Probabilistic Models for Expert Finding

Hui Fang and ChengXiang Zhai

University of Illinois at Urbana-Champaign, IL, USA
{hfang,czhai}@cs.uiuc.edu

Abstract. A common task in many applications is to find persons who are knowledgeable about a given topic (i.e., *expert finding*). In this paper, we propose and develop a general probabilistic framework for studying expert finding problem and derive two families of generative models (candidate generation models and topic generation models) from the framework. These models subsume most existing language models proposed for expert finding. We further propose several techniques to improve the estimation of the proposed models, including incorporating topic expansion, using a mixture model to model candidate mentions in the supporting documents, and defining an email count-based prior in the topic generation model. Our experiments show that the proposed estimation strategies are all effective to improve retrieval accuracy.

1 Introduction

The problem of *expert finding* is concerned with finding the experts on a specified topic. This problem has many real-world applications. For example, organizers of a conference need to assign submissions to the PC members based on their research interests and expertise. Customer service of a company needs to decide which staff should be assigned to solve a given complaint. Currently, people have to manually identify the experts, which is obviously labor-intensive and time-consuming. Thus, it would be very interesting to study how to automatically identify experts for a specified expertise area.

As a retrieval task, expert finding has recently attracted much attention mostly due to the launching of the Enterprise track [4, 11] of TREC [12]. The task setup in the Enterprise track includes the following three components: (1) a supporting document collection; (2) a list of expert candidates, which are specified by names and emails; (3) a set of topics (i.e., descriptions of expertise). The task of expert finding is to rank the expert candidates for a given topic query based on the information from the data collection. Expert finding is similar to the traditional ad hoc retrieval task in the sense that both tasks are to find relevant information items for a given topic. The key challenge in expert finding is to infer the association between a person (i.e., candidate expert) and an expertise area (i.e., topic) from the supporting document collection.

Participants in the Enterprise track have tried various methods. The methods mainly fall into two categories: profile-based methods and document-based methods. In profile-based methods [5, 1, 7, 2], researchers would first build a term-based expertise profile (called document reorganization in [5]) for each candidate, and rank the candidate experts based on the relevance scores of their profiles for a given topic by using traditional ad hoc retrieval models. In document-based methods [3, 2, 8], instead of creating such term-based expertise profiles, the researchers use the supporting documents as a “bridge” and rank the candidates based on the co-occurrences of topic and candidate mentions in the supporting documents. However, the existing methods are not general and usually rely on heuristics, such as rule-based methods to detect the candidate mentions in the supporting documents, to achieve reasonable retrieval accuracy.

In this paper, we develop a general probabilistic framework for studying expert finding. We derive two families of generative models based on the framework - candidate generation models and topic generation models. The derived models are analogous to the probabilistic models derived in [6] for traditional ad hoc retrieval. These models cover most existing probabilistic models for expert finding, including probabilistic versions of both profile-based and document-based methods [3, 2, 1, 8]. We further develop several techniques to improve the estimation of the proposed models, including incorporating topic expansion, using a mixture model to put different weights on the matching of different representations of an expert candidate, and defining a candidate prior for topic generation models based on the counts of email matches in the supporting documents. Evaluation on two standard TREC test collections shows that both families of models perform well empirically. In addition, we discover that how the judgements are made can affect the relative performance of different models. Experiment results also show that putting different weights on the matching of different representations of candidates with a simple mixture model is beneficial. Furthermore, it is also beneficial to compute the candidate prior for topic generation models based on the email counts in the supporting documents. However, topic expansion only improves the performance slightly when optimized.

2 A Probabilistic Framework for Expert Finding

Recognizing the similarity between expert finding and the traditional ad hoc retrieval task, we can apply the probabilistic ranking principle [10] to develop a general probabilistic framework for expert finding. Specifically, we will rank the candidates according to the probability that a candidate is “relevant” to the topic (i.e., expertise) specified in a query, and the key challenge is to compute this probability.

