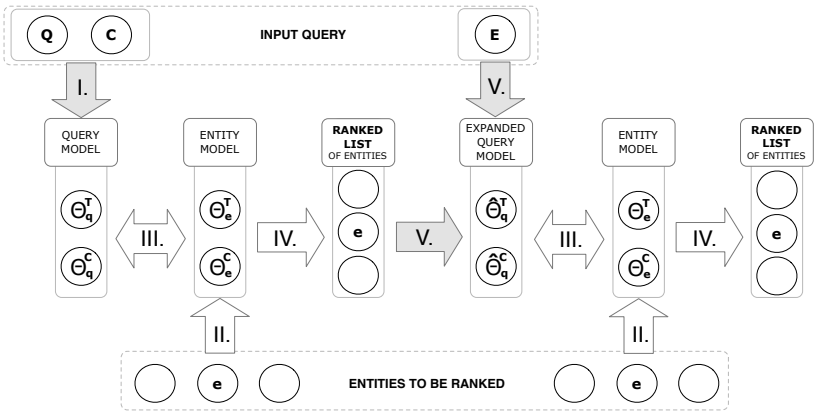

Fig. 1. A general scheme for entity ranking. The steps on which we focus are indicated with grey arrows; all steps are explained in §3.1.

entities. This input is translated into a query model, with a term-based and/or a category-based component (I). During retrieval (Step III) this model is compared against models created for indexed entities (derived in Step II). In Step IV a ranked list of entities is produced (based on Step III), which, in turn, may (optionally) give rise to an expanded query model (Step V)—from that point onwards Steps III, IV, V can be repeated.

Our focus is limited to the problem of modeling the query: (i) How can the user's input be translated into an initial query model (Step I)? And (ii) how can this—often sparse—representation be refined or extended to better express the underlying information need (Step V)? Specifically, we are interested in sources and components that play a role in estimating the term- and category-based representations of query models. Some 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 approach, as this concerns the ranking part (II, III and IV), and this information is not (explicitly) present on the query side (I and V).

3.2 A probabilistic model for entity ranking (Steps III and IV)

We introduce a probabilistic framework that implements the entity ranking approach depicted in Fig. 1. We rank entities according to their probability of being relevant given the query: $P(e|q)$. We apply Bayes' rule and rewrite this probability to:

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

where $P(q|e)$ expresses the probability that query q is generated by entity e, and $P(e)$ is the *a priori* probability of e being relevant, i.e., the entity prior. We assume that $P(e)$ is uniform, thus, does not affect the ranking.

Each entity is represented as a pair $\theta_e = (\theta_e^T, \theta_e^C)$, where θ_e^T is a distribution over terms and θ_e^C is a distribution over categories. The query is also represented as a pair: $\theta_q = (\theta_q^T, \theta_q^C)$, which is then (optionally) further refined, resulting in an expanded query model $\hat{\theta}_q = (\hat{\theta}_q^T, \hat{\theta}_q^C)$ that is used for ranking entities.

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

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

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 two distributions. Here, we opt for the Kullback-Leibler divergence. The term-based similarity is estimated as follows:

$$KL(\theta_q^T||\theta_e^T) = \sum_t P(t|\theta_q^T) \cdot \log \frac{P(t|\theta_q^T)}{P(t|\theta_e^T)}, \tag{3}$$

where $P(t|\theta_e^T)$ and $P(t|\theta_q^T)$ remain to be defined. Since KL divergence is a score (which is lower when two distributions are more similar), we turn it into a probability using Eq. 4:

$$P(\theta_q^T|\theta_e^T) \propto \max KL(\theta_q^T||\cdot) - KL(\theta_q^T||\theta_e^T). \tag{4}$$

The category-based component of the mixture in Eq. 2 is calculated analogously to the term-based case. Consequently, $P(c|\theta_e^C)$ and $P(c|\theta_q^C)$ remain to be defined.

