Formal Models for Expert Finding in Enterprise Corpora

Krisztian Balog
ISLA, University of Amsterdam
Kruislaan 403, 1098 SJ
Amsterdam
kbalog@science.uva.nl

Leif Azzopardi
Dept. of Computer and
Information Sciences
University of Strathclyde,
Glasgow G1 1XH
leif@cis.strath.ac.uk

Maarten de Rijke
ISLA, University of Amsterdam
Kruislaan 403, 1098 SJ
Amsterdam
mdr@science.uva.nl

ABSTRACT

Searching an organization's document repositories for experts provides a cost effective solution for the task of expert finding. We present two general strategies to expert searching given a document collection which are formalized using generative probabilistic models. The first of these directly models an expert's knowledge based on the documents that they are associated with, whilst the second locates documents on topic, and then finds the associated expert. Forming reliable associations is crucial to the performance of expert finding systems. Consequently, in our evaluation we compare the different approaches, exploring a variety of associations along with other operational parameters (such as topicality). Using the TREC Enterprise corpora, we show that the second strategy consistently outperforms the first. A comparison against other unsupervised techniques, reveals that our second model delivers excellent performance.

Categories and Subject Descriptors

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

General Terms

Algorithms, Measurement, Performance, Experimentation

Keywords

Expert finding, enterprise search

1. INTRODUCTION

A major challenge within any commercial, educational, or government organization is managing the expertise of employees such that experts in a particular area can be identified. Finding the right person in an organization with the appropriate skills and knowledge is often crucial to the success of projects being undertaken [14]. For instance, an employee may want to ascertain who worked on a particular project to find out why particular decisions were made without having to trawl through documentation (if there is any). Or, they may require a highly trained specialist to consult about a very specific problem in a particular programming language, standard, law, etc. Identifying experts may reduce costs and facilitate a better solution than could be achieved otherwise.

Initial approaches to expert finding employed a database housing the skills and knowledge of each individual in the organization [5, 11]. The database would be manually constructed and consulted. Such approaches required considerable effort to set up and maintain. Consequently, various automated approaches have been devised to mine the information repositories of organizations from which to build a profile of an employee's expertise. Most approaches focus on expert finding in specific domains extracting representations from known document types. More recently there has been a move to automatically extract such representations from heterogeneous document collections such as those found within a corporate intranet [2]. With much of this work performed in industry, details of solutions are restricted to high level designs and products tend to be subjected to in-house evaluations only.

The Text REtrieval Conference (TREC) has now provided a common platform with the Enterprise Search Track for researchers to empirically assess methods and techniques devised for expert finding [17]. The following scenario is presented: Given a crawl of the World Wide Web Consortium's web site, a list of candidate experts and a set of topics, the task is to find the experts for each of these topics.

We propose two models for accomplishing this task which draw upon and formalize existing strategies to expert finding. Our models are based on probabilistic language modeling techniques which have been successfully applied in other Information Retrieval (IR) tasks. Each model ranks candidates according to the probability of the candidate being an expert given the query topic, but the models differ in how this is performed. In our first model, we create a textual representation of the individuals' knowledge according to the documents with which they are associated. From this representation we assess how probable the query topic is to rank candidates. Our second model ranks documents according to the query, and then we determine how likely a candidate is an expert by considering the set of documents associated. Here, the documents act as a latent variable

SIGIR'06, August 6–11, 2006, Seattle, Washington, USA.

between the query and the candidate, and we model the process of finding experts via documents in the collection.

Then, we use the TREC test collection to evaluate and compare these models. This is achieved through a systematic exploration of different types of associations between documents and candidate experts, since such associations are crucial to the models' success. Our main research goals in experimentation are to understand how the quality of associations, along with the different search strategies, impact on the ability to successfully identify candidates.

The remainder of the paper is organized as follows. In Section 2 we briefly discuss related work. Then, in Section 3 we describe two ways of modeling the expert search task. Section 4 is devoted to different candidate extraction methods examined. Next, in Section 5 we present an experimental evaluation of our expert finding methods, using resources from the TREC 2005 Enterprise track. We conclude with a discussion of our main findings in Section 6.

2. SEARCHING FOR EXPERTS

Addressing the problem of identifying expertise within an organization has lead to the development of a class of search engines known as *expert finders* [19]. McDonald and Ackerman [12] distinguish several aspects of expert finding, including what they call expertise identification ("Who are the experts on topic X?") and expertise selection ("What does expert Y know?"). In this paper we are focused exclusively on the first question.

Early approaches to this type of expert search used a database containing a description of peoples' skills within an organization [20]. However, explicating such information for each individual in the organization is laborious and costly. The static nature of the databases often renders them incomplete and antiquated. Moreover, expert searching queries tend to be fine-grained and specific, but descriptions of expertise tend to be generic [10]. To address these disadvantages a number of systems have been proposed aimed at automatically discovering up-to-date expertise information from secondary sources. Usually, this has been performed in specific domains. For instance, there have been attempts to use email communications for expert finding in discussion threads and personal emails. Campbell et al. [1] analyzed the link structure defined by authors and receivers of emails using a modified version of the Hyperlink-Induced Topic Search (HITS) algorithm to identify authorities. They showed that improvements over a content-based approach were possible given two organizations, but the number of candidates used was limited, only fifteen and nine. An alternative approach to using email communications focused on detecting communities of expertise, positing that the signaling behavior between individuals would indicate expertise in a specific area, again using the HITS algorithm [4].

