Using Transactional Data from ERP Systems for Expert Finding

Lars K. Schunk[1] and Gao Cong[2]

[1] Dynaway A/S, Alfred Nobels Vej 21E, 9220 Aalborg Øst, Denmark
[2] School of Computer Engineering, Nanyang Technological University, Singapore
schunk@gmail.com, gaocong@ntu.edu.sg

Abstract. During the past decade, information retrieval techniques have been augmented in order to search for experts and not just documents. This is done by searching document collections for both query topics and associated experts. A typical approach assumes that expert candidates are authors of intranet documents, or that they engage in social writing activities on blogs or online forums. However, in many organizations, the actual experts, i.e., the people who work on problems in their day-to-day work, rarely engage in such writing activities. As an alternative, we turn to structured corporate data—transactions of working hours provided by an organization's ERP system—as a source of evidence for ranking experts. We design an expert finding system for such an enterprise and conclude that it is possible to utilize such transactional data, which is a result of required daily business processes, to provide a solid source of evidence for expert finding.

Key words: Expert Finding, Information Retrieval, Enterprise Resource Planning, ERP

1 Introduction

In many information-intensive organizations, one of the most prominent organizational challenges is the management of knowledge, and the ability to locate the appropriate experts for any given information need is essential. In small organizations, locating an expert may be a simple matter of asking around. However, in large organizations with several specialized departments, which may even be geographically scattered, this approach becomes infeasible.

A traditional solution is to maintain a database of employees and skills where each employee fills in his or her experience, skills, and fields of specialization [7]. This approach has some rather demotivating disadvantages. First, it requires a great deal of resources to maintain. Each employee will often be responsible for updating his or her profile in order to keep it current, which requires a very dedicated organization staff. Second, because of this human factor, the system is subject to imprecision, partly due to employees' over- or underrating of their skills, and partly due to a mismatch in granularity between profile descriptions, which tend to be mostly general, and queries, which tend to be specific.

During the past decade, *expert finding systems* have emerged. The purpose of such a system is to automate the process of associating people with topics by analyzing information that is published within the organization, such as task descriptions, reports, and emails. From a user point of view, it typically is a variant of a traditional search engine; the user inputs a query topic, but instead of retrieving a set of relevant documents, it retrieves a set of relevant people—supposedly experts on the topic suggested by the query.

An important aspect of expert finding systems is the association between documents and expert candidates. Existing expert finding approaches typically assume variations on the following points: 1) candidates write textual content such as papers or forum posts; 2) if the name, email address, or other identifier of a candidate appears in such a document, then that document is related to the candidate. In short, they assume that expert candidates are creators of information.

However, such assumptions can be problematic. E.g., the overview of workshop [2] states, among other things, that when "looking at the chain of emails in which a request for expertise is passed from one person to another, it is also clear that mere candidate mentions do not necessarily imply expertise."

Furthermore, in many organizations, the actual experts, i.e., the people who work on problems in their day-to-day work, are too busy to engage in such writing activities. In these settings, expert finding approaches such as those mentioned above are of limited use. In [8], for instance, it is noted that less than 10% of a workforce studied were engaged in writing blogs. Though this figure has been increasing, it likely has a natural limit far below 100%. Nevertheless, employees are expert candidates even if they are not active creators of textual information, and it would be useful to capture their expertise to facilitate expert finding. In this paper, we therefore disregard the assumptions above, noting that certain types of documents are written without direct candidate annotation. Instead, we turn to other means of forming associations between documents and candidates.

We explore the potential of enterprise resource planning (ERP) systems, such as Microsoft Dynamics AX, as a source of expertise evidence. Many organizations maintain enormous amounts of transactional data in such ERP systems. To each record of transactional data it is possible to attach textual documents, but such documents often do not contain candidate information within their textual contents. Reasons for this include: 1) context is captured by the structured data that surrounds the documents in the ERP system; 2) documents are initially written without any specific people in mind, and then later different people are associated with the documents through the organization's various business processes.

Our enterprise setting is that of thy:data,[3] a Danish software consulting and development company within the area of business solutions primarily based on Microsoft's ERP system Dynamics AX.[4] thy:data employs Dynamics AX for project management, and this system acts as our case in this paper. We believe

[3] `http://www.thydata.dk`
[4] `http://www.microsoft.com/dynamics/ax`

that thy:data is representative of many companies dealing with similar consulting and development services.