This completes Step III. Next, we describe the entity model component, i.e., Step IV. Steps I and V are discussed in §4.

3.3 Entity modeling (Step IV)

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 [33, 19] and set:

$$P(t|\theta_e^T) = \frac{n(t,e)+\mu^T \cdot P(t)}{\sum_t n(t,e)+\mu^T}, \tag{5}$$

where $n(t,e)$ is the number of occurences of t in e, $\sum_t n(t,e)$ is the total number of term occurrences, i.e., the document length, and $P(t)$ is the background model (the relative frequency of t in the collection). 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 [24, 18]. Alternatively, the entity model can be expanded with terms from related entities [10].

Category-based representation Analogously to the term-based case, we smooth the maximum-likelihood estimate with a background model. We employ Dirichlet smoothing, and use the parameter μ^C to avoid confusion with μ^T:

$$P(c|\theta_e^C) = \frac{n(c,e)+\mu^C \cdot P(c)}{\sum_c n(c,e)+\mu^C}. \tag{6}$$

In Eq. 6, $n(c,e)$ 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 and is set using a maximum-likelihood estimate:

$$P(c) = \frac{\sum_e n(c,e)}{\sum_c \sum_e n(c,e)}, \tag{7}$$

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

4 Estimating and Expanding Query Models

In this section we introduce methods for estimating and expanding query models: Steps I and V in Figure 1. We construct initial (θ_q) and expanded ($\hat{\theta}_q$) query models, which boils down to estimating the probabilities $P(t|\theta_q^T)$, $P(c|\theta_q^C)$, $P(t|\hat{\theta}_q^T)$, and $P(c|\hat{\theta}_q^C)$) as listed in Fig. 1 and discussed in §3.

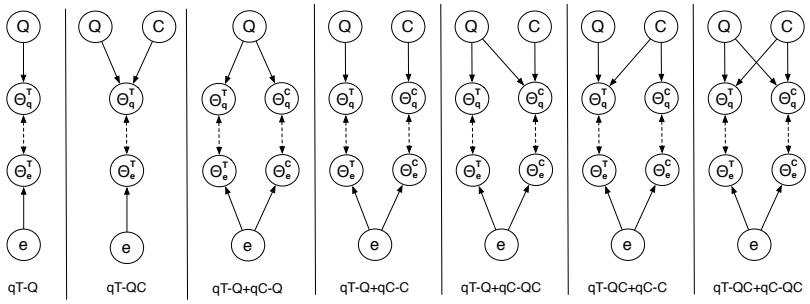

Fig. 2. Query models without expansion; Q stands for the topic title, E for example entities and C for the target categories; solid arrows from input to query model indicate the input is used to create the model; dashed arrows indicate a comparison between models; the models and acronyms are explained in §4.1.

4.1 Query models (Step I)

We define a series of query models, each consisting of a term-based component and/or a category-based component; graphical depictions of the models are given in Figure 2.

qT-Q This query model only has a term-based component and uses no category information (i.e., it amounts to standard language modeling for document retrieval). Writing $n(t, Q)$ for the number of times term t is present in query Q, we put

$$P(t|\theta_q^T) = P_{bl}(t|\theta_q^T) = \frac{n(t,Q)}{\sum_t n(t,Q)}. \tag{8}$$

qT-QC This model uses the names of input categories added to the term-based query model (θ_q^T); for the sake of simplicity, original terms and terms from category names contribute with the same weight to the total probability mass ($\alpha^T = 0.5$).

$$P(t|\theta_q^T) = \alpha^T \cdot P_{bl}(t|\theta_q^T) + (1 - \alpha^T) \cdot \frac{\sum_{c \in C} n(t,c)}{\sum_{c \in C} \sum_t n(t,c)}. \tag{9}$$

qT-Q+qC-Q This 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, i.e., 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.

