Query and Document Models for Enterprise Search

Krisztian Balog **Katja Hofmann** **Wouter Weerkamp** **Maarten de Rijke**

ISLA, University of Amsterdam
Kruislaan 403, 1098 SJ Amsterdam
http://ilps.science.uva.nl/

Abstract: We describe our participation in the TREC 2007 Enterprise track and detail our language modeling-based approaches. For document search, our focus was on estimating a mixture model using a standard web collection, and on constructing query models by employing blind relevance feedback and using the example documents provided with the topics. We found that settings performing well on a web collection do not carry over to the CSIRO collection, but the use of advanced query models resulted in significant improvements. In expert search, our experiments concerned document representation, identification of candidate experts, and combinations of expert search strategies. We find no significant difference in average precision but observe small overall positive effects of the advanced models, with large differences between individual topics.

1 Introduction

Our aim for the TREC 2007 Enterprise Track was to analyze the effect of query and document models on the performance of generative language models for enterprise search. The framework we use for our participation employs such models for document search and views expert search as an extension of these models. In addition, we describe how we address the new challenge of identifying candidate experts in the document collection.

The paper is organized as follows. We discuss our work on the document search task (Section 2) and on the expert search task (Section 3). We conclude in Section 4.

2 Document Search

The aim of the document search task is to retrieve documents that help a science communicator within an organization (in this case CSIRO) create an overview page for a given topical area. Relevant documents are therefore documents that discuss the given topic in detail and not the ones that only touch on the topic. Our aim for the document search task was to experiment with a document model based on a mixture of components and with different query models.

2.1 Modeling

We address the document search task using a language modeling approach. We rank documents according to the likelihood of the document being relevant given the query. Instead of calculating the probability $p(d|q)$ directly, we apply Bayes' rule and rewrite it to $p(d|q) = p(q|d)p(d)/p(q)$.

The probability of the query $p(q)$ can be ignored for the purpose of ranking documents, which leaves us with

$$p(d|q) \propto p(d)p(q|d). \quad (1)$$

Assuming that query terms are independent from each other, $p(q|d)$ is estimated by taking the product across terms in the query. Substituting this into Eq. 1 we obtain

$$p(d|q) \propto p(d) \prod_{t \in q} p(t|\theta_d)^{n(t,q)}, \quad (2)$$

where $n(t, q)$ denotes the number of times term t is present in query q. To prevent numerical underflows, we perform this computation in the log domain and rewrite our equation as $\log p(d|q) \propto \log p(d) + \sum_{t \in q} n(t, q) \log p(t|\theta_d)$.

Next, we generalize $n(t, q)$ so that it can take not only integer but real values. This will allow more flexible weighting of query terms. We replace $n(t, q)$ with $p(t|q)$, which can be interpreted as the weight of the term t in query q. If all query terms are equally important $p(t|q) = n(t, q)/|q|$, but we will see different weightings of query terms below.

Our final formula for ranking documents is the following:

$$\log p(d|q) \propto \log p(d) + \sum_{t \in q} p(t|q) \log p(t|\theta_d). \quad (3)$$

Next, we address the estimation of the components of our modeling: the document model $p(t|\theta_d)$ in Section 2.1.1 and the query model $p(t|q)$ in Section 2.1.2. The document priors $p(d)$ were assumed to be uniform.

2.1.1 Document model

The document model is built up from a mixture of components, where each component corresponds to certain fields or

segments of the document. Assuming that there are n components, our mixture model is constructed by taking their linear interpolation, smoothed with a background model:

$$p(t|\theta_d) = \sum_{i=1}^{n} \lambda_i p(t|d_i) + \lambda_c p(t), \quad (4)$$

where the component d_i is weighted with λ_i, and $p(t)$ is the maximum likelihood-estimate of the term t given the background (collection) model, weighted with $\lambda_c = 1 - \sum_i \lambda_i$. The components we consider are: title, meta, headings, body, and anchor text. The construction of the component indexes is described in Section 2.2. A key issue is the choice of component weights λ_i $(i = 1, \ldots, n)$. To this end we tested different configurations on the TREC 2004 Web Track [4] test set; it emerged that the title and anchor components are preferred over other components and that the influence of the background model should be very modest.

2.1.2 Query model

The core problem is to determine the probability $p(t|q)$. In our *baseline* query model we use only terms from the topic field of the query, and set

$$p(t|q) = n(t, q)/|q|. \quad (5)$$

For two of our runs we construct the query model using relevance models [6]. Given a set of feedback documents (along with relevance scores), relevance models return a number of feedback terms with an associated weight. We normalize the weights of the feedback terms so that they add up to 1 and use this weighted query to retrieve the final set of documents. We experimented with two ways of selecting feedback documents: one run (`uams07bfb`) uses blind relevance feedback by taking the top 10 documents returned by the original query; the second run (`uams07bfbex`) takes the examples of relevant documents provided by the topic descriptions. For both runs based on relevance models we combined the feedback terms and the original query terms with equal weights.

