Judging Expertise–WIM at Enterprise

Chen Lin
Department of Computer Science and Information Technology
Fudan University Shanghai 200433 P.R.C.
cheyenne.lin@gmail.com

Junyu Niu
Department of Computer Science and Engineering
Fudan University Shanghai 200433 P.R.C.
jyniu@fudan.edu.cn

February 7, 2007

Abstract

This document reports experiment and result of Fudan WIM group in Expert Search track in TREC 2006. Our research mainly focus on the measurement of expertise. Inspired by the human procedure of expert search, we construct 2 models, a language model and a social network model are compared. The association function of expert and document is modified. And email search techniques are employed in document retrieval.

1 Introduction

People often turn to experts when they want to solve some problems. The behavior of expert search is labor costly and frustrating. Hence an automated system assisting knowledge and expertise management is strongly needed. The goal of Expert Search is to conduct experiment with enterprise data and offer a platform of evaluation for researchers.

This is the first time we participate in Expert Search task. As a starter we refer to the experience of people finding experts in real life. Suppose that Brian is a software engineer facing a problem in his project, now he want to find someone with appropriate knowledge and skills to help him out. Generally there are two ways of searching experts: one is to search in social network, which is: ask someone for a possible recommendation and pass on. Brian should prudently choose the next person he asked, he may choose who most likely to know the potential expert. The other way is to search in the document repositories. In this case, Brian would prefer certain document types–technical reports, essays, presentations, which provide clear evidence of expertise. Both the two method is applicable. Hence our

experiment this year models the two procedure of human activity, one model is a language model, the other is a social network model.

The paper is organized as follows: Section 2 gives our model and framework. Section 3 describes the technical details of our system. Section 4 presents our results. Section 5 is the conclusion of this paper.

2 Model

The scenario of expert search task is: given a list of candidate — 1092 candidates in total, an enterprise corpora — the partial package of w3c web pages before 2004, and a list of topical queries — 55 topics in the form of topic, description and narrative, the system should return a ranked list of experts in each query, and a ranked list of supporting documents for each experts.

Hence we see the key point of expert search task: How can we judge expertise? A possible solution is to look for relevant documents, and find candidates who have made some contribution to the documents. Based on language model theory, we have *model 1*:

$$p(c \mid t) = \Sigma p(c \mid d, t) p(d \mid t) \tag{1}$$

Here $p(c \mid t)$ defines the probability of candidate c being an expert of topic t, $p(d \mid t)$ defines the probability of document d relevant to topic t, and $p(c \mid d, t)$ defines the strength of candidate c associated to document d in topic t. The candidate-document score $p(c \mid d, t)$ should quantitatively demonstrate the relation between the candidate and the document, e.g.: how much the document represent the candidate expertise, or how much is the candidate responsible to the document's context. To do so, we define

$$p(c \mid d, t) = \begin{cases} 1 & \text{if } c \text{ is the author of } d \\ \frac{tf(c|d)}{\Sigma_c(c|d)} & \text{else} \end{cases}$$

From 2 we see that if a candidate c is the author or editor of document d, c is fully responsible to the context of d. Otherwise, we should consider the frequency of c in d and the total frequency of all experts in d.

Recall the procedure of Brian searching for experts, we notice that social factors– authorities, collaborations, co-authorships can play an important part in judging experts. *model 2* introduce those social factors to rank the candidates. We use email communications, citations and co-occurrence in documents to build an expertise network.

A *Expertise Network* is an edge-weighted, directed graph $G(V, E, S, W)$, where every vertex $v_i \in V$, each $e_{ij} \in E$ is an edge from v_i to v_j. $w_{ij} \in W$ is the weight of e_{ij}, $s_i \in S$ is the score of v_i

In fact, every candidate has two scores, *authority score* and *active score*. The *authority score* s_i shows the authority of candidate c_i in social network, and *active score* c_i shows the existing experience of candidate in documents. Generally,

a candidate with more publications is more active, a candidate work with more experts is more authoritative.

In the submitted runs, we use language model's result to initialize social network model. That is, we assign the relevant score in language model as active score in social network, document relevancy in language model as the strength of communication panel in social network. And finally we re-rank the candidates by their authority score.

