AUTORITY AWARE EXPERT SEARCH: ALGORITHM AND SYSTEM FOR NSFC

[1]Xiaoyun Mai, [2]Guiguang Ding, [2]Jianmin Wang

[1]Department of Computer Science and Technology, Tsinghua University, Beijing, China
[2]School of Software, Tsinghua University, Beijing, China
even1027@gmail.com; {dinggg, jimwang}@tsinghua.edu.cn

Keywords: expert finding, language model, authority prior, application code, expertise modelling

Abstract

This paper describes the peer reviewer finding algorithm used in an aided system to assist the proposal reviewing task in the National Science Foundation of China (NSFC)[1]. We propose a new probabilistic language model in which expert authority is taken into consideration, by introducing a prior probability of candidate into the model. Application codes representing research areas are defined by NSFC and are available for both experts and proposals. We integrate code information into our model as we add a code probability term into the model, thus promote rankings of experts with matched code. All proposals submitted to NSFC are written in Chinese, and since Chinese word segmentation is non-trivial task whose accuracy affects the final results heavily, we try to improve segment accuracy by adding domain-specific terms to a user-editable dictionary. Terminologies are extracted from certain field of bibliography data in NSFC system. Experiments show that our algorithm is effective in real world system like NSFC.

1 Introduction

The National Science Foundation of China provides financial support to promising research topics in a variety of fields in natural science. Tens of thousands of proposals are submitted to NSFC each year but in fact only a small proportion of them really get the funding. Whether a proposal is worthy of being funded is decided via peer reviewing, where some experts with expertise from the field of the proposal give comments on the proposal independently and all opinions are combined to form the final decision. Finding experts with specialized knowledge given a proposal is essential in the reviewing process, for only experts from specific research area are able to give evaluations to proposals timely and accurately.

The workflow of proposal reviewing in NSFC is quite similar to that of conferences and journals, where proposals are first grouped into panel. After that, for each of such groups, experts with expertise in certain research area are found and asked to evaluate all proposals in the group for convenience of making comparisons within group. At present both proposal grouping and expert finding are done by NSFC staff making use of their empirical knowledge. As both of these tasks are quite verbose and time-consuming, they hope that a computer-aided system can be built, to help increase work efficiencies. We therefore develop such a system for NSFC using information retrieval techniques to help both tasks, and we focus on expert finding algorithm we use in the system in this paper.

Expert finding has been studied for long and has received wide attention recently since it is included in TREC [4] from 2005 to 2008. Expert finding is aimed at finding relevant experts given specific research areas. Input of expert finding algorithms can be in the form of query terms or documents, and a ranked list of experts is returned as output. A great deal of algorithms and models have been proposed to deal with the problem, in which statistical language models are most widely used, as they can achieve agreeable results while the computational complexity remains low. One representative work of language model was proposed by Balog [3] as the author formalized finding experts relating to query $\boldsymbol{q}$ as the probability of a candidate $\boldsymbol{ca}$ being an expert given $\boldsymbol{q}$, i.e., $\boldsymbol{p(ca|q)}$. This conditional probability can be translated to another probability $\boldsymbol{p(q|ca)}$ using Bayes' formula on assumption that the probability distribution of experts are uniform. However, this assumption does not hold in reality, as we suppose two experts whose expertise are equally related to query $\boldsymbol{q}$ with different authorities, we would rather choose the one with higher authority. Therefore, we introduce a candidate prior probability into our model where a combinational metric of applying and reviewing history is exploited.

Each proposal submitted to NSFC has 1 or 2 application codes attached, and experts archived in NSFC system also claim the research fields they are familiar with by application codes. An expert who claims an expert in the field of a proposal by codes is more likely to be an expert to the proposal. So when finding experts for a given proposal, the code information need to be put into consideration. We add code probability in calculating the associations between proposal and experts so as to promote rankings for those whose codes coincide with the proposal.

During early experiments we found that the result is not quite satisfactory and as we check the word list generated after word segmentation we found that there are terms wrongly separated by the tokenizer, and thus the similarities were miscalculated. The Chinese word tokenizer we use is for general purpose so not many domain-specific terms were included in the original model, however, it allows user-defined dictionary to be added when doing the segmenting. We add terms extracted from the proposal repository to the model, such that wrongly-separated words were kept as a whole after the segmentation.