2.2 Collection Processing

Multiple indexes were created: the content of the `<title>` tag was extracted for the title index; the content of `meta name="keywords"` and `meta name="description"` tags were used to construct the metadata index. The header index used the content of `<h1>`, ..., `<h6>` tags and for the anchor index anchors were normalized. The body index used all text between `<body>` tags, in which all HTML had been removed. The background index consisted of the body combined with metadata and title.

2.3 Runs

We submitted the following runs, all of which were automatic. We determined the best mixture of weights based on experiments on the TREC 2004 Web test set (see Section 2.1.1). This mixture gives the following weights to the components: title 0.30, headings 0.10, metadata 0.05, anchor text 0.40, body text 0.10 and the background model 0.05.

`uams07bl` the baseline run; uses the baseline query model (Eq. 5) and equal weights to all components in the mixture model (components: 0.18; background: 0.10).

`uams07pr` baseline query model (Eq. 5), mixture weights based on TREC 2004 Web test set.

`uams07bfb` same as `uams07pr`, but the query model is constructed using blind relevance feedback on the top 10 documents (returned for the original query).

`uams07bfbex` identical to `uams07bfb`, except for the addition of the example documents to the document set that is used to construct the relevance models.

2.4 Results

To compare our regular runs (i.e., `uams07bl` and `uams07pr`) to feedback runs using example documents (i.e., `uams07bfbex`) we need to remove the example documents from the results and qrels. To compare our regular runs to regular runs of other TREC 2007 participants, we should not do this. Therefore, results are split into two: with and without removal of example documents. The first three rows of Table 1 indicate the results without removal, the next four with removal.

An extra feature in the assessments is the identification of "possibly relevant pages" and "key pages"; the basic metrics MAP, P@10 and MRR allow only the "key pages" as relevant, but the normalized discounted cumulative gain (nDCG) differentiates between the two by assigning higher gains to "key pages". We report on all metrics in Table 1.

Run	MAP	P@10	MRR	nDCG
without removal of examples				
`uams07bl`	.3336	.4840	.7587	.6467
`uams07pr`	.3257	.4660	**.7971**	.6466
`uams07bfb`	**.3691**	**.5180**	.6937	**.6751**
with removal of examples				
`uams07bl`	.2921	.4100	.6717	.5740
`uams07pr`	.2855	.3960	.6739	.5729
`uams07bfb`	.3291	.4640	.6487	.5987
`uams07bfbex`	**.3587**	**.5120**	**.6849**	**.6395**

Table 1: Results for the document search task.

Our findings are as follows. First, mixture component weights estimated on the TREC 2004 Web test set do not seem to be optimal for the CSIRO collection (see `uams07bl` vs. `uams07pr`). This suggests that the CSIRO collection is unlike a "standard" web collection. We leave the confirmation of this hypothesis and further explorations as future work. Second, relevance-based query models proved to be beneficial (`uams07bfb`). Combining example documents with the top relevant documents used for blind

relevance feedback resulted in substantial improvements, and was our best performing run (uams07bfbex); this run was also best capable of ranking "key pages" higher than "possibly relevant pages" as indicated by the high nDCG score of this run.

3 Expert Search

The Expert Search task presents the following scenario: Given an organization's document repository, return (e-mail addresses of) people that are experts on a given topic.

3.1 Extracting Candidate Information

A list of candidate experts had to be extracted from the document collection. Candidates are identified by their primary e-mail addresses which follow the format *firstname.lastname@csiro.au*. We also extract candidate names which are then used to form document-candidate associations. Challenges include spam protection measures, the use of alternate e-mail addresses, and of different forms of names, such as nicknames and abbreviations.

To extract candidate information we first parse out e-mail addresses from mailto link elements. We extract corresponding person names from the e-mail address, but also use heuristics (e.g., capitalization, number of words) to extract person names from anchor text and text elements neighboring the link element, thus extracting alternative forms of names. Second, we use a small set of regular expressions to extract both plain e-mail addresses and e-mail addreses that are obfuscated for spam protection. Third, we extract person names from the titles of personal pages and construct the corresponding e-mail addresses from these names.

We post-process the resulting candidate list by matching alternate e-mail addresses to the same person based on exact name matching. Finally, we remove duplicates and addresses that do not refer to person names.