We focus on transactions of hours worked on projects by candidates as our evidence of expertise. To our knowledge, this is the first work to exploit transactional data from ERP systems as expertise evidence for expert finding. We employ two methods for ranking candidates; one based on the classic TF-IDF ranking approach, and the other based on language modeling approaches [3].

One of the major benefits of our approach is that we do not depend on active knowledge-sharing from the employees because we leverage information that is created by required daily business processes within the organization, namely registration of working hours spent on activities. This is in contrast to other sources of evidence, such as corporate blogs and discussion forums, which require that expert candidates engage in knowledge-producing activities of a more voluntary nature, as is used in most previous work (e.g., [1, 3, 8, 13]).

From a more pragmatic point of view, our work in this paper shows good opportunities for implementing expert finding systems that integrate directly with modern ERP systems. This could be done by developing an expert finder as an integral module of an ERP system. Such an effort would provide several benefits including: 1) direct access to all ERP data for establishing expertise evidence; 2) easy integration with human resource modules, etc.; 3) easy access to expert finding for all daily users of the ERP system, thus boosting their productivity with minimal extra effort.

2 Related Work

The expert finding task introduced in the TREC Enterprise track [5] in 2005 has generated a lot of interest in expert finding, and a number of approaches have been developed. One of the central issues is how to establish a connection between candidates and topics. Usually, expertise evidence is found by analyzing documents that somehow relate to the expert candidates.

The P@NOPTIC system [6] presents a simple approach in which this connection is established by building an *expert index*, which consists of *employee documents*—one document is created for each employee. The employee document representing a given employee is the concatenated text of all intranet documents in which that employee's name occurs. With the employee documents in place, the system can match queries against the expert index using any standard information retrieval technique, and retrieve in ranked order the employee documents that match. With the one-to-one correspondence between employee documents and employees, it is easy to go from matching employee document to relevant employee.

Nearly all systems that took part in the 2005 and 2006 editions of the expert finding task at TREC[5] adopted a language modeling approach (e.g., [9]),

[5] Text REtrieval Conference: `http://trec.nist.gov`

first introduced by Balog et al. [3]. Based on the idea in [10] of applying language modeling to information retrieval, Balog et al. rank candidates by their probabilities of generating a given query.

The association between candidates and documents can be refined in various ways. E.g., instead of capturing associations at document level, they may be estimated at snippet level around occurrences of candidate identifiers. Such use of proximity-based evidence has been found to achieve better precision in general. However, Balog et al. [4] note that "the mere co-occurrence of a person with a topic need not be an indication of expertise of that person on the topic." E.g., the name of a contact person may be mentioned in many documents and thus frequently co-occur with many topics, but the contact person is not necessarily an expert on the topics.

Recently, "Web 2.0 data," such as that provided by blogs and discussion forums, has been incorporated into expert finding systems based on the assumption that documents that generate much Web 2.0 data due to user activity are more interesting than documents that spawn only little activity, and that the users who exhibit activity around a document are related to the document [1, 8, 13]. As is also pointed out in [1], these systems cannot retrieve candidate experts who have never blogged or commented on another user's activity.

Apparently, there has not been much research that disregards the presumption that experts—in one way or another—are authors of document content. Furthermore, to our knowledge, neither has transactional context provided by ERP systems been the subject of expert finding research in the past. In this paper, we aim to capture the expertise of people who do not directly produce textual content by utilizing structured data from the organization's ERP system.

3 Corporate Transactional Data

We want to utilize transactional data to form document-candidate associations, and we will completely disregard the fact that candidate information may exist within the documents. Thus, document content is only used for matching query topics, just as in a traditional information retrieval system. When finding related experts, we want to rely entirely on transactional data from an ERP system.

Many companies maintain structured information about their employees, including how many hours they have spent working on different activities. If hours worked on activities can be allocated to documents, then this would seem a good starting point for establishing document-candidate associations necessary for expert search. Let us consider what it means when an employee has worked a large number of hours on some task. We can interpret this fact in at least two ways: 1) The task requires much work, and this employee has developed a valuable degree of expertise within the topics of the task. Thus, we assume that people who work on a task become knowledgeable on topics relevant to the task. This way we capture expertise even if the experts are too busy to write and publish their knowledge. We use this assumption in our basic models. 2) The employee has had difficulty completing the task because he is not an expert on the topic. The employee still may be a relevant person because he has spent time on the task and may have some valuable insight on the problems. However, if much of our evidence falls into this category, we may have to adjust the models. Therefore, after having presented the basic models, we propose an extension to take this potential shortcoming into account.