Instead of trying to find expertise residing in email communications others have examined the problem in the context of software engineering development. In [14], rules of thumb were applied to identifying candidates which had expertise on a particular software project and, more specifically, a piece of source code. Such heuristics were generated manually based on the current working practice. E.g., analyzing the source code to see who last modified the code and what part.

Instead of focusing on just specific document types there has been increased interest in systems that index and mine published intranet documents as sources of expertise evidence [7]. Such documents are believed to contain tacit knowledge about the expertise of individuals, as can be seen from the above examples of email and source code. However, by considering more varied and heterogeneous sources such as web documents, reports, and so forth, an expert finding system will be more widely applicable. One such published approach is the P@noptic system [2], which builds a representation of each candidate by concatenating all documents associated with that candidate. When a query is submitted to the system it is matched against this representation, as if it were a document retrieval system. This approach is the most similar to our proposed models. In our Model 1, we do exactly this but formalized in a language modeling framework, where we can weight the association between a candidate and a document instead of naively concatenating documents to form a representation. This can be considered as the baseline for comparison purposes. Our Model 2 (Section 3.4), takes a decidely different approach, where we rely upon how users themselves would find experts given a standard search engine [8] (i.e., look for documents, find out who wrote them, and then contact the author). Our approach automates this process in a principled manner, where we first find documents which are relevant to the query topic and then score each candidate by aggregating over all documents associated to that candidate.

While our work focuses exclusively on core algorithms for expert finding, it is important to realize that expert finders are often integrated into organizational information systems, such as knowledge management systems, recommender systems, and computer supported collaborative work systems, to support collaborations on complex tasks [6]. For a more complete account of expert finding systems we refer the reader to [20].

3. MODELING EXPERT SEARCH

In this section we detail the ways in which we model the expert finding task. First, we provide some background to language modelling applied to Information Retrieval; then we turn our attention to the document-candidate associations that are required for the definition of our two models of expert finding. Model 1 uses candidate models to discover a candidate's expertise, while Model 2 uses document models to get to a candidate's areas of expertise.

In recent years, language modeling approaches to information retrieval have attracted a lot of attention [9, 13, 15]. Language models are attractive because of their foundations in statistical theory, the great deal of complementary work on language modeling in speech recognition and natural language processing, and the fact that very simple language modeling retrieval methods have performed quite well empirically. The basic idea of these approaches is to estimate a language model for each document, and then rank documents by the likelihood of the query according to the estimated language model. In our modeling of expert search we collect evidence for expertise from multiple sources, in a heterogeneous collection, and integrate it with a restricted named entity extraction task—the language modeling setting allows us to do this in a transparent manner.

3.1 Problem Definition and Context

Our approach to expert search assumes that we have a heterogeneous document repository, such as a corporate in-

tranet, containing a mixture of different document types (e.g., technical reports, email discussion, web pages, etc). We assume that a document d in this collection is associated with a candidate ca, if there is a non-zero association $a(d, ca) > 0$. This association may capture various aspects of the relation between a document and a candidate expert; e.g., it may quantify the degree to which this document is representative of the candidate's expertise, or, vice-versa, it may capture the extent to which the candidate is responsible for the document's content. Forming document-candidate associations is a non-trival problem, which we consider in detail later in this paper (Section 4). For now, we present our formal models assuming we have these associations.

We state the problem of identifying candidates who are experts for a given topic, as follows:

> what is the probability of a candidate ca being an expert given the query topic q?

That is, we determine $p(ca|q)$, and rank candidates ca according to this probability. The top k candidates are deemed the most probable experts for the given query. The challenge, of course, is how to estimate this probability accurately. Instead of computing this probability directly, we apply Bayes' Theorem, and obtain

$$p(ca|q) = \frac{p(q|ca)p(ca)}{p(q)},$$

where $p(ca)$ is the probability of a candidate and $p(q)$ is the probability of a query. Thus, the ranking of candidates is proportional to the probability of the query given the candidate $p(q|ca)$.

To determine $p(q|ca)$ we adapt generative probabilistic language modeling techniques from Information Retrieval in two different ways. In our first approach (Model 1), we build a representation of the candidate (i.e., we build a candidate model) using the documents associated with the candidate, and from this model the query is generated. In our second approach (Model 2), the query and candidate are considered to be conditionally independent, and their relation is resolved through the document-candidate associations.