Formally, suppose $S = \{d_1, ..., d_{|S|}\}$ is a collection of supporting documents. Let $t = t_1, t_2, ..., t_n$ be the description of a topic, where t_i is a term in the description. Let c be an expert candidate whose email and name are denoted as $e(c)$ and $n(c)$, respectively. Let R be a binary random variable to denote relevance (1 for relevant and 0 for non-relevant). Given a query t and an expert candidate c, we are interested in estimating the conditional probability $p(R = 1|c, t)$, i.e., the probability that candidate c is relevant to topic t. After using odds ratio to rank the candidates and applying the Bayes' Theorem we have

$$p(R = 1|c, t) \overset{\text{rank}}{=} \frac{p(R=1|c,t)}{p(R=0|c,t)} \overset{\text{rank}}{=} \frac{p(c,t|R=1)}{p(c,t|R=0)} \qquad (1)$$

where $\overset{\text{rank}}{=}$ means “equivalence for ranking the candidates”.

We now discuss two different ways to factor the conditional probabilities $p(c, t|R = 1)$ and $p(c, t|R = 0)$. They correspond to two different families of probabilistic models, which are referred to as *candidate generation models* and *topic generation models*, respectively. The high-level derivation is in spirit the same as in [6] if we take a topic (i.e., t) as Q and a candidate (i.e., c) as D.

2.1 Candidate generation models

One way to factor the conditional probabilities in Equation 1 is as follows:

$$\frac{p(c, t|R = 1)}{p(c, t|R = 0)} = \frac{p(c|t, R = 1)p(t|R = 1)}{p(c|t, R = 0)p(t|R = 0)}$$

Here we assume that an expert candidate c is “generated” by a probabilistic model based on a query t. Thus, this family of probabilistic models is referred to as *candidate generation models*.

Since $p(t|R = 1)$ and $p(t|R = 0)$ are independent of the candidates, they can be ignored for the purpose of ranking candidates. Thus, the general retrieval function of the *candidate generation model* is:

$$p(R = 1|c, t) \overset{\text{rank}}{=} \frac{p(c|t, R = 1)}{p(c|t, R = 0)},$$

where $p(c|t, R = 1)$ is the probability of candidate c given the “expert generative model” topic t, while $p(c|t, R = 0)$ is the probability of c given the “non-expert generative model” of t. Thus the main question is how to estimate $p(c|t, R = 1)$ and $p(c|t, R = 0)$.

For $p(c|t, R = 1)$, we may use the supporting documents to connect t and c in the following way:

$$p(c|t, R = 1) = \sum_{d \in S} p(c|d, t, R = 1) \times p(d|t, R = 1) \approx \sum_{d \in S} p(c|d, R = 1) \times p(d|t, R = 1) \tag{2}$$

Here we assume that t and c are independent given the document d and the event $R = 1$. Intuitively, this formula is quite reasonable: $p(d|t, R = 1)$ allows us to model the probability that a document d matches a topic t while $p(c|d, R = 1)$ allows us to model the probability that a supporting document mentions a candidate c. A document d with higher values for both $p(c|d, R = 1)$ and $p(d|t, R = 1)$ would contribute more to the estimation of $p(c|t, R = 1)$, which intuitively makes sense. Indeed, this is essentially to exploit the co-occurrences of the topic terms and candidate mentions in the supporting documents, an idea already used in the existing work [2, 3],

Unfortunately, it is not immediately clear how we should estimate $p(c|t, R = 0)$, the “non-expert” model for topic t. The difficulty comes from the fact that we do not have evidence for a candidate not to be an expert for t. Thus as a possibly inaccurate simplification, we simply assume that $p(c|t, R = 0)$ is uniformly distributed, leaving more accurate estimation as our future work. This assumption allows us to drop $p(c|t, R = 0)$ and rank candidates solely based on $p(c|t, R = 1)$, which we estimate using Equation 2.