3.1 Enterprise Data Setting

At thy:data, they store *task descriptions* in an operational management system. These documents contain specifications for desired software functionality. Typically, a document describes one well-defined function of a larger system and represents a single unit of work that usually can be completed by one or two employees. The task descriptions are written by consultants, and afterwards a software developer must be assigned to the task. The employees who work on a task register their work hours in the system. Thus, the operational management system contains both structured data (hours worked) and unstructured data (task descriptions).

All of thy:data's activities are organized into a *project hierarchy*. There is a number of top-level projects, and each project contains a number of sub-projects, which in turn can contain sub-projects themselves, etc. Each project can have a number of *activities* associated with it. An activity usually represents a certain well-defined task such as a software development task. The activities have various textual data associated with them. This includes descriptions of the task that the activity represents, as well as notes written by the people who have worked on the activity. The textual data can be stored directly in dedicated fields in the database or in elaborate documents outside of the database. Besides textual data, activities have transactions associated with them. Such a transaction represents a number of working hours that a certain employee has spent on the activity.

3.2 A Model of the Data

We can view the activities as constituting a central entity that ties together employees and documents. The employees are connected to the activities via transactions, and the activities are connected to the documents. We can view the *hours worked* measure on the transactions as an indicator of how strongly a given employee is associated with a given document.

These entities can be modeled as a weighted bipartite graph with two disjoint sets of vertices: a set of documents and a set of employees. Weighted edges between the two sets are derived from the activity and transaction entities. An example of this is shown in Figure 1. Here we see that document d_1 is associated with employee e_1 because e_1 has worked 60 hours on the activity to which d_1 is attached.

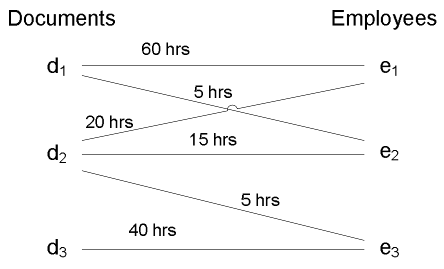

Fig. 1. Hours worked as a weighted bipartite graph with documents and employees.

4 Method Design

We propose two basic models for ranking candidates based on the hours worked measure described in Section 3. The two models are variations on two well-known approaches, namely the TF-IDF approach and the document language modeling approach. Both of our models rely on document-candidate associations, so this aspect will be discussed first.

4.1 Document-Candidate Associations

A central part of the document language model described in [3] are the document-candidate associations, which provide a measure of how strongly a candidate is associated with a document. Given a collection of documents D and a collection of candidates C, to each pair (d, ca), where $d \in D$ and $ca \in C$, a non-negative association score $a(d, ca)$ must be assigned such that $a(d, ca_1) > a(d, ca_2)$ if candidate ca_1 is more strongly associated with document d than candidate ca_2.

Balog et al. provide an $a(d, ca)$ measure by using a named entity (NE) extraction procedure that matches identification information of candidate ca with document d. E.g., if ca's email address occurs in d, then $a(d, ca) > 0$.

We want to replace this $a(d, ca)$ measure with one that takes hours worked into account instead of using NE extraction. To do this, we formalize the model of the hours worked that was developed in an intuitive manner in Section 3.2. Let D be the set of documents, and C the set of candidates. Let $G = (V, E)$ be a bipartite graph where $V = D \cup C$ is the set of vertices and $E = \{\{d, ca\} \mid d \in D \text{ and } ca \in C\}$ is the set of edges. To each edge $\{d, ca\} \in E$ we assign weight $w(d, ca)$ as follows:

$$w(d, ca) = \text{total number of hours worked on } d \text{ by } ca \tag{1}$$

Now we can introduce a simple document-candidate association measure $a(d, ca)$ in place of the one that was presented in [3]:

$$a(d, ca) = \begin{cases} w(d, ca) & \text{if } \{d, ca\} \in E \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

4.2 Modifying the TF-IDF Approach

In the classic TF-IDF approach used in traditional information retrieval, we calculate the relevance $r(d, Q)$ of a document d to a query Q as follows:

$$r(d, Q) = \sum_{t \in Q} TF(d, t) IDF(t) \tag{3}$$

where $TF(d, t)$ is the term frequency of term t in document d and $IDF(t)$ is the inverse document frequency of term t [11].

