Research on Expert Search at Enterprise Track of TREC 2005

Yunbo Cao[1], Jingjing Liu[2], Shenghua Bao[3], Hang Li[1]

[1] Microsoft Research Asia, Beijing China

[2] Nankai University, Tianjin China

[3] Shanghai Jiaotong University, Shanghai China

1. INTRODUCTION

We (MSRA team) participated in the Expert Search task at the Enterprise Track of TREC 2005. This document reports our experimental results on expert search.

In our research, we have mainly investigated the effectiveness of a new approach to expert search in which we employ what is referred to as two-stage language model. It consists of two parts, relevance model and co-occurrence model. The relevance model represents whether or not a document is relevant to the query. The co-occurrence model represents whether or not the query is associated with a person. Both models are based on statistical language modeling. We have also examined the effectiveness of the use of a number of sub-models in the two-stage model; each sub-model is based on extraction of one type of metadata.

In the body-body co-occurrence sub-model, for example, we consider the use of window-based co-occurrence. The co-occurrence is about whether the query and a person appear within the same window of text. In an extreme case the entire document is viewed as a window, and the sub-model is referred to as document-based co-occurrence sub-model.

We also consider using clustering technique in re-ranking of persons. Persons are clustered according to their co-occurrences with topics and other persons.

Thus, our experiments include the following items.

1. **Two-stage language model**
   We examined how useful the two-stage model is for expert search. Most of the previous work on expert search only used co-occurrence between keyword and person. We made comparison between our approach and the existing approaches.
2. **Window-based co-occurrence sub-model**
   We compared the performance of window-based sub-model and that of the document-based sub-model. We also tried to find the best window-size for the window-based sub-model.
3. **Use of metadata**
   We tested the usefulness of different sub-models based on extraction of different types of metadata. We tried to find the best way of combining sub-models.
4. **Clustering-based re-ranking**
   We tested the effectiveness of using clustering for re-ranking persons.

Our experimental results show that the use of two-stage model can perform better than the existing approaches of solely using co-occurrence between keyword and person. The results also show that the window-based sub-model works better than the document-based sub-model. The combination of using both body-body and title-author co-occurrence sub-models can achieve the best results among all possible combinations. The clustering-based re-ranking can boost the performance in terms of average precision.

All the results except those in Section 6 were obtained from the experiments on the training set. By incrementally adding the technologies described above, we were able to achieve the best result on the training data set. We then applied it to the test query set as described in Section 6.

2. TWO-STAGE LANGUAGE MODEL

We propose a two-stage language model for expert search. Within it, a relevance model and a co-occurrence model are combined together. The relevance model represents whether or not a document is relevant to the query. The co-occurrence model represents whether or not the query is associated with a person, given a document. More specifically, the two-stage model is defined as

$$\begin{aligned} P(e \mid q) &= \sum_d P(e, d \mid q) \\ &= \sum_d P(d \mid q) P(e \mid d, q) \end{aligned} \quad (1)$$

Here $P(d \mid q)$ denotes the relevance model and $P(e \mid d, q)$ denotes the co-occurrence model. Furthermore, *d* stands for a document, *q* stands for a query, and *e* stands for a person.

We employ the language modeling technique used in IR for constructing both the relevance model $P(d \mid q)$ and the co-occurrence model. Specifically, we estimate the co-occurrence score as follows:

$$p(e \mid d, q) = \mu \frac{pf(e, d)}{|d|} + (1 - \mu) \sum_{d^{'}: e \in d^{'}} \frac{pf(e, d^{'})}{|d^{'}|} \Big/ df_e \quad (2)$$

where *pf(e, d)* is frequency of person *e* in document *d*, |*d*| is total frequency of persons in *d*, and df_e is document frequency of person *e*. We use Dirichlet prior in smoothing of parameter *μ*:

$$\mu = \frac{|d|}{|d| + \kappa} \quad (3)$$

where κ is average length of term frequency of persons in the collection.

Here, the co-occurrence model is the simplest in the sense that it does not use any metadata information. In the co-occurrence model, the query and person co-occur in the text of a document (<body> of an HTML). We refer to such kind of co-occurrence model as document-based co-occurrence (sub-)model.