$$P(c|\theta_q^C) = P_q(c|\theta_q^C) = \begin{cases} P(Q|c) / \sum_{c \in C} P(Q|c), & \text{if } c \in \text{top } N_c \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

qT-Q+qC-C This model uses 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

$$P(c|\theta_q^C) = P_{bl}(c|\theta_q^C) = \frac{n(c,q)}{\sum_c n(c,q)}. \tag{11}$$

qT-Q+qC-QC This model combines qT-Q+qC-C with categories relevant to the query added to the category-based query model; again, to keep things simple we allocate the probability mass equally between the two components ($\alpha^C = 0.5$).

$$P(c|\theta_q^C) = \alpha^C \cdot P_{bl}(c|\theta_q^C) + (1 - \alpha^C) \cdot P_q(c|\theta_q^C). \tag{12}$$

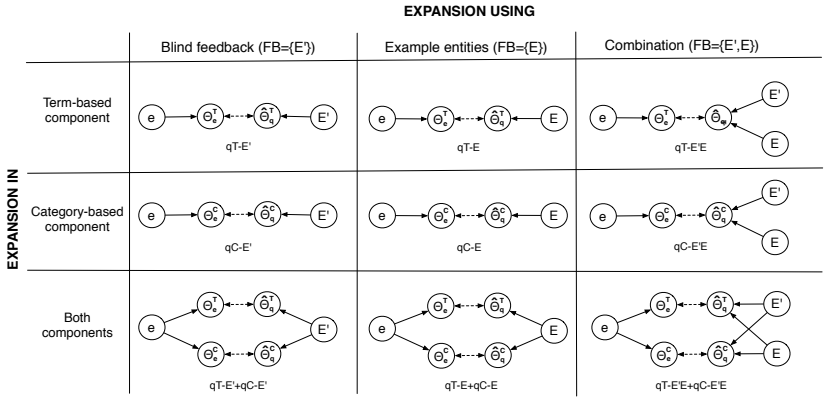

Fig. 3. Models with expansion; same graphical and notational conventions as in Fig. 2; acronyms explained in §4.2.

qT-QC+qC-C This model combines qT-Q+qC-C with names of input categories added to qT-Q (and contributes half of the probability mass).

qT-QC+qC-QC This model combines qT-Q+qC-QC and qT-QC+qC-C.

4.2 Expanded query models (Step V)

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

$$P(t|\hat{\theta}_q^T) = \lambda^T \cdot P_{ex}(t|\theta_q^T) + (1-\lambda^T) \cdot P_{bl}(t|\theta_q^T) \tag{13}$$

for the term-based component. And for the category-based component we set:

$$P(c|\hat{\theta}_q^C) = \lambda^C \cdot P_{ex}(c|\theta_q^C) + (1-\lambda^C) \cdot P_{bl}(c|\theta_q^C). \tag{14}$$

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 9 variations, shown in Fig. 3. Next, we construct expanded query models from a set of feedback entities FB.

Term-based expansion Given a set of feedback entities FB, the expanded query model is constructed as follows:

$$P(t|\hat{\theta}_q^T) = \frac{P_{K_T}(t|FB)}{\sum_{t'} P_{K_T}(t'|FB)}, \tag{15}$$

where $P_{K_T}(t|FB)$ denotes the top K_T terms with the highest $P(t|FB)$ value, calculated according to Eq. 16.

$$P(t|FB) = \frac{1}{|FB|} \sum_{e \in FB} \frac{n(t,e)}{\sum_t n(t,e)} \tag{16}$$

where $\sum_t n(t,e)$ is the total number of terms, i.e., the length of the document corresponding to e. (This is [3]'s best performing query model generation method using example documents, with all feedback documents assumed to be equally important.)