To compute $p(d|t, R = 1)$ efficiently, we apply Bayes' Theorem and rewrite the equation in the following way:

$$p(c|t, R = 1) = \sum_{d \in S} p(c|d, R = 1) \times \frac{p(t|d, R = 1)p(d|R = 1)}{\sum_{d' \in S} p(t|d', R = 1)p(d'|R = 1)}$$

Since $\sum_{d' \in S} p(t|d', R = 1)p(d'|R = 1)$ is the same for all the candidates, it can be dropped for ranking. $p(d|R = 1)$ can be regarded as a prior on d that can be exploited to favor a certain type of documents (e.g., email messages) in S. For simplicity, we assume that $p(d|R = 1)$ is uniform, which leads to

$$p(R = 1|c, t) \overset{\text{rank}}{=} \sum_{d \in S} p(c|d, R = 1) \times p(t|d, R = 1), \tag{3}$$

where $p(t|d, R = 1)$ is the probability that the topic t is relevant to the document d and $p(c|d, R = 1)$ is the probability that the candidate c is mentioned in the document d. Both of them can be computed efficiently by using an existing probabilistic retrieval model[14].

The candidate generation model shown in Equation 2 covers the two-stage model proposed in [3] as a special case and can be regarded as a probabilistic version of the document-based approaches to expert finding.

2.2 Topic generation models

The other way to factor the conditional probabilities in Equation 1 is as follows:

$$\frac{p(c, t|R = 1)}{p(c, t|R = 0)} = \frac{p(t|c, R = 1)p(c|R = 1)}{p(t|c, R = 0)p(c|R = 0)}$$

Here a topic t is assumed to be "generated" by a probabilistic model based on an expert candidate c, thus we call this family of probabilistic models *topic generation models*. The general retrieval function of *topic generation models* is:

$$p(R = 1|c, t) \overset{\text{rank}}{=} \frac{p(c|R = 1)}{p(c|R = 0)} \times \frac{p(t|c, R = 1)}{p(t|c, R = 0)},$$

where $p(t|c, R = 1)$ is the probability of topic t according to the "expertise topic model" of candidate c, $p(t|c, R = 0)$ is the probability of topic t according to "non-expertise topic model" of candidate c, $p(c|R = 1)$ is the prior probability that c is an expert, and $p(c|R = 0)$ is the prior probability that c is not an expert. The expert finding problem is thus reduced to the problem of estimating these probabilities.

As in the case of candidate generation models, we make a possibly inaccurate simplification assumption that $p(t|c, R = 0)$ is uniform due to the lack of appropriate data for estimating it. Thus the major task is to estimate $p(t|c, R = 1)$, the probability that t describes the expertise of candidate c. We discuss two possible ways to estimate it: *profile-based estimation* and *document-based estimation*, which correspond to the two categories of the existing methods for expert finding.

Profile-based estimation: The idea of profile-based estimation is to first estimate an expertise profile language model θ_c for every expert candidate c, and then compute the likelihood of t given the profile language model θ_c, i.e.:

$$p(t|c, R = 1) \approx p(t|\theta_c, R = 1) \tag{4}$$

Naturally, the key challenge is to estimate expertise profile θ_c for a candidate.

One possible estimation method proposed in [2] is as follows:

$$\begin{aligned} p(t|\theta_c, R = 1) &= \prod_{t_i \in t} p(t_i|\theta_c, R = 1)^{count(t_i,t)} & (5) \\ &= \prod_{t_i \in t} (\sum_{d \in S} p(t_i|d, R = 1) p(d|c, R = 1))^{count(t_i,t)} & (6) \end{aligned}$$

where $count(t_i, t)$ is the count of term t_i in the query t. This model (i.e., Model 1 in [2]) was shown to perform consistently worse than the other model in [2], but no clear explanation was given. When viewing the method in our probabilistic framework, we see that a main reason why this method does not work well is because the estimation of $p(t_i|\theta_c, R = 1)$ is not accurate. Specifically, the problem lies in that a supporting document matching candidate c well (i.e., with a high value of $p(d|c, R = 1)$) may not necessarily support that the candidate is an expert on a topic (i.e., $p(t|d)$ may be low). Based on Equation 6, as long as document d contains one query term t_i and mentions the candidate c, the document would be regarded as a useful document to support that c is an expert on topic t. This is clearly inaccurate because the document might not match the whole query concept even though it matches one query term very well.