3.2 Document-Candidate Associations

For both Model 1 and Model 2 (still to be defined), we need to be able to estimate the probability that a document d is associated with candidate ca. To define this probability, we assume that non-zero associations $a(d, ca)$ have been calculated for each document and for each candidate. We distinguish two ways of converting these associations to into probabilities, thus estimating their strength. The first, *document-centric* perspective is to estimate the strength of the association between d and ca in terms of the probability $p(d|ca)$. Here, we define

$$p(d|ca) = \frac{a(d, ca)}{\sum_{d' \in D} a(d', ca)}, \quad (1)$$

where D is the set of documents. Intuitively, if we rank documents using (1) (for a given candidate ca), the top documents will be the ones that the candidate expert is most strongly associated with.

In the second, *candidate-centric* way of estimating the strength of the association between documents d and candidates ca, we use the probability $p(ca|d)$, and put

$$p(ca|d) = \frac{a(d, ca)}{\sum_{ca' \in C} a(d, ca')}, \quad (2)$$

where C denotes the set of possible candidate experts. The idea here is this: d is a document produced by our enterprise, and ca is one of the people in the enterprise, who made some kind of contribution to d; when we rank candidates using (2) (for a fixed d), we find the candidate who made the biggest contribution to d.

Observe that there is a proportional relation between the document-centric and candidate-centric views, via Bayes' rule:

$$p(d|ca) = \frac{p(ca|d)p(d)}{p(ca)}.$$

3.3 Using Candidate Models: Model 1

Our first formal model for the expert finding task (Model 1) builds on well-known intuitions from standard language modeling techniques applied to document retrieval. A candidate ca is represented by a multinomial probability distribution over the vocabulary of terms (i.e., $p(t|ca)$). Since $p(t|ca)$ may contain zero probabilities, due to data sparsity, it is standard to employ smoothing. Therefore, we infer a candidate model θ_{ca} for each candidate ca, such that the probability of a term given the candidate model is $p(t|\theta_{ca})$.

We can then estimate the probability of the query being generated by the candidate model θ_{ca}. As usual, the query is represented by a set of terms, such that t is in q if the number of times t occurs in q, $n(t, q)$, is greater than zero. Each query term is assumed to be generated independently, and so the query likelihood is obtained by taking the product across all the terms in the query, such that:

$$p(q|\theta_{ca}) = \prod_{t \in q} p(t|\theta_{ca})^{n(t,q)}.$$

To obtain an estimate of $p(t|\theta_{ca})$, we first construct an empirical model $p(t|ca)$ using our list of associations, and then smooth this estimate with the background collection probabilities. Specifically, within the *document-centric* perspective, the probability of a term given a candidate, can be expressed as

$$p(t|ca) = \sum_d p(t|d)p(d|ca),$$

and under the *candidate-centric* perspective it is expressed as

$$p(t|ca) \propto \sum_d p(t|d)p(ca|d),$$

where $p(t|d)$ is the maximum likelihood estimate of the term in a document. By marginalizing over all documents we obtain an estimate of $p(t|ca)$. The candidate model is then constructed by a linear interpolation of the background model $p(t)$, and the smoothed estimate:

$$p(t|\theta_{ca}) = (1 - \lambda)p(t|ca) + \lambda p(t).$$

Now, if we let $f(d, ca)$ denote either $p(d|ca)$ or $p(ca|d)$ and put together our choices so far, we obtain the following final estimation of the probability of a query given the candidate

model:

$$p(q|\theta_{ca}) = \prod_{t \in q} \left\{ (1-\lambda) \left(\sum_d p(t|d) f(d, ca) \right) + \lambda p(t) \right\}^{n(t,q)} \quad (3)$$

This, then, is Model 1. In words, Model 1 amasses all the term information from all the documents associated with the candidate and uses this to represent that candidate. This model is used to predict how likely this candidate would produce a certain query q, which can be intuitively interpreted as the probabilty of this candidate talking about this topic, where we assume this is indicative of their expertise.

3.4 Using Document Models: Model 2

Instead of directly creating a candidate model as in Model 1, we can compute the probability $p(q|ca)$ by assuming conditional independence between the query and the candidate. Thus, the probability of a query given a candidate can be viewed as the following generative process:

- Let a candidate ca be given.
- Select a document d associated with ca (using either the *document-centric* or the *candidate-centric* approach with probability $p(d|ca)$ or $p(ca|d)$, respectively).
- From this document, generate the query q, with probability $p(q|d)$.

By taking the sum over all documents d, we obtain $p(q|ca)$. Formally, this can be expressed in either a document-centric way, as

$$p(q|ca) = \sum_d p(q|d)p(d|ca),$$

or a candidate-centric way, as

$$p(q|ca) \propto \sum_d p(q|d)p(ca|d).$$

Conceptually, Model 2 differs from Model 1 because the candidate is not directly modelled. Instead, the document acts as a hidden variable in the process, separating the query from the candidate. Under this model, we can think of the process of finding an expert as follows. Given a collection of documents ranked according to the query, we examine each document and if relevant to our problem, we then see who is associated with that document (we assume they have knowledge about that topic). Here, the process is taken to the extreme where we consider all documents in the collection.