Category-based expansion Here we put

$$P(c|\hat{\theta}_q^C) = \frac{P_{K_C}(c|FB)}{\sum_{c'} P_{K_C}(c'|FB)}, \tag{17}$$

where $P_{K_C}(c|FB)$ denotes the top K_C categories with the highest $P(c|FB)$ value, calculated according to Eq. 18, (where, as before, $n(c,e)$ is 1 if e belongs to c):

$$P(c|FB) = \frac{1}{|FB|} \sum_{e \in FB} \frac{n(c,e)}{\sum_t n(c,e)}. \tag{18}$$

5 Experimental Evaluation

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

5.1 Experimental setup

Test Collection We use test sets of the INEX Entity Ranking track (INEX-XER) [8, 9], that 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: assignments of categories to articles are not always consistent [8].

Tasks INEX-XER has two tasks: *entity ranking* and *list completion*. An entity ranking topic specifies a keyword query (Q) and target categories (C). In the list completion task, the topic also specifies example entities (E) in addition. We also consider a variation of the task where all three input sources (Q, C, and E) are provided by the user.

Topics and judgments Two sets of topics are available for INEX-XER. For XER2007 a test set of 46 topics was created for the entity ranking track, 25 of which were specifically developed and assessed by track participants. For XER2008, 35 topics were developed and assessed by track participants [8]. We report on Mean Average Precision (MAP) and Mean Reciprocal Rank (MRR) for XER2007. For XER2008, xinfAP replaces MAP [31] as a better estimate of AP in the case of incomplete assessments [9].

Parameter settings Our models involve a number of parameters. In this section we apply our baseline settings for these parameters and we use them for all models. 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, i.e., $\lambda = \lambda^T = \lambda^C = 0.5$. In §5.4 we briefly analyze the sensitivity of our models w.r.t. these parameters.

5.2 The performance of query models

We examine the effectiveness of our query models and of the use of two components—for terms and categories—for the *entity ranking* task. In the experiments that involve the keyword query in the construction of the category-component of the query model, we

Model	λ	θ_q^T	θ_q^C	XER2007 MAP	XER2007 MRR	XER2008 xinfAP	XER2008 MRR
(1) qT-Q	1.0	Eq. 8	-	0.1798	0.2906	0.1348	0.2543
(2) qT-QC	1.0	Eq. 9	-	0.1706	0.3029	0.1259	0.2931
(3) qT-Q+qC-Q	0.5	Eq. 8	Eq. 10	0.2410	0.3830	0.1977	0.3190
(4) qT-Q+qC-C	0.5	Eq. 8	Eq. 11	0.2162	0.4168	0.3099	0.4783
(5) qT-Q+qC-QC	0.5	Eq. 8	Eq. 12	**0.2554**	**0.4531**	**0.3124**	**0.5024**
(6) qT-QC+qC-C	0.5	Eq. 9	Eq. 11	0.1881	0.2948	0.2911	0.4439
(7) qT-QC+qC-QC	0.5	Eq. 9	Eq. 12	0.2255	0.3346	0.2950	0.4357

Table 1. Entity ranking results, no expansion. Best results per collection in boldface.

set $N_c = 10$ (see Eq. 10). Table 1 list the results for the query models defined in §4.1, using the default parameter settings detailed in §5.1; in §5.4 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 $p = 0.05$. Simply flattening the target category information and adding category names as terms to the term component is not an effective strategy; see (1) vs. (2), (4) vs. (6), and (5) vs. (7). When we consider category-based information provided with the input query as a separate component, we see improvements across the test sets: see (1) vs. (3) (significant for 2007 and 2008) and (2) vs. (6) (significant for 2008). The switch from using only the keyword query for the construction of the category component to using target categories defined explicitly by the user ((3) vs. (4)) 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 ((4) vs. (5) (significant on the 2007 set) and (6) vs. (7)) leads to consistent improvements for all tasks and measures.