Based on this analysis, it would be reasonable to hypothesize that when we selectively use the documents that truly reflect the expertise of candidate c, instead of using every document in the collection as shown in Equation 7, such profile-based estimation will perform better. We have not further explored this direction; instead, we will use the document-based estimation method (to be described below) in our experiments.

Document-based estimation: Instead of creating an expertise profile language model, we could use supporting documents to connect a candidate and a topic as in the candidate generation models. Specifically, making similar assumptions as we have made in estimating candidate generation models, we have

$$p(t|c, R = 1) = \sum_{d \in S} p(t|d, c, R = 1) \times p(d|c, R = 1) \approx \sum_{d \in S} p(t|d, R = 1) \times p(d|c, R = 1).$$

Thus, the expert candidates can be ranked according to

$$p(R = 1|c, t) \overset{\text{rank}}{=} \frac{p(c|R = 1)}{p(c|R = 0)} \times \sum_{d \in S} (p(t|d, R = 1) \times p(d|c, R = 1)) \tag{7}$$

Similarly, we rewrite $p(d|c, R = 1)$ in terms of $p(c|d, R = 1)$, which can be efficiently computed by treating representations of the candidate c as a query and using a standard probabilistic retrieval method to compute $p(c|d, R = 1)$. In addition, as in the candidate generation models, we assume $p(d|R = 1)$ is uniform, which leads to

$$p(R = 1|c, t) \overset{\text{rank}}{=} \frac{p(c|R = 1)}{p(c|R = 0)} \times \sum_{d \in S} (p(t|d, R = 1) \times \frac{p(c|d, R = 1)}{\sum_{d' \in S} p(c|d', R = 1)}), \qquad (8)$$

where $p(c|R = 1)$ is the prior probability that a candidate c is an expert, $p(c|R = 0)$ is the prior probability that a candidate is not an expert, $p(t|d, R = 1)$ is the probability that topic t is discussed in document d, and $p(c|d, R = 1)$ is the probability that an expert candidate c is mentioned in the document d.

Comparing the two models in Equations (3) and (8), a main difference is that the topic generation model contains a candidate normalizer, $N(c) = \sum_{d \in S} p(c|d, R = 1)$, while the candidate generation model does not. Since a larger $N(c)$ indicates a more popular c, this normalizer would penalize a popular candidate. As we will show in Section 3, this normalization factor could provide an explanation for the performance difference between the two models.

The Model 2 proposed in [2] and the model used in [8] are both special cases of the model we presented in Equation 7 when $p(c|R = 1)$ and $p(c|R = 0)$ are assumed to be uniform. As we will show later, it is possible to specify a non-uniform prior to improve retrieval accuracy.

2.3 Estimation of component models

So far, we have derived two families of generative models, as shown in Equation 3 and Equation 8. They contain the following three component models: (1) $p(c|d, R = 1)$, the probability that candidate c is mentioned in document d; (2) $p(t|d, R = 1)$, the probability that topic t is discussed in document d; (3) $p(c|R = 1)$ and $p(c|R = 0)$, the prior probabilities of a candidate. We now discuss how we estimate each of them.

Modeling candidate mentions $p(c|d, R = 1)$: In general, $p(c|d, R = 1)$ can be computed by treating the description of candidate c (e.g., name and/or email) as a query and using a standard retrieval method to score document d.

The simplest method is to concatenate the email $e(c)$ and the name $n(c)$ to form a query to represent candidate c. $p(c|d, R = 1)$ could thus be computed as

$$p(c|d, R = 1) = p(\text{“}e(c), n(c)'' |d, R = 1) \qquad (9)$$

However, intuitively, a name and an email have different characteristics. For example, using email alone to identify an expert could generate high-precision results, while using name alone to identify an expert may lead to high-recall results due to partial matching of names. Thus intuitively, to achieve the best results, it would be reasonable to combine them with appropriate weights. To capture this intuition, we propose to model $p(c|d, R = 1)$ using a mixture model involving both $p(e(c)|d, R = 1)$ and $p(n(c)|d, R = 1)$. That is, $p(c|d, R = 1)$ can be computed as

$$p(c|d, R = 1) = \lambda_e \cdot p(e(c)|d, R = 1) + (1 - \lambda_e) \cdot p(n(c)|d, R = 1), \qquad (10)$$

where λ_e is the weight of the email model. Since $e(c)$ and $n(c)$ are both text, $p(e(c)|d, R = 1)$ and $p(n(c)|d, R = 1)$ can be computed using query generation retrieval model with Dirichlet prior smoothing as described in [14].