To determine the probability of a query given a document, we infer a document model θ_d for each document d. The probability of a term t given the document model θ_d becomes:

$$p(t|\theta_d) = (1-\lambda)p(t|d) + \lambda p(t),$$

and the probability of a query given the document model is:

$$p(q|\theta_d) = \prod_{t \in q} p(t|\theta_d)^{n(t,q)}.$$

The final estimation of Model 2, then, is:

$$p(q|ca) = \sum_d \left\{ \prod_{t \in q} \left((1-\lambda)p(t|d) + \lambda p(t) \right)^{n(t,q)} \right\} f(d, ca), \quad (4)$$

where $f(d, ca)$ is $p(d|ca)$ or $p(ca|d)$, depending on whether we adopt a document-centric or candidate-centric perspective (as with the final estimation of Model 1, cf. Equation 3).

An advantage of Model 2 over Model 1 is that, given a set of document-candidate associations it can easily be implemented on top of a standard document index, whereas Model 1 requires that a separate candidate-term index be created and maintained.

3.5 Using Topicality

So far, we have assumed that all documents in the collection are used in the computations of Model 1 and 2. This implicitly assumes that all the documents associated with a candidate are related and about one particular topic of expertise. Often a candidate will have expertise in several areas. Therefore, it may be more appropiate to only use a subset of the collection for expert finding, those that are related to the query topic at hand.

We can simply obtain a query-biased cluster of documents by submitting a query to the collection and only using the top n documents retrieved for the computations underlying Model 1 and 2. One of our experiments in Section 5 is aimed at understanding the impact on the overall expert finding task of increasing the topicality of the underlying document collection in this manner.

4. BUILDING ASSOCIATIONS

Document-candidate associations form an essential part of the models presented in Section 3. We now turn to the task of building such associations. Specifically, we need to assign non-negative association scores $a(d, ca)$ to pairs of documents d and candidate experts ca.

We assume that a list of possible candidates is given, where each candidate is represented with a unique *person_id*, one or more *names* and one or more *e-mail* addresses. While this is a specific choice, and while different choices are possible (e.g., involving social security number, or employee number instead of, or in addition to, the representations just listed), the representations chosen are generic and nothing in our modeling depends on *this* particular choice.

The recognition of candidates (through one of these representations) is a (restricted and) specialized named entity recognition task. We approach it in a rule-based manner. We introduce four binary association methods `A`i ($i = 0, \ldots, 3$) that return 0 or 1 depending on whether the document d is associated with the candidate ca; the first three attempt to identify a candidate by their name, the last uses the candidate's email address:

- `A0`: `EXACT MATCH` returns 1 if the name appears in the document exactly as it is written.
- `A1`: `NAME MATCH` returns 1 if the last name and at least the initial of the first name appears in the document.
- `A2`: `LAST NAME MATCH` returns 1 if the last name appears in the document.
- `A3`: `EMAIL MATCH` returns 1 if the e-mail address of the candidate appears in document d.

Note that for $i = 1, 2$, the method `A`i extends upon the results achieved by `A`$i-1$, thus increasing recall while (probably) giving up some precision. The results of `A3` are independent from the other three methods.

Since A0–A2 on the one hand, and A3 on the other, identify candidates by orthogonal means, it makes sense to combine extraction methods from the two groups, and to consider linear combinations of their outcomes. This, then, is how we define association scores:

$$a(d, ca) := A_\pi(d, ca) = \sum_{i=0}^{k} \pi_i \mathtt{A}i(d, ca), \quad (5)$$

where $\pi = \{\pi_1, \ldots, \pi_k\}$ and $\sum_{i=0}^{k} \pi_i = 1$. In the following we will use the general form (5) when refering to an association method.

One of the questions we will address below is to which extent the quality of the candidate extraction method (and hence of the document-candidate associations) impacts the task of expert finding.

5. EXPERIMENTAL EVALUATION

We now present an experimental evaluation of our models for expert finding. We specify our research questions, describe our data set, and then address our questions.

5.1 Research Questions

We address the following research questions:

- How effective are our candidate extraction methods? (Subsection 5.3)
- How do different smoothing methods behave given our models and what are their optimal parameters? (Subsection 5.4)
- How do Model 1 and Model 2 perform compared to each other? (Subsection 5.5)
- What is a more effective way of capturing the strength of an association between a document and a candidate: the document centered approach or the candidate centered approach better? (Subsection 5.6)
- What is the impact of the candidate extraction methods on our performance on the expert finding task? (Subsection 5.7)
- Does the combination of extraction methods improve performance? (Subsection 5.8)
- What is the effect of using a topically focused subset of documents on our performance on the expert finding task? (Subsection 5.9)

Each of the research questions above identifies a dimension along which we explore the models we introduced. To remain focused, and because of lack of space, we will usually consider each dimension in isolation and forego exploring large numbers of combinations of parameter settings, etc.