Experimental results

Table 1. Using relevance model v.s. without using relevance model

	Average Precision	Bpref	Relevant Retr@10
Co-occurrence model only	0.3834	0.9477	4.10
Two-stage model	0.4438	0.9477	4.70

In the experiment, we evaluated the advantage of the two-stage model for expert ranking. In Table 1, we see that the two-stage model incorporating relevance model can significantly boost the performance in terms of both average precision and relevant retr@10.

3. WINDOW-BASED CO-OCCURRENCE SUB-MODEL

It has been proved that co-occurrence between keyword and person can be used as evidence in expert search. Various methods can be used in determining the strength of co-occurrence.

In the window-based model, the co-occurrence is about whether the query and a person appear in the same window of text. In an extreme case, the entire document is viewed as a window, which is referred to as document-based co-occurrence. (In Section 4.2, we will take a different approach in which we consider using the structure of a document in determining window size).

Experimental results

In the experiments, we tested the performances of the window-based model in different sizes of windows. We also compared the results with that of the document-based model.

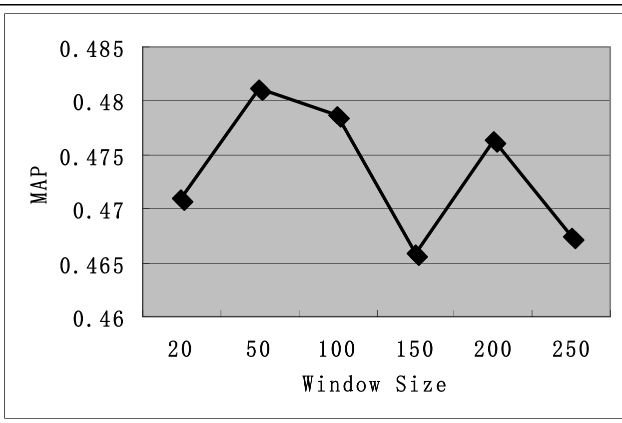

Figure 1. Window-based model in different window sizes

Figure 1 shows the performances of the window-based model with different window sizes. The best result is obtained when the window size is set as 50.

Table 2. Window-based model v.s. document-based model

	Average Precision	Bpref	Relevant Retr@10
Document-based model	0.4438	0.9477	4.70
Window-based model	0.4811	0.9362	5.00

Table 2 shows a comparison between the window-based co-occurrence model with the document-based co-occurrence model. From Table 2, we see that all the three evaluated measures can be significantly improved in the window-based model.

4. USE OF METADATA

We extract metadata such as <body>, <title>, <author>, <anchor text>, and <section> and make use of them in building co-occurrence models.

Here <anchor text> is the texts in the in-links of a document. <title> and <author> are the text and person names appearing at the beginning of a document, respectively. We will explain what <section> is later.

4.1 Sub-models based on metadata

We further construct the co-occurrence model by using a number of sub-models. The sub-models are models created on the basis of metadata extraction. Given a query, the title-author sub-model, for example, represents the association relation when the query appears in the title and the person appears as the author of document. All the sub-models are in the form of statistical language model. Table 1 gives the sub-model we used. Here model M1 is the co-occurrence model described in Section 3.

Table 3. Sub-models based on metadata

Model ID	Query	Person
M1	Body	Body
M2	Anchor Text	Body
M3	Title	Body
M4	Body	Author
M5	Anchor Text	Author
M6	Title	Author

The sub-models are linearly combined together to form the co-occurrence model in the following way:

$$p(e|d,q) = \sum_m \lambda_m p_m(e|d,q) \qquad (4)$$

where $p_m(e|d,q)$ denotes the m^{th} co-occurrence sub-model. Each sub-model is estimated separately using Equation (2). For example, in M2, if the query occurs in the anchor text of a document and the person occurs in the body of the document, we estimate the score using Equation (2). Each sub-model has a weight. M6 has a large weight because <title> and <author> are important metadata.

Experimental results

In the experiments, we evaluated the contribution of each co-occurrence sub-model. Table 4 shows the performance of each individual sub-model. From the results, we can see that M1 performs best, followed by M2 and M3. All of the three sub-models used person information in the <body>s of documents. Thus, we can say that persons in <body>s cover more answers than those in <author>s.