We evaluate the performance of three different models on the NSFC 2008 dataset. One basic model where no candidate prior or code similarity are included, and two modified models with knowledge of authority level and application code integrated. Experiment results show that our models outperform the basic one in terms of precision and DCG metric.

The rest of this paper is organized as follows. In section 2 we briefly summarize related works. In section 3 we introduce the basic language model and several improvements we made to it. Experiment results are given in section 4 and we conclude in section 5.

2 Related Work

Sometimes when we seek for solutions in some specific areas, it is not very likely that we get satisfying answers from general use search engines, and in such situations we need to seek help from experts in certain fields. Finding experts with expertise in specific field is what ***expert search*** aims to do, as introduced by TREC Enterprise track [4] in 2005. Expert search is widely used in both industrial and academic areas, in which matching paper with reviewers is one important application in academic fields and a lot of studies and systems have been proposed. As expert search is usually formalized as an information retrieval task, many typical IR models have been applied to this field.

Vector Space Model is a fundamental IR model and is used by Hettich [2] in his system built for National Science Foundation of the United States. The author represented both proposal and reviewer by a TF-IDF vector. Experts whose expertise vectors have higher coverage degree of the given proposal ***p***'s vector are regarded as reviewers of ***p*** with higher probability.

Many different probabilistic models have also been put to use in expert search. Balog [3] for the first time state the problem of associating experts for a query as the probability of a candidate ***ca*** being an expert given query ***q***, i.e., ***p(ca|q)***. This probability can be converted to p(q|ca)*p(ca)/p(q) using Bayes' Formula. As the query prior p(q) is irrelevant in ranking experts and the candidate prior p(ca) is treated as uniform, ranking experts can thus be done according to ***p(q|ca)***. Balog proposed two models for estimating p(q|ca): candidate model where each candidate is represented by a pseudo document; document model where real documents serve as bridge between queries and candidates. Mimno [5] further developed Balog's work to three models for calculating p(q|ca), which are single-document, max-document and document-sum models.

Advanced models such as topic models have also been applied in the field of expert search. Author-topic model [6] is one of those pioneering works in using generative model to deal with expert search, in which content of documents and the interests of authors are simultaneously modeled in a single model. Each document is represented by a multinomial of topic, which is determined by the authors of it. The multinomial distributions of authors are learned from sampling the training data. Various improvements have been made to Author-Topic model, such as Mimno's Author-Persona-Topic model [5], Tang's Author-Conference-Topic Model [7] and so on.

Although topic model outperform statistical language model in many ways, its high computational complexity is a bottleneck for being employed in our system for practical use. We choose to use language model in our system; however, we believe that authority information of experts needs to be considered, that more authoritative experts should have higher probability of being selected by the algorithm. To the best of our knowledge, however, in current studies most language models assume the candidate prior to be uniform. Authority information can only gets involved in ranking experts in subsequent refining steps after the initial scores are generated by language model, and relationships among experts are necessary. Serdyukov's multi-step random walk model on user-document bipartite graph is one representative work of this kind whose time-complexity is also high [10]. Thus we try to modify the language model itself in this work to integrate both authority and similarity factors in one single model. In this way we achieve high precision when the computational complexity remains low.

3 Authority-aware Expert Search

Using language model in expert search is proved effective by many studies. The probability of candidate ca being an expert to query q, p(ca|q), is regarded the ranking criteria for finding experts for q. As we search experts for a group of proposals, we actually need to determine p(ca|g), which is:

$$p(ca|g) = \frac{p(g|ca)p(ca)}{p(g)} \propto p(g|ca)p(ca) \quad (1)$$

As each proposal in one group is assumed to be independently generated, Equation (1) can be further transformed as:

$$p(g|ca) = \prod_{p \in g} p(p|ca)$$

$$p(ca|g) \propto p(ca) \prod_{p \in g} p(p|ca) \quad (2)$$

To determine the candidate prior p(ca) we make use of one's applying and reviewing history in NSFC. And the conditional probability p(p|ca) is calculated using generative probabilistic language models, with application code information included. Details on how to estimate these two probabilities are given in Section 3.2 and 3.3 respectively, and before that we introduce the data resource we use to model expert expertise in Section 3.1. Improvement strategies on Chinese Word Segmentations are given in Section 3.4.