5.2 Test Collection

We use the data set used at the 2005 edition of the TREC Enterprise track [17]. The document collection used is the W3C corpus [18], a heterogenous document repository containing a mixture of different document types crawled from the W3C website. The six different types of web pages were lists (email forum), dev, www, esw, other, and people (personal homepages). In our experiments these were all handled and processed in the same way, as HTML documents. While different documents might require special treatment in order to exploit the richer representation of the content's type, we leave this as future work. The W3C corpus contains 330,037 documents, adding up to 5.7GB.

We took the topics created within the expert finding task of the 2005 Enterprise track: 50 in total; these are the topics for which experts have to be sought. In addition, a list of 1092 candidate experts was made available, where each candidate is described with a uniqe candidate_id, name(s) and one or more e-mail addresses.

Evaluation measures used for the expert finding task were mean average precision (MAP), R-precision, mean reciprocal rank (MRR), and precision@10 and precision@20 [17]. Evaluation was done using the trec_eval program.[1]

5.3 Extraction

How effectively can names be identified (and associations be established) in the document collection, using the extraction mechanisms described in Section 4? This issue can be addressed in an intrinsic way (assessing the extraction component in isolation, which is what we do in the present subsection) and in an extrinsic way (which is what we do in Subsection 5.7).

method	%cand	%rel_cand	#avg	%docs
Extraction by name:				
A0: EXACT MATCH	63.74%	62.34%	466	41.27%
A1: NAME MATCH	69.32%	68.01%	468	42.23%
A2: LAST NAME MATCH	84.62%	83.96%	1023	64.17%
Extraction by e-mail address:				
A3: EMAIL MATCH	41.76%	40.06%	162	17.93%
Combining methods:				
A0 and A3:	66.03%	64.55%	552	42.59%
A1 and A3:	70.51%	69.26%	556	43.44%
A2 and A3:	85.35%	84.73%	1094	64.68%

Table 1: Results of different candidate extraction methods.

Table 1 contains the results of the different candidate extraction methods: %cand denotes the percentage of all possible candidates that have been identified; %rel_cand is the percentage of relevant candidates, that is, the real experts, that have been identified; #avg is the average number of associations (different documents) per candidate. Finally, %docs is the percentage of documents in the data set which have been associated to at least one candidate. The three horizontal sections of the table show the performance of name extraction (top), e-mail extraction (center), and the combination of these methods (bottom).

As expected, the LAST NAME MATCH method has the highest recall of all the methods based on extraction by name. The EMAIL MATCH method seems to pick up a few candidates (and associations) not covered by the name-based methods, and combinations with this method lead to (small) improvements for each of the name-based methods.

5.4 Smoothing

Next, we turn to smoothing. We experimented with two smoothing methods: Jelinek Mercer and Bayes Smoothing [21], and observed that these behave differently in Model 1 and 2. In the case of Model 1 both methods performed similarly, with a slight advantage of Bayes Smoothing, while in case of Model 2, Jelinek Mercer was significantly better.[2]

[1]For registered participants trec_eval is available from the TREC web site http://trec.nist.gov.

[2]To determine whether the observed differences between two

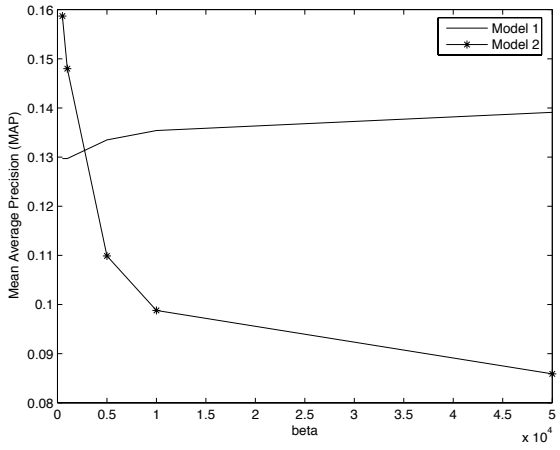

Figure 1: Mean average precision of Model 1 and Model 2, using A0 and different β values.

The optimal β parameter for Bayes Smoothing differs for the two models. Model 1 performed best with a high β value (50000), but for Model 2 it is the opposite: the highest performance was achieved by using low values for β (100); see Figure 1. This behavior can be explained by considering the details of the two models. Low values of β have been found to be optimal for short documents, while high values are known to be optimal for long documents. Model 2 uses normal (short) documents, while for Model 1 we represent each candidate as the sequence of terms in the documents associated with the person—resulting very long documents.

In the following sections we use the Jelinek Mercer smoothing method, with $\lambda = 0.5$. While this is not the best possible setting for all extraction methods, models, and estimations, it is a reasonable setting that provides acceptable performance across the parameter space.