5.3 The performance of expanded query models

We report on the effectiveness of the expansion models depicted in Fig. 3. 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, i.e., the top N obtained using methods detailed §5.1; and E denotes a set of example entities. When we use example entities for expansion, we need to remove them from the runs, i.e., use the list completion qrels: 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, i.e., when we use only 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 2 presents the results of query expansion, applied on top of the best performing run from §5.1 (qT-Q+qC-QC). Category-based feedback helps more than term-based feedback; term-based blind feedback does not lead to significant improvements and hurts in one case (entity ranking, 2007, MAP); category-based expansion improves in terms of MAP scores on both tasks, in both years (significantly in 2008). For list completion, category-based feedback using examples leads to the biggest improvements (both years, both measures); relative improvements can be up to +47% in MAP (2007) and +50% in MRR (2008). Blind feedback plus examples improves over blind feedback alone, but is outperformed by category-based feedback using examples.

Model	FB	$\hat{\theta}_q^T$	$\hat{\theta}_q^C$	XER2007 MAP	XER2007 MRR	XER2008 xinfAP	XER2008 MRR
Entity ranking (blind feedback only)							
BASELINE (no expansion)	-	-	-	0.2554	**0.4531**	0.3124	0.5024
qT-E'	$\{E'\}$	Eq. 15	-	0.2511	$0.3654^{\triangledown}$	0.3214	0.4694
qC-E'	$\{E'\}$	-	Eq. 17	**0.2590**	0.4516	$0.3317^{\triangle}$	**0.5042**
qT-E'+qC-E'	$\{E'\}$	Eq. 15	Eq. 17	0.2536	0.4144	**0.3369**$^{\triangle}$	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. 15	-	0.2449	0.3818	0.2814	0.4139
qC-E'	$\{E'\}$	-	Eq. 17	0.2251	0.3858	$0.2970^{\triangle}$	0.4777
qT-E'+qC-E'	$\{E'\}$	Eq. 15	Eq. 17	0.2188	0.3576	$0.3022^{\triangle}$	0.4768
Examples							
qT-E	$\{E\}$	Eq. 15	-	$0.2376^{\triangle}$	0.3875	0.2886	0.4274
qC-E	$\{E\}$	-	Eq. 17	$0.3139^{\triangle}$	**0.5380**$^{\triangle}$	$0.3750^{\triangle}$	$0.6127^{\triangle}$
qT-E+qC-E	$\{E\}$	Eq. 15	Eq. 17	**0.3254**$^{\triangle}$	$0.5357^{\triangle}$	**0.3827**$^{\triangle}$	**0.6526**$^{\triangle}$
Blind feedback plus examples							
qT-E'E	$\{E', E\}$	Eq. 15	-	0.2200	$0.3193^{\triangledown}$	0.2843	0.4036
qC-E'E	$\{E', E\}$	-	Eq. 17	$0.2563^{\triangle}$	0.4421	$0.3299^{\triangle}$	0.5048
qT-E'E+qC-E'E	$\{E', E\}$	Eq. 15	Eq. 17	$0.2474^{\triangle}$	0.3961	$0.3319^{\triangle}$	0.4701

Table 2. Results, with expansion. Best results in boldface. Baseline corresponds to model (5) in Table 1. Significance differences with baseline denoted with $^{\triangle}$and $^{\triangledown}$.

5.4 Analysis

We briefly analyze the sensitivity of our models w.r.t. their parameters. For each parameter we perform a sweep,[2] 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 3. 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. Runs without query expansion involve only one parameter, λ; the best empirically found value is 0.7. The feedback runs are insensitive to the number of feedback terms; we use $K_T = 15$, as before, however, more weight ($\lambda^T = 0.7$) is given to the expanded (term-based) query model. For the blind feedback runs, using a small number of feedback entities ($N = 3$) performs best. When example entities are given, relying heavily ($\lambda^C = 0.8$) on a small number of feedback categories ($K_C = 3$) performs best; for blind feedback a conservative strategy pays off ($K_C = 10, \lambda^C = 0.6$).