Modeling topic-document relationship $p(t|d, R = 1)$: Since t is a piece of text, $p(t|d, R = 1)$ can be computed using the query generation retrieval method with Dirichlet prior smoothing [14]. However, the original topic description t tends to be quite short, so it may not be informative. We thus propose to use some pseudo feedback method (e.g., the model-based feedback method proposed in [13]) to estimate an enriched topic query model θ_t, and incorporate this query model into our candidate finding model through generalizing the topic likelihood $p(t|d, R = 1)$ as the cross entropy of the query model θ_t and the document model θ_d estimated based on d using Dirichlet prior smoothing. That is,

$$p(t|d, R = 1) \propto exp(\sum_w p(w|\theta_t) log(p(w|\theta_d)). \tag{11}$$

Clearly, if we set θ_t to the empirical word distribution in t, this would be equivalent to the original topic likelihood.

Setting the candidate prior $\frac{p(c|R=1)}{p(c|R=0)}$: $p(c|R = 1)$ and $p(c|R = 0)$ are the prior probabilities of a candidate. In most existing work [3, 2], they are assumed to be uniform. However, as shown in Section 3, reasonable prior can help improve the retrieval accuracy.

Based on Bayes' Theorem, we can rewrite the candidate prior as

$$\frac{p(c|R = 1)}{p(c|R = 0)} \overset{\text{rank}}{=} \frac{p(R = 1|c)}{p(R = 0|c)} = \frac{p(R = 1|c)}{1 - p(R = 1|c)},$$

where $p(R = 1|c)$ is the probability that a candidate is an expert. To estimate it, we may reasonably assume that a candidate whose email has been mentioned many times has a high prior probability of being an expert. We adapt a formula that is similar to BM25 term frequency normalization formula [9], i.e., $p(R = 1|c) \propto \frac{count(e(c),S)}{2\times count(e(c),S)+\beta}$. Thus, the candidate prior is

$$\frac{p(c|R = 1)}{p(c|R = 0)} \propto \frac{count(e(c), S)}{count(e(c), S) + \beta}, \tag{12}$$

where $count(e(c), S)$ is the count of mentions of the email of candidate c in the collection S, and β is the parameter to control the skewness of the prior. A larger β would reduce the skewness of the prior (i.e., leading to a weaker prior), thus it can be interpreted as being inversely proportional to our confidence in this prior.

3 Experiments

We evaluate the proposed models on two TREC enterprise collections [4] - (1) **Ent05:** W3C corpus with topics EX01-EX50. The topics are the names of working groups. This is the collection used in the Enterprise track of 2005. (2) **Ent06:**

W3C corpus with topics EX51-EX105. The topics are contributed by the participants of enterprise track in 2006, and represent the real world information need. This is the collection used in the Enterprise track of 2006.

We use the entire corpus, and only the titles of topic descriptions. We have done minimal preprocessing, where we apply stemming with a Porter stemmer and no stop word is removed. We evaluate the methods with mean average precision (MAP), which is the official evaluation measure of expert finding task in enterprise track. We conduct six sets of experiments. First, we evaluate the proposed models, and compare them with the baseline models. Second, we examine the effectiveness of using a mixture model (i.e., Equation 10) to model the candidate mentions. Third, we study the effectiveness of topic expansion described in Equation 11. Fourth, we demonstrate the effectiveness of candidate prior proposed in Equation 12. Fifth, we examine the parameter sensitivity. Finally, we compare our results to the official TREC runs.

3.1 Comparison of proposed models

In Table 1, we compare the optimal performance of the proposed two models with a state-of-the-art baseline model. "Cand-gen (mixture)" is the candidate generation model (Equation (3)) estimated using the mixture model in Equation (10). "Topic-gen (mixture+prior) " is the topic generation model (Equation (8)) estimated using the mixture model in Equation (10) and the prior in Equation (12). In both models, $p(t|d, R = 1)$ is computed using query generation model with Dirichlet prior smoothing as described in [14]. "Baseline" is the topic generation model in Equation (8) estimated using Equation 9 and a uniform prior; this model is essentially similar to the model 2 proposed in [2].