5.5 Models

Next we turn to our core question: which of Model 1 and Model 2 is most effective for finding experts? In the following set of experiments we compare the two formal models introduced in Section 3. A quick scan of Table 2 reveals that Model 2 outperforms Model 1 for nearly all settings. Let us focus on the best performing configurations for both, i.e., the candidate-centric probability estimation using the `A0` extraction method. Figure 2 shows the MAP scores over the 50 topics achieved by the two models; the topics are sorted according to the result achieved by Model 1. Clearly, Model 2 outperforms Model 1 on nearly all topics. The differences between Model 1 and Model 2 are statistically significant, for both estimation methods, and all extraction methods except for `A3`; this may mean that Model 2 performs better only when there are sufficiently many associations.

5.6 Centricity

Which of the two ways of estimating the strength of association between a document and a candidate performs better? Document-centric or candidate-centric? Consider Table 2 again. If we do a cell-by-cell comparison of document-centric and candidate-centric estimation, we see that candidate-centric estimation outperforms document-centric estimation for almost all measures. For Model 2, candidate-centric estimation is significantly better than document-centric estimation, for all extraction methods. For Model 1, we find a significant improvement only in the case of the extraction method `A3`; Model 1 seems insensitive to the probability estimation if there are sufficiently many associations.

Based on the outcomes of our experiments we froze this parameter and use the candidate-centric estimation method in the following parts of the section.

5.7 Extraction Revisited

The effectiveness of different candidate extraction methods in terms of the number of candidates extracted and documents associated, has already been examined in Subsection 5.3. At this point we compare the accuracy achieved by these methods in an extrinsic manner, based on their impact on the effectiveness of expert finding.

As discussed previously, results (candidates and associations) found by extraction the methods `A`i ($i = 0, 1$) are also found by `A`$i+1$. Table 2 shows that for each $i = 1$, 2, the extraction method `A`$i-1$ outperforms the methods `A`i, according to nearly all models, measures, and estimation methods. Moreover, in spite of the low number of identified candidates and associations, `A3` achieves reasonably high precision in several cases. Hence, the quality and not the number of the associations has the main impact on precision. According to our experiments, `A0 (EXACT MATCH)` performs best for building associations. In the followings, we refer to this configuration as a baseline (`BASE`).

	#rel	MAP	R-prec	MRR	P10	P20
Model 1 (candidate model):						
document-centric:						
`A0`	492	0.1221	0.1576	0.3574	0.236	0.209
`A1`	486	0.1138	0.1537	0.3246	0.214	0.204
`A2`	447	0.0919	0.1476	0.3288	0.206	0.175
`A3`	424	0.1234	0.1778	0.4096	0.262	0.194
candidate-centric:						
`A0`	**511**	0.1253	**0.1914**	0.2759	0.236	**0.227**
`A1`	507	0.1189	0.1851	0.2537	0.216	0.206
`A2`	471	0.0951	0.1654	0.2604	0.202	0.186
`A3`	430	**0.1347**	0.1819	**0.4839**	**0.280**	0.208
Model 2 (document model):						
document-centric:						
`A0`	560	0.1731	0.2245	0.4783	0.284	0.241
`A1`	554	0.1670	0.2243	0.4590	0.280	0.237
`A2`	513	0.1294	0.1902	0.4195	0.228	0.214
`A3`	430	0.1222	0.1768	0.4770	0.238	0.187
candidate-centric:						
`A0`	**580**	**0.1880**	**0.2332**	**0.5149**	**0.316**	**0.260**
`A1`	575	0.1790	0.2262	0.4958	0.296	0.254
`A2`	543	0.1537	0.2173	0.4872	0.274	0.235
`A3`	439	0.1337	0.1934	0.4898	0.256	0.205

Table 2: Results of the different models, extraction methods and document/candidate-centric probability estimations. The columns of the table are: extraction method, number of relevant retrieved candidates, mean average precision, R-precision, mean reciprocal rank, precision after 10 and 20 candidates retrieved. Best results are in bold face.

expert finding approaches are statistically significant, we use Student's t-test, and look for significant improvements (one-tailed) at a significance level of 0.95 [16].

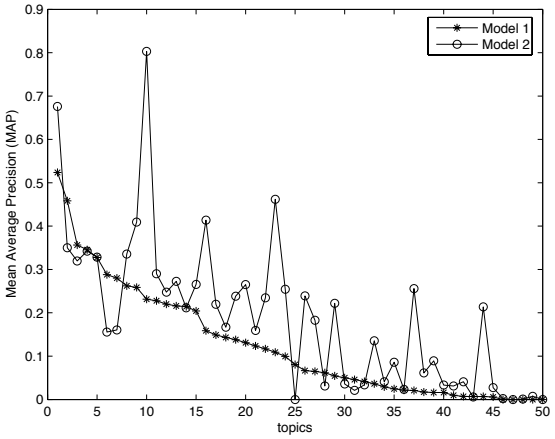

Figure 2: Comparison of Model 1 and Model 2, using A0 extraction method and candidate-centric probability estimtion.