Now we modify this measure so that we can rank candidates. We want to establish a measure of the relevance $r(ca, Q)$ of a candidate ca to a query Q, much like the measure $r(d, Q)$ above. Suppose we have found the *one and only* document d that is relevant to the query Q. Then we can define the relevance of candidate ca to Q like this:

$$r(ca, Q) = a(d, ca) \tag{4}$$

The candidate who has worked the most hours on document d will be the top ranked candidate in terms of relevance to Q. However, many documents may be relevant to Q, some more than others. If we use the relevance of documents $r(d, Q)$ as weights on the document-candidate associations, we have the following definition of $r(ca, Q)$:

$$r(ca, Q) = \sum_{d \in D} r(d, Q) a(d, ca) \tag{5}$$

where D is the set of all documents in the collection. The more relevant documents that a candidate has worked on, the more likely it is that he is a relevant candidate, which we take into account by summing over all documents.

4.3 Modifying the Document Language Modeling Approach

In the document language modeling approach introduced in [3], the ranking of a candidate is calculated as the probability of that candidate generating a query. This is expressed as follows:

$$P(Q|ca) = \sum_{d \in D_Q} P(Q|d) P(d|ca) \tag{6}$$

where

$$P(Q|d) = \prod_{t \in Q} ((1 - \lambda) P_{mle}(t|d) + \lambda P(t|D)) \tag{7}$$

is the probability of the query Q given document d's language model by employing the Jelinek-Mercer smoothing method [12], and

$$P(d|ca) = \frac{a(d, ca)}{\sum_{d' \in D} a(d', ca)} \tag{8}$$

is the probability of document d given candidate ca. Put simply, given candidate ca, the document d with highest probability $P(d|ca)$ will be the document with which ca is most strongly associated.

We can tailor this approach to the present setting by simply replacing the document-candidate association measure $a(d, ca)$ with another one that relies on hours worked instead of the rule-based method using NE extraction. Having provided such a substitute in Section 4.2, it is straightforward to plug this into the document language model.

4.4 Extending the Basic Models

Now that the basic models are in place, we will consider some possible extensions. One could imagine some adjustments to the document-candidate association measure presented in Section 4.1. Consider the following scenario. Suppose that, given a query Q, the set D_Q are deemed relevant documents. Furthermore, candidates ca_1 and ca_2 are deemed relevant candidates. ca_1 has worked hundreds of hours on just one relevant document while ca_2 has worked moderate numbers of hours, say 10–20, on several relevant documents. Which candidate is more relevant? By Equation 5, ca_1 is likely to score higher because the hundreds of hours worked on one document will boost his score significantly. But the work on this *single* document may represent an exception. In contrast, if someone has worked moderate numbers of hours on *several* relevant documents, this may reflect the fact that he actually is an expert who completes his tasks quickly.

To take this into account, we introduce another measure, *document count*, denoted by $dc(ca, D_Q)$, which is the number of relevant documents in D_Q to which candidate ca is associated:

$$dc(ca, D_Q) = |\{d \mid d \in D_Q \text{ and } a(d, ca) > 0\}| \tag{9}$$

We can extend Equation 5 with the document count measure:

$$r(ca, Q) = dc(ca, D_Q) \sum_{d \in D_Q} r(d, Q)a(d, ca) \tag{10}$$

Likewise, we can extend Equation 6:

$$P(Q|ca) = \frac{dc(ca, D_Q)}{|D_Q|} \sum_{d \in D_Q} P(Q|d)P(d|ca) \tag{11}$$

where the document count has been converted to a probability.

Applying these to the example scenario above would boost ca_2's relevance score to reflect the fact that he has worked on several relevant tasks even if the total hours worked are less than those of ca_1.

5 Evaluation

As identified in workshop [2], a major challenge in expert finding research is obtaining real data for evaluation purposes. Currently, existing data sets are built from publicly accessible pages of organizational intranets [2]. These collections do not contain alternative non-textual sources of evidence such as transactional data from ERP systems used in our work, so they are not applicable to our evaluation purposes. The lack of both annotated resources and relevant queries renders the evaluation of our approaches particularly challenging.

For now, our prototype works on a document collection from the Aalborg department of thy:data, which consists of 1319 documents and 28 employees. To get an idea of the effectiveness of the system, we interviewed some key employees, posed a set of ten expertise queries, and noted the corresponding expert employees. We fed the queries to the system and observed how well the candidate experts ranked at different cutoff points. The results are shown in Table 1 for P@1, P@3, and P@5 (precision at rank 1, 3, and 5). For the language modeling approach, we set $\lambda = 0.5$ by following previous work [3]. We tested the TF- IDF approach both with and without the document count (DC) extension described in Section 4.4.