6 Conclusions

We have introduced a probabilistic framework for entity search. The framework allows us to systematically explore combinations of query and category information as well as example entities to create query models. It also allows us to integrate term-based and category-based feedback information. We explored our models along many dimensions; experimental evaluation was performed using the 2007 and 2008 editions of the

[2] The sweep for mixture model parameters is performed in 0.1 steps in the $[0 \ldots 1]$ range; for the number feedback entities/terms/categories we use values $\{3, 5, 10, 15, 20, 25\}$.

Model	XER2007 MAP	XER2007 MRR	XER2008 xinfAP	XER2008 MRR
Entity ranking				
No expansion, default parameters	0.2554	0.4531	0.3124	0.5024
No expansion, λ optimized	0.2873$^{\Delta}$	0.4648	0.3156	0.5023
Expansion (blind feedback), default parameters	0.2590	0.4516	0.3369	0.4984
Expansion (blind feedback), optimized parameters	**0.3863**$^{\Delta}$	**0.6509**$^{\Delta}$	**0.3703**	**0.5849**
Best performing INEX run	0.306	-	0.341	-
List completion				
No expansion, default parameters	0.2202	0.4042	0.2729	0.4339
No expansion, λ optimized	0.2410	0.3997	0.2784	0.4693
Expansion (using examples), default parameters	0.3254	0.5357	0.3827	0.6526
Expansion (using examples), optimized parameters	**0.3384**	**0.5909**	**0.4182**$^{\Delta}$	**0.7041**
Best performing INEX run	0.309	-	0.402	-

Table 3. Results, using default parameters vs. parameters optimized for MAP. Significance tested against default parameter setting. Best results for each are in boldface.

INEX Entity Ranking track. We demonstrated the advantage of a category-based representation over a term-based one 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 category-based feedback is used with example entities (provided along with the keyword query). State-of-the-art performance was achieved on the entity ranking and list completion tasks on all available test sets. In future work we plan a more detailed result analysis than we were able to include here and to examine ways of automatically estimating parameters that are topic dependent (i.e., dependent on the query terms, and/or target categories and/or example entities).

Acknowledgments This research was supported by the DAESO and DuOMAn projects carried out within the STEVIN programme which is funded by the Dutch and Flemish Governments under project numbers STE-05-24 and STE-09-12, and by the Netherlands Organisation for Scientific Research (NWO) under project numbers 640.001.501, 640.002.501, 612.066.512, 612.061.814, 612.061.815, 640.004.802.

References

[1] K. Balog. *People Search in the Enterprise*. PhD thesis, University of Amsterdam, 2008.

[2] K. Balog, L. Azzopardi, and M. de Rijke. Formal models for expert finding in enterprise corpora. In *SIGIR '06*, pages 43–50, 2006.

[3] K. Balog, W. Weerkamp, and M. de Rijke. A few examples go a long way. In *SIGIR'08*, pages 371–378, 2008.

[4] K. Balog, I. Soboroff, P. Thomas, N. Craswell, A. P. de Vries, and P. Bailey. Overview of the TREC 2008 enterprise track. In *TREC 2008*. NIST, 2009.

[5] J. Chu-Carroll, K. Czuba, J. Prager, A. Ittycheriah, and S. Blair-Goldensohn. IBM's PIQUANT II in TREC 2004. In *Proceedings TREC 2004*, 2004.

[6] J. Conrad and M. Utt. A system for discovering relationships by feature extraction from text databases. In *SIGIR '94*, pages 260–270, 1994.

[7] N. Craswell, G. Demartini, J. Gaugaz, and T. Iofciu. L3S at INEX2008: retrieving entities using structured information. In Geva et al. [12], pages 253–263.