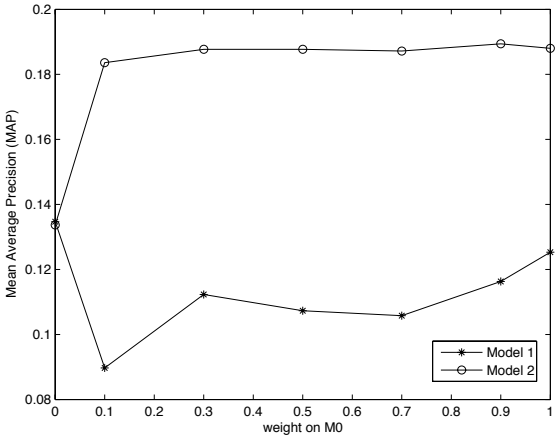

Figure 3: The impact of varying the weight on A0 (π_0). Note that the left ($\pi_0 = 0$) and right ($\pi_0 = 1$) boundaries of the plot refers to a single extraction method A3 and A0, respectively.

	#rel	MAP	R-prec	MRR	P10	P20
Model 1 (candidate model):						
A3	430	**0.1347**	0.1819	**0.4839**	**0.280**	0.208
A0	511	0.1253	**0.1914**	0.2759	0.236	**0.227**
COMB	**514**	0.1163	0.1785	0.3358	0.190	0.199
Model 2 (document model):						
A3	439	0.1337	0.1934	0.4898	0.256	0.205
A0	580	0.1880	0.2332	**0.5149**	**0.316**	**0.260**
COMB	**590**	**0.1894**	**0.2434**	0.5043	**0.316**	**0.260**

Table 3: Results of combining name extraction (A0) and e-mail extraction (A3). Candidate-centric estimation.

5.8 Combining Extraction Methods

Does a combination of candidate extraction methods increase performance on the expert finding task? A combination could consist of any number of non-zero associations (4); we restricted ourselves to combinations of the best performing name extraction method (A0) and the e-mail extraction method (A3). Let π_0 the weight on A0 and π_3 the weight on A3, where $\pi_0 + \pi_3 = 1$. Figure 3 shows the effect of varying the weights on these methods. Table 3 contains detailed results of the best performing combination (COMB) with weights $\pi_0 = 0.9$, $\pi_3 = 0.1$ compared to A0 and A3.

Our experiments show promising but mixed results. Combinations of extraction methods did not achieve improvements on all measures. The optimal weights are rather ambiguous; the two models and the various measures might involve different π values, however the number of relevant retrieved candidates is increased in case of both models. The significant difference observed here was for Model 2 with the combination vs A3.

5.9 Topicality

The aim of this, our final set of experiments is to find out how the topicality of documents used to build the representations influences the performance on the expert finding task. Instead of using the full collection, we use a subset of documents defined by taking the top n documents returned by a standard document retrieval run when using the topic as query. Table 4 shows the results achieved by using different values for n, the document cut-off. Note that BASE corresponds to $n = |D|$, in other words, models are built on the full document set.

	#rel	MAP	R-prec	MRR	P10	P20
Model 1 (candidate model):						
BASE	**511**	**0.1253**	**0.1914**	0.2759	**0.236**	**0.227**
n=10000	474	0.1025	0.1654	0.2988	0.190	0.173
n=5000	484	0.0973	0.1494	0.2732	0.176	0.165
n=1000	483	0.1092	0.1703	0.3128	0.186	0.183
n=500	474	0.1035	0.1609	0.3245	0.188	0.173
n=100	443	0.1113	0.1699	**0.4037**	0.192	0.173
Model 2 (document model):						
BASE	**580**	**0.1880**	0.2332	0.5149	**0.316**	0.260
n=10000	**580**	0.1869	0.2330	0.5140	**0.316**	0.257
n=5000	**580**	0.1878	0.2336	0.5146	**0.316**	0.257
n=1000	566	0.1868	0.2379	0.5040	**0.316**	**0.263**
n=500	527	0.1845	**0.2384**	0.5310	0.312	0.261
n=100	380	0.1712	0.2244	**0.5954**	**0.316**	0.234

Table 4: Results of using a subset of top documents

For Model 1, the use of a restricted subset of documents was not of benefit to retrieval across the set of thresholds applied. In contrast, for Model 2 we witnessed improvements in some cases. Importantly, the time needed for model building is proportional to the number of documents. Using a subset of documents could speed up the process of expert finding significantly while the desired accuracy could be controlled by adjusting the value of n (size of the subset). None of the differences observed between the runs for Model 2 was significant. This means that topicality does not hurt performance in the case of Model 2: using a restricted set of documents improves responsiveness.

5.10 Comparison to Other Systems

Compared to the runs submitted by the TREC 2005 Enterprise Track participants [3], our best results would be in the top 5. Note that for a fair comparison it is important to take into account how others approached the task. Unlike some of the top 5 performing approaches, our models are unsupervised; no manual efforts were made to increase the

performance. Moreover, unlike some other systems we did not make any assumptions with regard to the data collection and the topics. In particular, we did not resort to a special treatment of some of the documents (such as e.g., discussion lists), and we did not utilize the fact that the test topics were names of W3C working groups. In our approach the former could be exploited, e.g., to build high-quality associations, whilst the latter approaches the task as a "working group finding" task rather than an expert finding task and would, we believe, suffer from not be generalizable beyond searching for W3C working groups; with our approach we can search for experts on any topic.