3.2 Modeling

The approach we take does not require an explicit definition of the concept "being an expert," instead, we assume that people closely associated with a topic are experts on that area [5, 7]. Hence, the challenge is to estimate the strength of the association between candidate ca and topic q, which we express as $p(ca|q)$, the probability of a candidate given a topic. Instead of computing this directly, we use Bayes' Theorem and estimate: $p(ca|q) = p(q|ca)p(ca)/p(q)$, where $p(ca)$ is the probability of a candidate, and $p(q)$ is the probability of a query. Since $p(q)$ is a constant, it can be ignored for the purpose of ranking. The *apriori* belief that candidate ca is an expert, $p(ca)$, is assumed to be uniform. Thus, we rank candidates in proportion to $p(q|ca)$, the probability of the query given the candidate.

Balog et al. [2] introduce two models for estimating the probability $p(q|ca)$. *Model 1* is *candidate-based*; the candidate is modeled as a distribution over words, based on documents she is associated with. Thus, we define $p(q|\theta_{ca})$ as

$$\prod_{t \in q} \{(1-\lambda)\left(\sum_d p(t|d)p(d|ca)\right) + \lambda p(t)\}^{n(t,q)} . \quad (6)$$

Here, the candidate language model is a linear interpolation between an empirical model and the background language model $p(t)$; $n(t,q)$ is the number of times term t occurs in query q, while $p(d|ca)$ expresses the strength of the association between document d and candidate ca.

In contrast, *Model 2* is a *document-based* approach; first, documents that are relevant to the query are located, then each candidate is scored by aggregating over all relevant documents associated with the candidate: $p(q|ca) = \sum_d p(q|d)p(d|ca)$, where $p(q|d)$, the probability of a query given the document model, is calculated using Eq. 3.

For further details concerning the models we refer to [2]. This year we also experimented with combining Model 1 and 2 in order to integrate focused information from personal pages of candidates with the document-based approach. We constructed candidate language models from personal pages, where we define a personal page as a document that contains the full name of a candidate in the title field.

3.3 Document-Candidate Associations

Document-candidate associations form a key part of the models presented in Section 3.2. We need to estimate $p(d|ca)$, the probability which expresses the strength of the association between document d and candidate ca. First, association scores $a(d, ca)$ are formed for document-candidate pairs: $a(d, ca) = 0.55 \cdot A_n(d, ca) + 0.45 \cdot A_e(d, ca)$, where $A_n(d, ca)$ and $A_e(d, ca)$ are binary functions, taking either 0 or 1 as a value, depending on whether the name or e-mail address of candidate ca is recognized in document d. This particular choice of weights was acquired from previous years' experiments [1, 3]. We estimate $p(d|ca)$ using the strength of the association scores $a(d, ca)$ in two ways. Our *naive* approach uses the association scores as is: $p(d|ca) = a(d, ca)$. Our *candidate-centric* approach [2] assumes that candidate-document associations are stronger the more frequently the candidate is mentioned in the document compared to other candidates: $p(d|ca) = a(d, ca)/(\sum_{ca' \in C} a(d, ca'))$.

3.4 Supporting Documents

Runs submitted to the Expert Search task require a ranked list of experts and a ranked list of (up to 20) documents for each returned candidate that support the person's expertise on the topic at hand. For each topic q_i we ranked documents according to $p(q_i|d)$. For each candidate ca, considered as an expert, the top (up to) 20 documents associated with the person ($a(d, ca) > 0$) were returned as support.

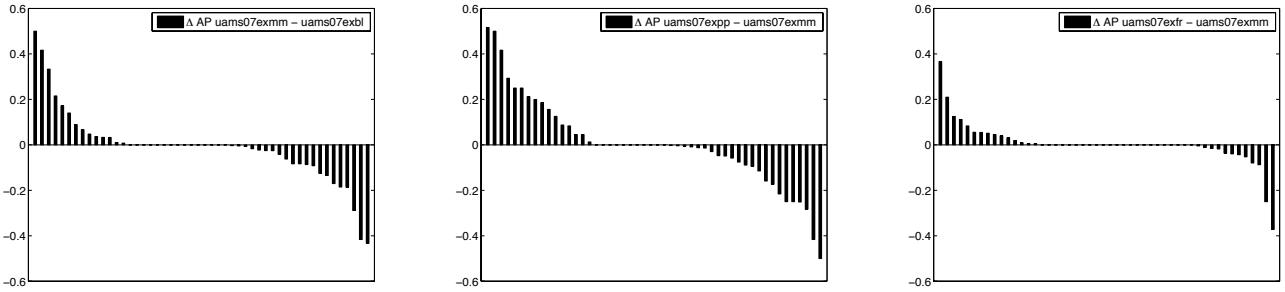

Figure 1: Per-topic differences in AP scores.