3.1 Representing Expertise

In order to model the expertise of an expert, usually we use a set of documents which can represent one's research interests. Typically, published papers are used and it was mentioned in Hettich's [2] job that proposals funded in previous years have several advantages over publications, which is more suitable for organizations like NSF. As NSFC keeps records of funded projects in previous years, we follow Hettich's idea and use previous projects to represent experts.

However, data sparsity is one big concern as out of more than 13,424 experts in the area of information technology recorded by NFSC, only 7,357 of them have ever applied science fund. Nevertheless, NSFC provides information indicating experts' research interests, in forms of keywords and system-defined application codes. With supplementary knowledge, altogether 9,280 candidates can be represented in the form $e=\{P_e, K_e, C_e\}$ with at least one of P, K and C not null. P represents the set of funded projects where e appears as project leader and K and C stands for the keyword and discipline code set respectively. Expertise document of e is a concatenation of P_e and K_e.

3.2 Modeling the Candidate Prior

Given a proposal p and two candidate experts e_1 and e_2 whose associations with p are equal, i.e., $p(p|e_1)=p(p|e_2)$ in Equation (2), then the ranking of e_1 and e_2 is determined by candidate prior p(e). Let's say e_1 is a more authoritative expert than e_2, then e_1 should have higher probability score than e_2. Previous work which assumes all candidates uniformly distributed fails to take this authority information into considerations, though.

We introduce a weighted candidate prior here as we estimate experts' authority level based on both applying and reviewing history in NSFC system. The system archives 16,298 funded proposals and review information of them. For those who had application history in NSFC, the average number of projects funded is 2.22. And among them 2,108 experts had served as reviewers in NSFC, and they had reviewed an average of 10.3 proposals. Statistics on applying and review history of experts are given in Figure 1.

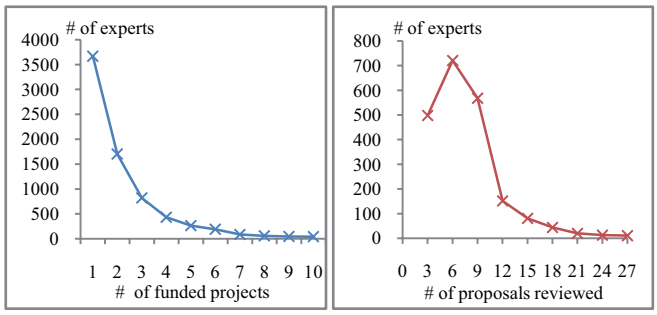

Figure 1: # of proposals applied and reviewed by experts

Figure 1 displays a logarithmic decrease of expert number as the number of proposals increase, hence we choose logarithm functions to transform the number of proposals into the level of authority. We define an ***authority score*** a_e for expert ***e*** as in Equation (3) as rp_e stands for the number of proposals ***e*** had reviewed and fp_e denotes the number of projects funded by NSFC where expert ***e*** serves as project leader. First term in the equation is regarded as the ***review authority*** as the more proposals one reviewed the more experience and thus higher authority he should have. Second term models one's ***research ability***, as those who had successfully applied science funding are regarded more authoritative than those who hadn't.

$$a_e = \log_2(2 + rp_e) * \ln(e + fp_e) \quad (3)$$

We give different base numbers in calculating the two factors because we believe review authority has higher significance than research ability in terms of proposal reviewer. Constants are added to both terms to make sure that a_e is no less than one for all experts even when he had never applied projects or reviewed proposals in the NSFC system.

The probability of an expert ***e*** being chosen from all experts is proportional to his authority score. Thus the candidate prior p(ca) is obtained by normalizing a_e on the whole expert set as shown in Equation (4).

$$p(e) = \frac{a_e}{\sum_{e \in E} a_e} \cdot a_e \quad (4)$$

We refer to the authority-aware model with candidate prior as **MA** and the basic model where candidate prior is ignored as **MB**. We give evaluations on both models in Section 4.