Table 1. Performance comparison of different models

Models	Ent05	Ent06
Baseline	0.151	0.191
Cand-gen (mixture)	0.186	**0.449**
Topic-gen (mixture+prior)	**0.196**	0.334

Table 1 shows that our proposed models perform much better than the baseline model. In addition, the optimal performance of topic generation model on Ent05 is 0.196, which is better than the best reported performance (i.e., 0.1894) of a similar model using rule-based name matching (i.e., Model 2 in [2]).

Table 2. Average popularity of candidates

	Judgements	Results of Topic-gen	Results of Cand-gen
Ent05	0.91	1.18	2.27
Ent06	2.37	1.21	2.02

Compared with topic generation model, the candidate generation model performs slightly worse on Ent05 but much better on Ent06. When looking into it, we found that such performance difference may be caused by the different ways of creating judgements in these two years' Enterprise track. In Ent05, the topics are the names of working groups, and the judgements are independent of the document collection. But in Ent06, the topics are contributed by the participants, and the judgements are created based on the information from the document collection. Such different ways of creating judgements directly affect the characteristics of the relevant experts, especially their occurrences in the collection (i.e., the popularity of the experts). Using the normalizer $N(c) = \sum_{d \in S} p(c|d, R = 1)$ as a measure of candidate popularity, we show some popularity statistics in Table 2. We see that, as expected, the relevant experts in Ent06 are more popular in the collection compared with those identified in Ent05, which means that we should expect penalizing popular experts to help more (or harm less) for Ent05 than for Ent06. Indeed, the fact that the topic-generation model performs better than the candidate-generation model on Ent05, but worse on Ent06 suggests that such penalization is beneficial for Ent05 but harmful for Ent06; this is because the two models differ mainly in the penalization of popular candidates as discussed in Section 2. From Table 2, we can also see that the average popularity of the candidates returned by the topic-generation model is indeed much lower than that returned by the candidate-generation model.

It is surprising that the different ways of how Ent05 and Ent06 are created can affect the relative performance of the two models, making it interesting to further explore how to create appropriate test collections for expert finding.

3.2 Effectiveness of mixture model for candidate mentions

Table 3. Effectiveness of mixture model for candidate mentions

		Ent05	Ent06
Cand-gen	merge	0.130	0.280
	mixture	0.186 (43%)	0.449 (60%)
Topic-gen (prior)	merge	0.155	0.193
	mixture	0.196 (26%)	0.334 (73%)

We discussed two ways to model candidate mentions as shown in Equation 9 (denoted as "merge") and in Equation 10 (denoted as "mixture"). We compare these two estimations and report the results in Table 3. The results show that it is more effective to use a mixture model to model the candidate mentions.

3.3 Effectiveness of topic expansion

We proposed two possible ways to model the topic-document relationship: (1) "query likelihood", which is the query generation model described in [14]; (2) "feedback", which is the model-based feedback method described in [13]. We compare these two strategies and report the optimal performance in Table 4. The results show that topic expansion consistently improves the performance when optimized, but the performance improvement is smaller compared with the performance improvement in traditional ad hoc retrieval problem [13].

Table 4. Effectiveness of topic expansion

	Ent05		Ent06	
	query likelihood	expansion	query likelihood	expansion
Cand-gen(mixture)	0.186	0.196 (5%)	0.449	0.465 (4%)
Topic-gen(mixture+prior)	0.196	0.204 (4%)	0.334	0.359 (7%)

3.4 Effectiveness of candidate prior in topic generation models

In the topic generation models, we proposed to compute the candidate prior based on the counts of email using Equation 12, which is denoted to as "email-prior". In contrast, most existing work assumes that the candidate prior is uniform, which is denoted as "uniformprior". Table 5 shows the performance difference between these two strategies. As shown in Table 5, incorporating email-based prior could improve the performance. It is interesting that when we estimate the candidate mentions with the mixture model as in Equation 10, the email-based prior improves the performance more.