[8] A. de Vries, A.-M. Vercoustre, J. A. Thom, N. Craswell, and M. Lalmas. Overview of the INEX 2007 entity ranking track. In Fuhr et al. [11], pages 245–251.
[9] G. Demartini, A. de Vries, T. Iofciu, and J. Zhu. Overview of the INEX 2008 entity ranking track. In Geva et al. [12], pages 243–252.
[10] S. Fissaha Adafre, M. de Rijke, and E. Tjong Kim Sang. Entity retrieval. In *Recent Advances in Natural Language Processing (RANLP 2007)*, September 2007.
[11] N. Fuhr, J. Kamps, M. Lalmas, and A. Trotman, editors. *Focused Access to XML Documents (INEX 2007)*. Springer, 2008.
[12] S. Geva, J. Kamps, and A. Trotman, editors. *Advances in Focused Retrieval (INEX 2008)*. Springer, 2009.
[13] Z. Ghahramani and K. A. Heller. Bayesian sets. In *NIPS 2005*, 2005.
[14] GoogleSets, 2009. http://labs.google.com/sets, accessed Jan. 2009.
[15] J. Jämsen, T. Näppilä, and P. Arvola. Entity ranking based on category expansion. In Fuhr et al. [11], pages 264–278.
[16] J. Jiang, W. Liu, X. Rong, and Y. Gao. Adapting language modeling methods for expert search to rank wikipedia entities. In Geva et al. [12], pages 264–272.
[17] R. Kaptein and J. Kamps. Finding entities in wikipedia using links and categories. In Geva et al. [12], pages 273–279.
[18] J. Lafferty and C. Zhai. Document language models, query models, and risk minimization for information retrieval. In *SIGIR'01*, pages 111–119, 2001.
[19] D. Losada and L. Azzopardi. An analysis on document length retrieval trends in language modeling smoothing. *Information Retrieval*, 11(2):109–138, 2008.
[20] G. Mishne and M. de Rijke. A study of blog search. In *ECIR'06*, volume 3936 of *LNCS*, pages 289–301. Springer, 2006.
[21] H. Raghavan, J. Allan, and A. Mccallum. An exploration of entity models, collective classification and relation description. In *LinkKDD'04*, 2004.
[22] D. E. Rose and D. Levinson. Understanding user goals in web search. In *WWW '04*, pages 13–19, 2004.
[23] M. Sayyadian, A. Shakery, A. Doan, and C. Zhai. Toward entity retrieval over structured and text data. In *WIRD'04*, 2004.
[24] F. Song and W. B. Croft. A general language model for information retrieval. In *CIKM'99*, pages 316–321, 1999.
[25] T. Tsikrika et al. Structured document retrieval, multimedia retrieval, and entity ranking using PF/Tijah. In Fuhr et al. [11], pages 306–320.
[26] A.-M. Vercoustre, J. Pehcevski, and J. A. Thom. Using wikipedia categories and links in entity ranking. In Fuhr et al. [11], pages 321–335.
[27] A.-M. Vercoustre, J. A. Thom, and J. Pehcevski. Entity ranking in wikipedia. In *SAC '08*, pages 1101–1106, 2008.
[28] A.-M. Vercoustre, J. Pehcevski, and V. Naumovski. Topic difficulty prediction in entity ranking. In Geva et al. [12], pages 280–291.
[29] E. Voorhees. Overview of the TREC 2004 question answering track. In *Proceedings of TREC 2004*, 2005. NIST Special Publication: SP 500-261.
[30] W. Weerkamp, J. He, K. Balog, and E. Meij. A generative language modeling approach for ranking entities. In Geva et al. [12], pages 292–299.
[31] E. Yilmaz, E. Kanoulas, and J. A. Aslam. A simple and efficient sampling method for estimating AP and NDCG. In *SIGIR'08*, pages 603–610, 2008.
[32] H. Zaragoza, H. Rode, P. Mika, J. Atserias, M. Ciaramita, and G. Attardi. Ranking very many typed entities on wikipedia. In *CIKM'07*, pages 1015–1018, 2007.
[33] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214, 2004.
[34] J. Zhu, D. Song, and S. Rüger. Integrating document features for entity ranking. In Fuhr et al. [11], pages 336–347.