3.3 Proposal-Candidate Associations

We follow the single-document author model in [5] and build a document d_e for each expert ***e*** which is a concatenation of all proposals he had been funded together with keywords in his research area which he filled in the system himself. As terms of proposal are generated independently, we obtain the likelihood of an expert ***e*** generating proposal ***p*** as in Equation (5) where θ_e denotes the smoothed language model of expert ***e***. We follow Balog [3] to use Jelinek Mercer smoothing [9] and also follow him to set λ to 0.5, as in Equation (6).

$$p(p|e) = \prod_{t \in p} p(t|\theta_e)^{n(t,p)} \quad (5)$$

$$p(t|\theta_e) = (1 - \lambda)p(t|d_e) + \lambda p(t) \quad (6)$$

Besides text information of title, keyword and abstract, every proposal has 1 to 2 application codes attached, indicating its research field. Application codes are defined by NSFC system and represent a hierarchical organization of disciplines. As we mentioned in Section 3.1, experts recorded in NSFC also have a group of application codes they filled in themselves to state that they are familiar with the research fields the codes stand for. Suppose a proposal ***p*** whose application code is ***c***, and we have two experts e_1 and e_2 having equal term likelihood with ***p*** using Equation (5). e_1 claims that ***c*** is a code he is familiar with while e_2 doesn't, thus we of course would like to rank e_1 higher in results. We introduce a code generative probability $p(C_p|C_e)$ to model the probability of expert ***e*** being an expert of proposal with code ***c***. This probability is multiplied to term likelihood to get the final results as shown in Equation (7).

$$p(p|\delta_e) = p(p|e) \cdot p(C_p|C_e) \quad (7)$$

Application codes are organized in a tree structure: a complete code has 7 characters and a child code indicates a branch field under parent category. For example F02 denotes the discipline of *computer science* while F0202 indicates *computer software* and F020202 indicates *software engineering*. We regard codes such as F02, F0202 and F020202 as 1st, 2nd and 3rd level code from here on in this paper. Details of NSFC code structure can be found on NSFC website [1].

Intuitively, two proposals sharing the same code are regarded similar to each other, while this similarity may be influenced by the range of this code as predefined codes in NSFC differ in their coverage. Heuristically, the more limited area a code ***c*** stands for, the more likely two proposals having c is similar. Thus, the code generating probability should be in negative correlation with the coverage of the code. Thus we define the code generating probability $p(C_p|C_e)$ as in Equation (8) where p(c) is the probability of code ***c*** among all codes, the number of proposals with c divided by total number of proposals. We set λ to 0.5 here as we do in smoothing the language model.

$$p(C_p|C_e) = \max_{c_1 \in C_p, c_2 \in C_e} \{(1 - \lambda)p(c_1|c_2) + \lambda p(c_1)\} \quad (8)$$

Some proposal writers, as well as experts would like to use 2nd level codes to indicate research area, and we believe prefix matched code also indicates certain degree of association. 1st level codes, however, are discarded in this part as they are too ambiguous to make sense of. When c_1 is a 3rd level code and c_1=c_2 or c_2 is parent of c_1, we have $p(c_1|c_2)$=1, and when c_1 is a 2nd level code and parent of c_2, we estimate $p(c_1|c_2)$ using (9) and otherwise we have $p(c_1|c_2)$=0.

$$p(c_1|c_2) = \frac{\#\text{ of proposals with } c_2}{\#\text{ of porposals with } c_1 \text{ or its children codes}} \quad (9)$$

The new model with application code taken into consideration will be called **MC** from here on, and evaluations on all three models are given in Section 4.

3.4 Refining Chinese Word Segmentation

Chinese word segmentation is a non-trivial task and even the state-of-art technique is far from optimal. When we check up the wordlist generated after processing the whole corpus, we find that many terminologies are not recognized as they were splitted into multiple parts, which will harm the performance of term likelihood using Equation (5). We use a shared toolkit ICTCLAS [8] to do Chinese word segmentation which allows user-editable term dictionary so we add all terms which has 2 to 5 characters appearing in the keyword field of proposals in current and previous years to the term dictionary, so that they won't be splitted by the tokenizer. For example, a 5-character term *directed acyclic graph* will be splitted into five separated Chinese characters by the original general purpose tokenizer, while it can be regarded as a whole term after domain-specific terms were added. There are altogether 15,111 terms added to the user dictionary in our system.