3.5 Runs

We submitted the following 4 runs:

uams07exbl the baseline run; uses a background LM and naive document-candidate associations.

uams07exmm uses the same document model as the document search run `uams07pr` with a mixture model with experimentally derived weights as described in 2.1.1. Naive document-candidate associations are used.

uams07expp identical to `uams07exmm` but a candidate-based document model is constructed from personal pages and combined with results of the previous run using equal weights.

uams07exfr again identical to `uams07exmm` but uses candidate-centric document-candidate associations.

3.6 Results

Candidate extraction produced a list of 2556 candidates. Despite the large number, we missed some of the experts of the result set created by the assessors. The result set contained 150 unique experts (2 experts were listed twice). Our extraction method found 105 (70%) of these result experts. Personal pages were found for 1038 (40.6%) of all identified experts and 68 of the experts in the result set (45.3%).

Run	#rel_ret	MAP	P@5	P@10	MRR
uams07exbl	94	**0.3090**	**0.2080**	**0.1360**	**0.4776**
uams07exmm	94	0.3011	0.1840	0.1280	0.4562
uams07expp	94	0.3066	0.1960	0.1260	0.4662
uams07exfr	**95**	0.3051	0.1960	0.1280	0.4608

Table 2: Results for the Expert Search task (supporting documents are not taken into account).

Table 2 lists the number of relevant experts, MAP scores, early precision, and mean reciprocal rank for our submitted runs. The baseline, which only uses the background document model, performs best overall. The remaining three runs rely on the mixture model from the document search task. The drop in performance parallels the lower relevance scores of the corresponding document search run.

The use of personal pages in `uams07expp` improves performance over `uams07exmm` but does not compensate for the drop in performance due to the document model used. Fig. 1 points to a large per-topic difference in AP scores between these two runs. We attribute this to the fact that personal pages were found for less than half of the candidate experts. In cases where experts for a topic did have personal pages, this information could help to improve performance on this topic. Without such information, performance may have been hurt as scores of less relevant experts with personal pages would be ranked higher.

The use of candidate-centric instead of naive document-candidate associations shows differences in the topic-wise comparison. However, differences are smaller than in the case where we use personal pages, possibly because only about 50% of the association scores differed. When only one candidate is mentioned in a document the association scores are the same as for naive document-candidate associations, and then results should equal those of `uams07exmm`.

4 Conclusions

We described our participation in the TREC 2007 Enterprise track. Building on our earlier work [1, 3], we employed a standard language modeling setting for both document and expert tasks. Our aim for the document search task was to experiment with the weights of our mixture model components using a standard web collection, and to construct query models by employing blind relevance feedback and using the example documents provided with the topics. Results show that weight settings optimized on a web collection do not carry over to an enterprise (CSIRO) collection. The use of advanced query models, on the other hand, results in significant improvements.

As to the expert search task, our experiments concerned document representation, identification of candidate experts, and combinations of expert search strategies. Advanced models display large differences in precision between individual topics, but no significant difference in average precision. These results suggest promising directions for future research, such as more sophisticated combinations of models incorporating additional information.

Acknowledgments

Balog and De Rijke were supported by the Netherlands Organisation for Scientific Research (NWO) under project numbers 220-80-001, 600.065.120 and 612.000.106. Hofmann and De Rijke were supported by NWO under project number 612.066.512. De Rijke was also supported by NWO under project numbers 017.001.190, 264-70-050, 354-20-005, 612-13-001, 612.066.302, 612.069.006, 640.001.501, 640.002.501, and by the E.U. IST programme of the 6th FP for RTD under project MultiMATCH contract IST-033104.

References

[1] L. Azzopardi, K. Balog, and M. de Rijke. Language Modeling Approaches for Enterprise Tasks. In *The Fourteenth Text Retrieval Conference (TREC 2005)*. NIST, 2006.

[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, E. Meij, and M. de Rijke. Language Models for Enterprise Search: Query Expansion and Combination of Evidence. In *The Fifteenth Text Retrieval Conference (TREC 2006)*. NIST, 2007.

[4] N. Craswell and D. Hawking. Overview of the TREC-2004 Web Track. In *Proceedings of TREC-2004*, Gaithersburg, Maryland USA, November 2004.

[5] N. Craswell, A. de Vries, and I. Soboroff. Overview of the TREC-2005 Enterprise Track. In *The Fourteenth Text Retrieval Conference (TREC 2005)*. NIST, 2006.

[6] V. Lavrenko and W. Croft. Relevance-based language models. In *SIGIR '01*, pages 120–127, 2001.

[7] I. Soboroff, A. de Vries, and N. Craswell. Overview of the TREC 2006 Enterprise Track. In *The Fifteenth Text Retrieval Conference (TREC 2006)*. NIST, 2007.