Therefore we give the algorithms

- *step 1:pre-process* choose specific types of document: technical reports, essays, etc.
- *Step 2:candidate recognition* identify the candidates in documents, assign each document-candidate pair an association score.
- *Step 3:document retrieval* use lemur to retrieve relevant document in corpora, assign each document a relevant score.
- *Step 4:expert ranking* rank experts by the product of document score and document-candidate score.
- *Step 5:social network* use social network to re-rank experts.

3 Framework

Our system consists of three parts, candidate recognition, pre-processing and expert ranking modules. In expert ranking module, we use the two models we described above to measure expertise for each candidate. Figure 1 shows the framework of our system.

3.1 Candidate Recognition

In expert search task, we are given a list of candidates including candidates The purpose of candidate recognition is to identify the candidate's occurrence and compute the weight of candidate-document pair $p(c \mid d, t)$.

TREC provides us a candidate list including each candidate's full name and official email address. However, this is not enough. A candidate can appear in a document in 4 forms.

- *case 1: exact match* the candidate's full name appears in the document exactly as it is.
- *case 2: last name match*only the candidate's last name appears in the document. Abbreviation or nickname is used to replace the original first name.
- *case 3: email match*only the candidate's email address appears in the document. The email address may not be the official one in the candidate list.

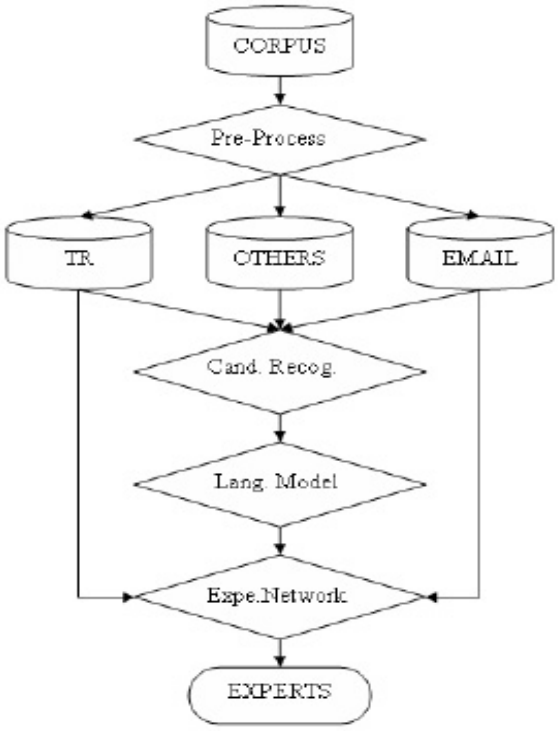

Figure 1: system framework

A candidate could appear as an author, an editor, a reviewer, or a reference. As described above, we distinguish the author and editor of a document. We approach it in a rule-based manner. We implement the candidate recognition procedure in the following 4 steps:

- *step 0* start from the candidate list, for each candidate, we have tuple $< c_id, c_fullname, email_1 >$
- *step 1* exactly match the full name of candidates. For each document, we have tuple $< d, c_1, c_2, , , c_n >$.
- *step 2* employs a nickname dictionary to match the nicknames and abbreviations, if the candidate's last name appears. After the step, we got tuple $< c_id, c_fullname, c_nickname, c_abbname >$.For each document, we update tuple $< d, c_1, c_2, , , c_n >$.
- *step 3*we go through the w3c email archives, if a candidate c_i appears in tuple $< d_j, c_1, c_2, , , c_n >$, and if c_i writes d_j from a different address, we update the tuple as $< c_id, c_fullname, c_nickname, c_abbname, email_1, email_2, , , email_n >$
- *step 4*repeat step 3 and step 4 until no tuple $< d, c_1, c_2, , , c_n >$ is updated.

url	type
lists-***-*******	mail archive
www-***-*******	www pages
dev-***-*******	development archives, codes
esw-***-*******	esw, wiki
people-***-*******	people archives
other-***-*******	others

Table 1: w3c corpus

- *step 5*mark the position of each occurrence of candidate, a 10-word window of context before the position is extracted to form a feature vector $< f_1, f_2, , , f_{10} >$of document-candidate pair $< d_j, c_i >$.If feature f_i is "author", "editor" or other keywords we predefined, the c_i is the author or editor of document d_j.

3.2 Pre-processing

In our model, we assume that some document types explicitly manifest the candidate's expertise , which are: technical reports, presentations(slides) and w3c working drafts. Since the w3c web site is organized hierarchically, publications are congregated and can be easily found. Text classification methods are also used here to identify the specific document types.