6. CONCLUSION AND FURTHER WORK

We focused on developing models for searching an organization's document repositories for experts on a given topic using generative probabilistic models. We explored our models along many dimensions; evaluations of these models were performed on the W3C TREC Enterprise collection.

We found that Model 2 performs better than Model 1 on practically all measures. However, this is not the only reason that leads us to favor Model 2. Various things we have tried out behaved more according to our expectations on Model 2 than on Model 1 (see e.g., in terms of how additions to the extraction method worked (or did not work) according to expectation). On top of that, Model 2's online behavior is much better than Model 1's: Model 2 produces the results in a reasonable time which makes it useable even in an online application. It does not require a separate index to be created, like Model 1, but can immediately be applied to an indexed collection, given a set of associations. In practical terms it means that it could be implemented on top of a standard search engine with very limited effort, only requiring a list of candidate-document associations.

Looking forward, possible improvements might be pursued in the named entity extraction (NE) component of the system. E.g., looking at other ways of forming associations, using NE recognition and trying to ascertain whether that person wrote the document, or is the owner of the document (e.g., their personal website). It would also be interesting to extract not only individuals, but also groups and communities and methods for identifying people within these.

7. ACKNOWLEDGMENTS

This research was supported by the Netherlands Organization for Scientific Research (NWO), project 220-80-001.

8. REFERENCES

[1] C. S. Campbell, P. P. Maglio, A. Cozzi, and B. Dom. Expertise identification using email communications. In *CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management*, pages 528–531. ACM Press, 2003.

[2] N. Craswell, D. Hawking, A. M. Vercoustre, and P. Wilkins. P@noptic expert: Searching for experts not just for documents. In *Ausweb*, 2001. URL: http://es.csiro.au/pubs/craswell_ausweb01.pdf.

[3] N. Craswell, A. de Vries, and I. Soboroff. Overview of the trec-2005 enterprise track. *TREC 2005 Conference Notebook*, pages 199–205, 2005.

[4] R. D'Amore. Expertise community detection. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 498–499. ACM Press, 2004.

[5] T. H. Davenport and L. Prusak. *Working Knowledge: How Organizations Manage What They Know*. Harvard Business School Press, Boston, MA, 1998.

[6] F. Hattori, T. Ohguro, M. Yokoo, S. Matsubara, and S. Yoshida. Socialware: Multiagent systems for supporting network communities. *Communications of the ACM*, 42(3): 55–61, 1999.

[7] D. Hawking. Challenges in enterprise search. In *Proceedings Fifteenth Australasian Database Conference*, 2004.

[8] M. Hertzum and A. M. Pejtersen. The information-seeking practices of engineers: searching for documents as well as for people. *Inf. Process. Manage.*, 36(5):761–778, 2000.

[9] D. Hiemstra. *Using Language Models for Information Retrieval*. PhD thesis, University of Twente, 2001.

[10] H. Kautz, B. Selman, and A. Milewski. Agent amplified communication. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96)*, pages 3–9, 1996.

[11] M. E. Maron, S. Curry, and P. Thompson. An inductive search system: Theory, design and implementation. *IEEE Transaction on Systems, Man and Cybernetics*, 16(1):21–28, 1986.

[12] D. W. McDonald and M. S. Ackerman. Expertise recommender: a flexible recommendation system and architecture. In *CSCW '00: Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pages 231–240. ACM Press, 2000.

[13] D. Miller, T. Leek, and R. Schwartz. A hidden Markov model information retrieval system. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 214–221, 1999.

[14] A. Mockus and J. D. Herbsleb. Expertise browser: a quantitative approach to identifying expertise. In *ICSE '02: Proceedings of the 24th International Conference on Software Engineering*, pages 503–512. ACM Press, 2002.

[15] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *SIGIR '98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 275–281, New York, NY, USA, 1998. ACM Press.

[16] M. Sanderson and J. Zobel. Information retrieval system evaluation: effort, sensitivity, and reliability. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 162–169. ACM Press, 2005.

[17] TREC. Enterprise track, 2005. URL: http://www.ins.cwi.nl/projects/trec-ent/wiki/.

[18] W3C. The W3C test collection, 2005. URL: http://research.microsoft.com/users/nickcr/w3c-summary.html.

[19] D. Yimam. Expert finding systems for organizations: Domain analysis and the demoir approach. In *ECSCW 999 Workshop: Beyond KNowledge Management: Managing Expertise*, pages 276–283, New York, NY, USA, 1996. ACM Press.

[20] D. Yimam-Seid and A. Kobsa. Expert finding systems for organizations: Problem and domain analysis and the demoir approach. *Journal of Organizational Computing and Electronic Commerce*, 13(1):1–24, 2003.

[21] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 334–342. ACM Press, 2001.