Table 5. Performance comparison for candidate prior

Topic-gen		Ent05	Ent06
merge	uniformprior	0.151	0.191
	emailprior	0.155 (3%)	0.193 (1%)
mixture	uniformprior	0.172	0.204
	emailprior	0.196 (14%)	0.334 (64%)

3.5 Parameter Sensitivity

There are four parameters in both candidate generation models and topic generation models: μ_t, μ_e, μ_n and λ_e. Topic generation models have one extra parameter for the candidate prior, i.e., β.

μ_t, μ_e and μ_n are the Dirichlet prior smoothing parameters used to compute $p(t|d, R = 1)$, $p(e(c)|d, R = 1)$ and $p(n(c)|d, R = 1)$, respectively. We examine the parameter sensitivity for these three parameters in Figure 1. In every case, we change the value of one parameter while fixing the value of the other parameters. The plots show that the performance is relatively more stable to the change of μ_e compared with the change of μ_t and μ_n. It is interesting that the optimal values of μ's are generally around 100, which is much smaller than 2000, the recommended value in traditional ad hoc retrieval [14].

λ_e controls the relative weight on email matching to name matching (as in Equation 10). We now examine how it affects the performance. The left plot in Figure 2 shows that the performance is relatively stable when $\lambda_e < 0.9$.

β is a prior confidence parameter defined in Equation 12. The right plot in Figure 2 shows the parameter sensitivity curve of this parameter. We observe that the optimal values of β are different for the two data sets. Larger β leads to better performance in Ent06, while it leads to worse performance in Ent05. Such observation implies that we could be more confident in the prior on Ent05 than on Ent06, which might be related to the different characteristics of two sets of queries. We leave the further analysis as our future work.

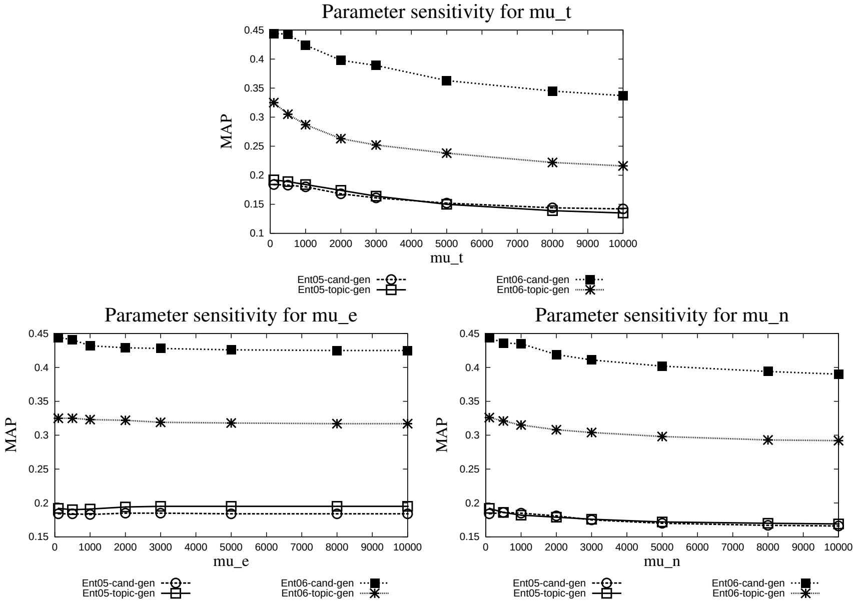

Fig. 1. Performance Sensitivity of μ_t(upper), μ_e (lower left) and μ_n (lower right)