- *Step 1:URL analysis* we assign documents with url prefix "www.w3.org/TR/" as positive samples.
- *Step 2: feature selection* the context between html head tag (¡h1¿,¡h2¿...¡h6¿) are extracted as features of the document.
- *Step 3: text classification* use svm to classify all the documents in www pages. The svm is trained by positive samples gained by step 1.

3.3 Document Retrieval

The w3c corpus consists of 6 parts.

In our system, the email archives and web pages are treated differently. Emails are not formal texts, for better understanding, we should analyze email's characters. We clean their quotes, combine all the context in one thread.

4 Result

We submit 3 runs: *FDUSF* , *FDUSO*, *FDUSN*.

- *FDUSF* full corpora

run-tag	map	p5	p10	p100
fduSF	0.4706	0.6898	0.6082	0.2014
fduSO	0.4814	0.7020	0.6306	0.2033
fduSN	0.4672	0.6898	0.6143	0.2027

Table 2: overall performance of submitted runs

- *FDUSO* only technical reports
- *FDUSN* social network

As our result shows, the performance of submitted runs differ in topics. However *FDUSO* and *FDUSN*has a higher overall precision and map than *FDUSF*, which suggest the affect of different document type in judging expertise. In several particular topics, our system achieved lower score than median results. We believe that query expansion will improve our performance in these topics.

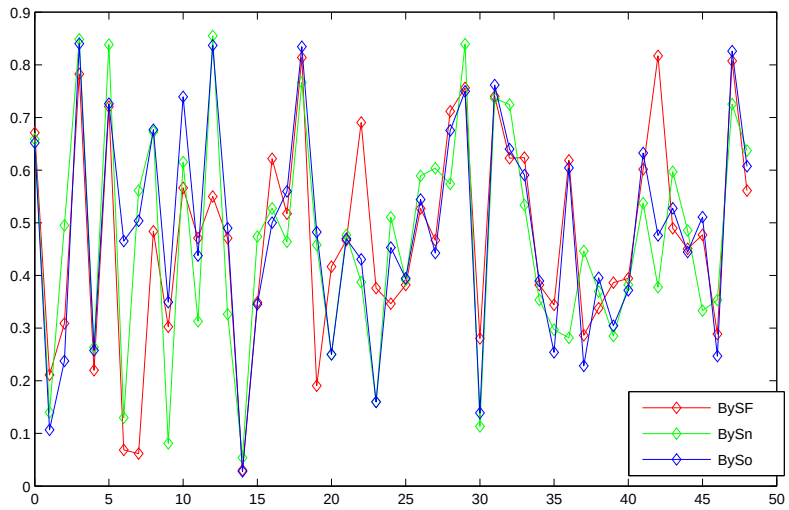

Figure 2: submitted runs on map

5 Conclusions

From the result we can conclude that: both language model and social network model yield good precision. The main contribution of our system is exploring the potential of combining social network and standard IR methods. However, in our submitted runs, the ranking algorithm of social network model is noisy and depends

too much on the result of text retrieval model. Our future work remain on mining the structure and latent semantic of expertise network.

References

[1] K.Balog, L.Azzopardi and M.D.Rijke Formal Models for Expert Finding in Enterprise Corpora *ACM Trans. Program. Lang. Syst.*, 15(5):795–825, November 1993.

[2] D. W. McDonald and M. S. Ackerman. Just talk to me: A field study of expertise location. In CSCW, pages 315C324, 1998.

[3] N. Craswell, A. de Vries, and I. Soboroff. Overview of the trec-2005 enterprise track. TREC 2005 Conference Notebook, pages 199C205, 2005.

[4] M. Herlihy. A methodology for implementing highly concurrent data objects. *ACM Trans. Program. Lang. Syst.*, 15(5):745–770, November 1993.

[5] L. Lamport. *LaTeX User's Guide and Document Reference Manual*. Addison-Wesley Publishing Company, Reading, Massachusetts, 1986.

[6] S. Salas and E. Hille. *Calculus: One and Several Variable*. John Wiley and Sons, New York, 1978.

[7] S.Wasserman and K.Faust Social Network Analysis:Methods and Applications. Cambridge University Press, 1994

[8] Finding experts and their details in e-mail corpora,

[9] J.Zhang and M.S.Ackerman Searching For Expertise in Social Networks: A Simulation of Potential Strategies Group '05

[10] 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.