Quality-aware skill translation models for expert finding on StackOverflow

Arash Dargahi Nobari, Mahmood Neshati, Sajad Sotudeh Gharebagh

Faculty of Computer Science and Engineering, Shahid Beheshti University, G.C.

Abstract

StackOverflow has become an emerging resource for talent recognition in recent years. While users exploit technical language on StackOverflow, recruiters try to find the relevant candidates for jobs using their own terminology. This procedure implies a gap which exists between recruiters and candidates terms. Due to this gap, the state-of-the-art expert finding models cannot effectively address the expert finding problem on StackOverflow. We propose two translation models to bridge this gap. The first approach is a statistical method and the second is based on word embedding approach. Utilizing several translations for a given query during the scoring step, the result of each intermediate query is blended together to obtain the final ranking. Here, we propose a new approach which takes the quality of documents into account in scoring step. We have made several observations to visualize the effectiveness of the translation approaches and also the quality-aware scoring approach. Our experiments indicate the following: First, while statistical and word embedding translation approaches provide different translations for each query, both can considerably improve the recall. Besides, the quality-aware scoring approach can improve the precision remarkably. Finally, our best proposed method can improve the MAP measure up to 46% on average, in comparison with the state-of-the-art expert finding approach.

Keywords:
Expertise Retrieval, Statistical Machine Translation, Semantic Matching, StackOverflow, Expert Finding, Word Embedding

Email addresses: a.dargahinobari@mail.sbu.ac.ir (Arash Dargahi Nobari), m_neshati@sbu.ac.ir (Mahmood Neshati), s.sotudeh@mail.sbu.ac.ir (Sajad Sotudeh Gharebagh)

1. Introduction

Nowadays community question answering (CQA) websites have gained a lot of interest among people due to their capabilities in solving different kinds of problems. Over the recent years, a swift growth in the number of users of these networks has been tracked. The popularity of these networks can be noticed by observing the traffic of the renowned CQA websites such as StackOverflow[1], Quora[2], and Yahoo! Answers[3]. Currently, with the growing resource of information, CQA websites provide users a valuable platform for information sharing and searching [1]. Users can contribute and interact by posting questions and answers, commenting, voting, and etc. Additionally, in some CQAs such as StackOverflow they can mark the best answer among provided answers which bring about the concept of accepted answer.

The vital key to the success of CQA platforms is the users who can provide high-quality answers to the more challenging questions posted in community [2]. In recent years, many studies have been made to address the expertise retrieval as a superior Information Retrieval (IR) task. Indeed, expert finding has recently attracted much attention in IR community [3, 4, 5] and become a well-studied field. The task of expert finding is defined as detecting a set of persons with relevant expertise for the given query [6].

Expert finding has many real-world applications. One of its trending applications is talent acquisition which benefits organizations significantly [7]. By analyzing historical data, recruiters can detect potential experts to evolve their organization's business. Moreover, as a part of revenue model, CQA platforms such as StackOverflow aim to find experts on different topics and then propose them to organizations [8, 9]. These examples demonstrate the high importance of expertise retrieval task.

As mentioned before, expert finding is a well-established study in the field of IR and simultaneously it is a challenging task. In recent years, several research studies have been conducted in different domains including CQA platforms [10], bibliographic networks [7], and organizations [11]. Evidently, the task of expert finding is not completely the same in these

[1] stackverflow.com
[2] quora.com
[3] answers.yahoo.com

domains. Although some similarities exist between these domains, there are some remarkable disparities. The most critical affinity is that document associated with a candidate is the most noticeable evidence of his/her expertise on the subject of related topic and expertise level of the candidate can be estimated using some properties like vote (score) of the document. As in [6], the quality of a paper is estimated based on citation count of the paper. Nonetheless, in CQA platforms, the vote count of a document cannot merely show expertise of the author. Rather, it can also represent the popularity and novelty of the subject [12]. Moreover, in CQA platforms, the questions and the answers contain technical aspects of the language, hence in many cases, there is a deep gap between the main query and the language which is used by the expert people of the question-related community. As an illustration, consider the "Android" query. As expert users of the Android community do not use the term itself directly in their answers and instead they use some terms like "fragment" and "broadcastreceiver" which have a direct relation with "Android", we can say that a deep gap exists between the "Android" as main query and the language used by experts.

In a typical CQA community, each question has one or more tags which indicate the required skills to answer that question. These tags can basically be considered as skill areas which recruiters are interested in. For example, consider the following question on StackOverflow, "What is the difference between JPA and Hibernate?". This question can be tagged by "jpa", "hibernate", "java-ee", and "orm" (i.e. Object Relational Mapping) which are important skill areas in Java programming language. In this paper, our goal is to detect and rank users who are skillful in a given skill area (tag) with respect to the title and body of the questions which are given.