Table 4. Comparing sub-models based on metadata

	Average Precision	Bpref	Relevant Retr@10
M1	0.4811	0.9362	5.00
M2	0.4318	0.7501	4.57
M3	0.4610	0.7948	4.57
M4	0.2259	0.4111	2.89
M5	0.0691	0.1057	1.50
M6	0.1523	0.2557	2.33
M1+M6	0.4830	0.9561	4.70

We also tried various combinations of the sub-models. The experiments show that combination of M1 and M6 with weights 1 and 45 achieve the best result. The last row in Table 4 gives the performance of the best combination.

4.2 Using <section> metadata

We use the <section> metadata in two ways.

First, we use <section> as a constraint for restricting window-based co-occurrence model. Here, <section> denotes the text block within one of the HTML tags listed in Table 5.

Table 5. HTML tags for identifying <section>

<Table>,   , <p>,   , <pre>,   , <dl>, <dt>, <tr>, <hr>

When the query matches key words in a document, we look for persons surrounding the key words within the <section> that is the smallest embodied in the given window. If the left or the right boundary of the <section> exceeds the left or right boundary of the window, the left or right boundary of the window will be used.

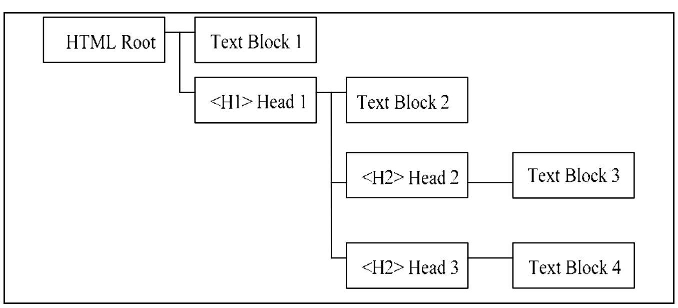

Figure 2. Example on tree-section

Second, we use the <section>s in constructing a new sub-model on the basis of the 'section-tree' (cf., Figure 2). We denote the sub-model by section-tree co-occurrence sub-model. The sub-model is used as one of the co-occurrence sub-models as described in Section 4.1.

Here, we use the HTML tags, <H1>, <H2>, <H3>, <H4>, <H5>, and <H6>, to define <section>s in a section-tree. We define a preference order over the tags: <H1> $\succ$ <H2> $\succ$ <H3> $\succ$ <H4> $\succ$ <H5> $\succ$ <H6>. We build a tree-structure, for example, as that in Figure 2. We then build a sub-model as Figure 3. The model represents the association of queries and persons within the HEAD fields of HTML and the text blocks.

In line 3 of Figure 3, function *AncestorPath(Ti)* is used to concatenate all the texts in the path from HTML Root to text block *Ti*. For example, in Figure 2, the ancestor path for Text Block 3 is "HTML Root--><H1> Head1--> <H2> Head 2". In line 4, the function *match (AncestorPath (Ti), q)* embodies the rules of matching query q to *AncestorPath(Ti.)* It returns *true* when all the query words in the query *q* appear in the ancestor path; otherwise, it returns *false*.

```
1)  For a given query q
2)    Foreach Text Block Ti{
3)      Get the ancestor path AncestorPath (Ti)
4)      if (match (AncestorPath (Ti), q)){
5         Foreach candidate experts e appears in the Text Block Ti{
6           count the co-occurrence of e and the topic q;}
8     }
9  }
```

Figure 3. Section-tree model

Experimental results

We conducted an experiment to evaluate the effectiveness of use of <section>s. The experiments include that using them as constraints in the window-based co-occurrence sub-model and that using them in the section-tree co-occurrence sub-model.

Table 6. Using <section> metadata

	Average Precision	Bpref	Relevant Retr@10
Baseline	0.4830	0.9561	4.70
Baseline + Section-constraint	0.4876	0.9370	5.00
Baseline + Section-constraint + Section-tree	0.6036	0.9505	6.20

In Table 6, the baseline is the model of combing M1 and M6 as given in Section 4.1. From Table 6, we see that both ways of using <section>s can boost the performance. When incorporating the section-tree co-occurrence sub-model into the co-occurrence model, we set its weight as 1000.