Generally, these results indicate that the system is fairly accurate with most results being above 70%. The table also shows a slight improvement when we apply the document count extension to the TF-IDF approach. It may seem surprising that the TF-IDF approach performs slightly better than the language modeling approach because the latter is generally considered superior. We note that this is not a full-scale evaluation, which to this point has not been feasible for this project. However, the primary objective of this work was to facilitate expert finding when expertise evidence is *not* available within documents. This objective has been fulfilled. The approaches taken are based on previous results that have performed well in full-scale empirical studies. This constitute the "subjective" aspect of this work. By augmenting these approaches to take hours worked into account, we have added an "objective" aspect. Given the assumption that hours worked are correlated with level of expertise, we can safely incorporate this measure when ranking the employees.

Approach	P@1	P@3	P@5
Lang. Model without DC	0.700	0.733	0.665
TF-IDF without DC	0.900	0.833	0.670
TF-IDF with DC	1.000	0.867	0.710

Table 1. P@1, P@3, and P@5 for TF-IDF and language modeling approaches.

6 Conclusion and Future Work

We proposed an approach for capturing expertise of people when we cannot rely on the assumption that expert candidates are creators of information. Companies often maintain structured data that may indicate associations between documents and candidates. We utilized one such type of structured data, namely hours worked, so that we no longer need to rely on the assumption that candidate information exists *within* documents, an assumption that may not always be warranted. Our basic models are simple and apparently effective. Because registration of hours worked is a required daily business process, it becomes a solid source of evidence, which even captures expert candidates who may never have written a single document.

We discussed potential shortcomings of using hours worked as expertise evidence, and we proposed one extension to account for these. We can think of more ways to improve precision of expert finding by using structured ERP data. E.g., measures such as employee seniority or salary class may be used as weights on the hours worked, assuming that hours worked by experienced employees are more indicative of expertise than hours worked by newcomers. Furthermore, we may consider the recency of hours worked, assuming that recent transactions imply recently applied expertise, whereas very old transactions may be ignored. Finally, the approach presented here could be incorporated into traditional approaches in order to increase general expert finding precision.

References

1. E. Amitay, D. Carmel, N. Golbandi, N. Har'El, S. Ofek-Koifman, and S. Yogev. Finding people and documents, using web 2.0 data. In *Proceedings of SIGIR Workshop: Future Challenges in Expertise Retrieval*, pp. 1–5, 2008.
2. K. Balog. The sigir 2008 workshop on future challenges in expertise retrieval (fcher). *SIGIR Forum*, 42(2):46–52, 2008.
3. K. Balog, L. Azzopardi, and M. de Rijke. Formal models for expert finding in enterprise corpora. In *Proceedings of ACM SIGIR*, pp. 43–50, 2006.
4. K. Balog and M. de Rijke. Non-local evidence for expert finding. In *Proceedings of CIKM*, pp. 489–498, 2008.
5. N. Craswell, A. de Vries, and I. Soboroff. Overview of the trec 2005 enterprise track. In *The 14th Text REtrieval Conference Proceedings (TREC 2005)*, 2006.
6. N. Craswell, D. Hawking, A.-M. Vercoustre, and P. Wilkins. P@noptic expert: Searching for experts not just for documents. In *Poster Proceedings of AusWeb*, 2001.
7. T. Davenport and L. Prusak. *Working Knowledge: How Organizations Manage What They Know.* Harvard Business School Press, 1998.
8. P. Kolari, T. Finn, K. Lyons, and Y. Yesha. Expert search using internal corporate blogs. In *Proceedings of SIGIR Workshop: Future Challenges in Expertise Retrieval*, pp. 7–10, 2008.
9. D. Petkova and W. B. Croft. Hierarchical language models for expert finding in enterprise corpora. In *Proceedings of ICTAI*, pp. 599–608, 2006.
10. J. Ponte and W. Croft. A language modeling approach to information retrieval. In *Proceedings of ACM SIGIR*, pp. 275–281, 1998.
11. A. Silberschatz, H. Korth, and S. Sudarshan. *Database System Concepts, 5th Edition.* McGraw-Hill, 2006.
12. C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of ACM SIGIR*, pp. 334–342, 2001.
13. Y. Zhou, G. Cong, B. Cui, C. Jensen, and J. Yao. Routing questions to the right users in online communities. In *Proceedings of ICDE*, pp. 700–711, 2009.