The state-of-the-art models proposed by Balog *et al.* [13] can be used in order to rank expert users in StackOverflow. In these models, the question's associated tags can be considered as the main queries, as mentioned earlier, the body of the provided answers can be regarded as his/her evidence of author's expertise. The pitfall of these models is the vocabulary gap existed between the textual representation of skills (i.e. tags) and the body of answers provided by expert candidates. Indeed, these models fail to address vocabulary gap problem as they are based on exact and not semantic matching. Over the last few years, several models have been proposed to solve the vocabulary gap problem [14]. Precisely, statistical translation models [15], topic modeling approach [16], and more lately word embedding methods [3] which are among outstanding models to overcome this problem.

In this paper, we propose two models to translate a given skill area (e.g. "Android", "orm" and etc.) to a set of relevant words. These translations can be useful to improve the matching between expert finding queries and the technical textual evidence (i.e. answers) associated with each candidate. They can also be used independently by recruiters to detect important aspects of each skill area. For example, "java-ee" skill area can be translated to *application, web, spring, bean, service, http, session, request, controller* and *ejb* which are important aspects of "java-ee" in StackOverflow. Our first translation model (i.e. MI) is a statistical model based on mutual information and the second one (i.e. WE) is based on a word embedding method utilizing the specific structure and CQA's data to translate a skill area to its relevant words. After finding the appropriate translations, we have used four scoring approaches to combine the result of each translation to find the final ranking of experts for a given skill area.

The basic idea of our work has been recently published as a short paper in SIGIR conference [3]. However, the conference paper does not include a complete description of the proposed algorithms due to the page limit. This paper is a significant extension of the published short paper. In this paper, we have made four significant contributions including 1) We have added a new dataset (i.e. "PHP" dataset) to evaluate our proposed models. 2) Deeper analysis of the results are conducted. 3) Having adopted translation models which bridge the vocabulary gap issue, we aimed to take the quality of documents into account and have improved results considerably using Voteshare based scoring approaches. 4) Our entire sources including code and datasets have been uploaded and are publicly available for researchers[4].

2. StackOverflow

In this section, we briefly introduce the StackOverflow, its fundamentals and the major properties of its concepts including questions, answers, and user interactions.

As one of the most prominent CQAs, StackOverflow provides its users a quick access to expertise and knowledge. Users can ask questions, answer them, vote up or down and also edit posts. Some of these actions (e.g. voting

[4]http://tiny.cc/sofef

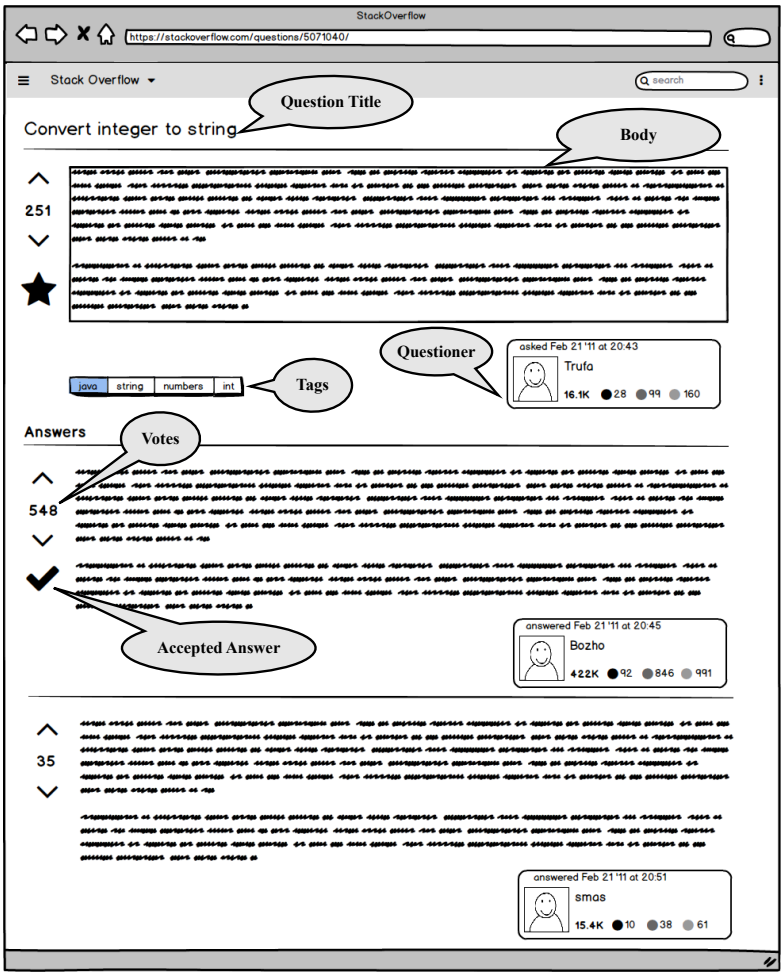


Figure 1: A sample question and its associated answers in StackOverflow. Title, body, tags, etc are highlighted.