4 Experiments

We test our expert recommendation algorithm on the NSFC 2008 dataset on the discipline of computer science (F02). It is generally believed that evaluating the quality of expert search automatically is hard and usually human judgements are used as criteria. We follow this in our paper and human annotation of 2(highly relevant), 1(relevant) and 0(irrelevant) is obtained making use of both NSFC data and supplementary knowledge we crawled from a bibliography website C-DBLP [11].

We recommend 10 experts with highest rankings for proposal group in real workflow and we here evaluate the performance of expert search of individual proposal instead, as proposals are independent from each other in one group. The algorithm performs differently in terms of popularity level of proposal topic: for popular topic, relevant experts are easier to find and the precision tend to be higher, while for unpopular ones, few experts match with the topic and performance declines. Thus we carry out our experiments on two proposal sets **P** and **UP** standing for popular and unpopular topic respectively, each of them containing proposals randomly selected from the whole corpus. We evaluate all the three models in terms of p@n and Discounted Cumulative Gain (DCG). Experimental results are shown in Figure 2.

	P-MB	P-MA	P-MC	UP-MB	UP-MA	UP-MC
p@1	1.0	1.0	1.0	0.90	0.90	0.80
p@5	0.90	0.90	0.90	0.76	0.78	0.82
p@10	0.79	0.80	0.81	0.62	0.65	0.67
DCG	7.432	7.526	7.627	5.581	5.713	5.985

Figure 2: Evaluations on three models

We can see that both precision and DCG value gets improved when we add authority information and code information step by step. Note that p@1 and p@5 of **P** are not much affected, and that's because higher ranked expert are usually dominant in term likelihood and thus will not be re-ranked by additional reweighting terms. While lower ranked experts tend to have closer similarities in basic model and thus are more likely to be influenced by added authority and code information. And that's why the performance gets much more improved in term of p@10 and further positions.

We also notice that our algorithms outperform the basic one much more obviously on proposal set **UP** than **P**. This is easy to understand as very few experts get matched with unpopular proposal in term likelihood, for not having applied for similar projects in the past. While those relevant experts will indicate their research interest by code information and their authority will be high, and thus will be ranked higher in our model.

5 Conclusions

Expert search has been studied for years and language models are most widely used in practical applications. The authority of experts, however, was rarely considered in previous model and we try to deal with this problem in this paper. An expert prior is introduced to our model making use of applying and reviewing history of researchers and thus we can rank more authoritative experts higher. We also take code information into consideration in our model to promote rankings for those experts with matched codes. Experimental results show that our model achieves agreeable results on the NSFC dataset.

Acknowledgements

This work is supported by National Natural Science Foundation of China (Grant No. 61050010 and No. 60972096).

References

[1] http://www.nsfc.gov.cn/nsfc/cen/daima/index.htm
[2] S. Hettich and M. J. Pazzani. Mining for proposal reviewers: lessons learned at the national science foundation. In KDD, pages 862–871, 2006.
[3] K. Balog, L. Azzopardi, M. de Rijke. Formal models for expert finding in enterprise corpora. In SIGIR, pages 43–50, 2006.
[4] http://trec.nist.gov/data/enterprise.html TREC Enterprise track, 2005-2008.
[5] D. M. Mimno and A. McCallum. Expertise modeling for matching papers with reviewers. In KDD, pages 500–509, 2007.
[6] M. Rosen-Zvi, T. Griffiths, M. Steyvers and P. Smith. The author-topic model for authors and documents. In AUAI, pages 487–494, 2004.
[7] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su. Arnetminer: Extraction and mining of academic social networks. In SIGKDD, pages 990–998, 2008
[8] http://ictclas.org/ ICTCLAS 官方网站
[9] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In ACM SIGIR, pages 334–342, 2001.
[10] P. Serdyukov, H.Rode, D.Hiemstra. Modeling multi-step relevance propagation for expert finding. CIKM, 2008.
[11] http://www.cdblp.cn/ Academic Search in China.