5. CLUSTERING-BASED RE-RANKING

In the clustering-based re-ranking, we try to utilize relations between people to enhance expert search results. The relationships we use belong to two categories:

1. Persons appear in the similar contexts
2. Persons co-occur in some local contexts.

For the first category, we construct a context vector to represent keywords co-occurring with a person. The key-words are those appearing in the context of a person. For the second category, we construct a context vector which represents persons (their names) co-occurring with a person. Both contexts are defined as texts in a fixed-size window. The window size we used in the experiments was 100.

Next, we concatenate the two context vectors into a single context vector for a given person. Thus, the context vector has two categories of information.

Then, we cluster the persons according to the similarity in the combined context vectors. We use K-Means as the clustering algorithm. The clustering results are used in re-ranking as in Equation (5),

$$p_{new}(e \mid q) = (1-\lambda)\, p(e \mid q) + \lambda \sum_{e' \in E} p(e' \mid q) \Big/ \mid E \mid \qquad (5)$$

where E repents the cluster which e' belongs to and $|E|$ represents the size of E.

Finally, we use the new score for re-ranking the candidates from rank 10. That is to say that we assume that the top 10 ranks are correct.

Experimental results

In the experiments, we used 20 clusters for clustering people.

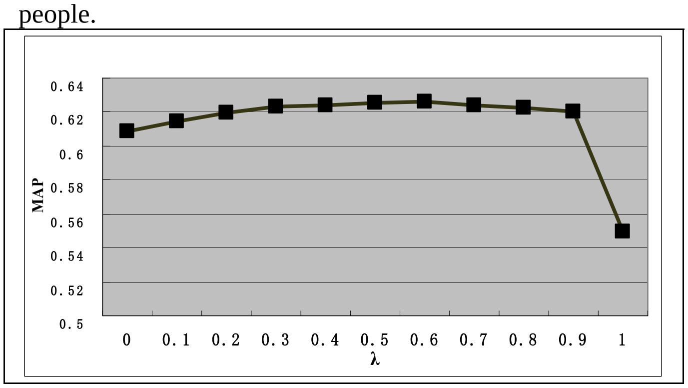

Figure 4. The effect of different λ for re-ranking

Obviously, the value of parameter λ is the key for the clustering-based re-ranking. We tried several values of λ to find the best result. Figure 4 shows the results in terms of MAP when λ changes. From Figure 4, we see that the performance of the re-ranking approach is insensitive to λ within a wide range. In the following experiments, we used 0.5 for λ .

Table 7 shows the performance of clustering-based re-ranking. Here the baseline is the model performing the best in Section 4.2. We see that the clustering-based re-ranking can further boost average precision.

Table 7. Clustering-based re-ranking

	Average Precision	Bpref	Relevant Retr@10
Baseline	0.6036	0.9505	6.20
Baseline + re-ranking	0.6253	0.9505	6.20

6. SUBMITTED RUNS

We submitted five runs using the test queries. All the five runs were based on the basic model used in Section 4.2. The basic model consisted of a relevance model and a co-occurrence model including M1, M6 and section-tree sub-models. The weights for the sub-model M1, M6 and section-tree were 1, 45, and 1000, respectively. The major differences between the five runs are described below.

1. MSRA051 -- Using the queries that are exactly the same as the texts in the <title> fields of the test query set.
2. MSRA052 -- Using the queries from the <title> fields of the test query set with acronym normalization. In acronym normalization, for example, “Extensible Markup Language” is converted into its acronym form “XML”. It is assumed here that a dictionary of acronyms is available. The dictionary can also be automatically constructed from corpus.
3. MSRA053 -- Using the queries from the <title> fields of the test query set with redundant acronyms removed. For example, the redundant acronym in “Extensible Markup Language (XML)” is removed and then “Extensible Markup Language” is obtained.
4. MSRA054 – Using the same queries as in MSRA051 and employing clustering-based re-ranking in Section 5.
5. MSRA055 – Using the same queries as in MSRA052 and employing clustering-based re-ranking in Section 5.

Table 8 shows the results of the five runs on the test queries.

Table 8. The five submitted runs

	Average Precision	Bpref	Relevant Retr@10
MSRA051	0.2573	0.5552	3.70
MSRA052	0.2503	0.5571	3.58
MSRA053	0.2569	0.5588	3.64
MSRA054	0.2688	0.5685	3.70
MSRA055	0.2600	0.5655	3.58