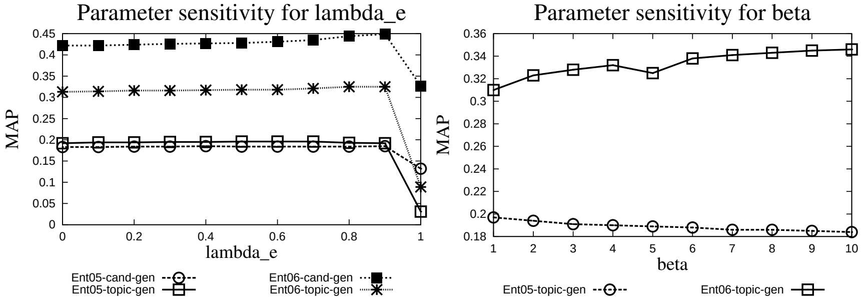

Fig. 2. Performance Sensitivity of λ_e(left) and β (right)

3.6 Comparison with TREC results

Our best results are 0.204 for Ent05 and 0.465 for Ent06 as shown in Table 4. Compared with the official results of the TREC Enterprise track [4, 11], our best results would be in the top 5 for Ent05 and top 10 for Ent06. The top performing systems tend to use various kinds of heuristics, which we did not use. Since our models are general and orthogonal to many of these heuristics, we can expect our models to perform even better when we add these heuristics.

4 Conclusions and Future Work

In this paper, we present a general probabilistic framework to solve the problem of expert finding. We derive two families of generative models, i.e., candidate generation models and topic generation models. These models cover most existing probabilistic models for expert finding. To improve the estimation of the proposed models, we further propose the following three techniques: (1) a mixture model for modeling the candidate mentions, which allows us to put different weights on different representations of an expert candidate; (2) topic expansion for modeling the topic document relationship, which expands the original queries with more informative terms; (3) email-based candidate prior, which provides a better estimation of prior probability that a candidate is an expert. Empirical results have demonstrated the effectiveness of the proposed models and estimation strategies.

There are many interesting future research directions. First, we need to study how to automatically set all the parameters through statistical estimation. Second, we could explore alternative ways to estimate the components in the models. Finally, it would be interesting to study how to estimate a reasonable expertise profile for every expert candidate in a principled way since such profiles can potentially be used for many other tasks in addition to expert finding.

5 Acknowledgments

This material is based in part upon work supported by the National Science Foundation under award numbers IIS-0347933 and IIS-0428472. We thank Lixin Zhou and three anonymous reviewers for their useful comments.

References

1. L. Azzopardi, K. Balog, and M. de Rijke. Language modeling approaches for enterprise tasks. In *Proceedings of TREC-05*, 2006.
2. K. Balog, L. Azzopardi, and M. de Rijke. Formal models for expert finding in enterprise corpora. In *Proceedings of SIGIR-06*, 2006.
3. Y. Cao, J. Liu, S. Bao, and H. Li. Research on expert search at enterprise track of trec2005. In *Proceedings of TREC-05*, 2006.
4. N. Craswell, A. P. de Vries, and I. Soboroff. Overview of the trec-2005 enterprise track. In *Proceedings of TREC-05*, 2006.
5. Y. Fu, W. Yu, Y. Li, Y. Liu, M. Zhang, and S. Ma. Thuir at trec 2005: Enterprise track. In *Proceedings of TREC-05*, 2006.
6. J. Lafferty and C. Zhai. Probabilistic relevance models based on document and query generation. *Language Modeling and Information Retrieval, Kluwer International Series on Information Retrieval*, 13, 2003.
7. C. Macdonald, B. He, V. Plachouras, and I. Ounis. University of glasgow at trec 2005: Experiments in terabyte and enterprise tracks with terrier. In *Proceedings of TREC-05*, 2006.
8. D. Petkova and W. B. Croft. Umass notebook 2006: Enterprise track. In *Proceedings of TREC-06*, 2007.
9. S. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proceedings of SIGIR'94*, 1994.
10. S. E. Robertson. The probability ranking principle in ir. *Journal of Documentation*, 33(4):294–304, Dec. 1977.

11. I. Soboroff, A. P. de Vries, and N. Craswell. Overview of the trec 2006 enterprise track. In *Proceedings of TREC-06*, 2007.
12. E. Voorhees and D. Harman, editors. *Proceedings of Text REtrieval Conference (TREC1-9)*. NIST Special Publications, 2001. http://trec.nist.gov/pubs.html.
13. C. Zhai and J. Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of CIKM-01*, 2001.
14. C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of SIGIR-01*, 2001.