up or down, commenting) are restricted to active members of the community. Users can gain or even lose reputation and badges with regard to their behavior and contribution to the community. For instance, the community of StackOverflow will reward users with 15 points, if their answers get accepted by the asker. Additionally, they can earn different levels of badges based on their high-quality contribution (e.g. an answer score of 100 or more will leads to "Great Answer" badge) which exemplifies gamification methods to motivate users.

Fig. 1 demonstrates a sample question and the related answers on StackOverflow. Each post (i.e questions and answers) includes body, vote count (i.e. subtraction of down-votes from up-votes), a doer (i.e. questioner and answerer), a set of tags which have been chosen by the questioner, and possibly a set of comments by users. Besides, the view count of the post, the posting date and time As illustrated in Fig. 1. When a questioner feels satisfied with an answer, he/she can accept it. *Acceptance* is determined by a green check-mark next to the accepted answer and is characterized as follows. *"Accepting an answer is not meant to be a definitive and final statement indicating that the question has now been answered perfectly. It simply means that the author received an answer that worked for him or her personally, but not every user comes back to accept an answer, and of those who do, they may not change the accepted answer if a newer, better answer comes along later."* [17]

Moreover, in order to improve the quality of questions asked in the community, StackOverflow adopts some qualification methods. In particular, low-quality questions will be put on hold state if they don't suit the community. When a question is put on hold, it can not be answered but can be edited to get suited for answering [18].

3. Observations on data

In this section, we discuss the necessity of translation models and show why they are important to retrieve more relevant results and accordingly improve the result of expertise retrieval task.

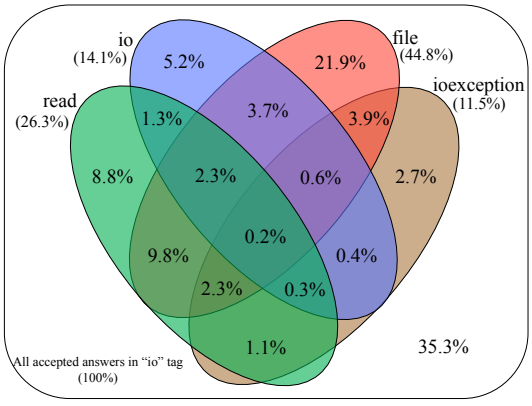


Figure 2: The Venn diagram of answers associated with questions tagged by "io"

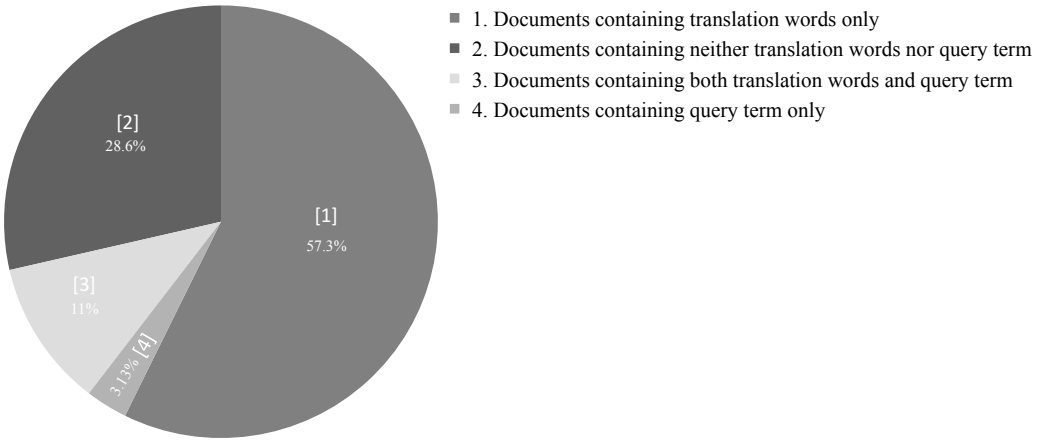


Figure 3: Share of "io" related documents retrieved by retrieval models

As mentioned in previous sections, a challenging problem of expert finding in StackOverflow is the vocabulary gap issue referring to the gap which exists between the main query and the terms used by experts.

Fig. 2 represents a Venn diagram visualizing the occurrence of top four translation words (by MI translation approach described in section 5.1) among the answers which are related to "io" (i.e. they have "io" as at least one of their tags). These translations are "file", "io", "read" and "ioexception". As indicated, only 14.1% of them include "io" in their bodies whereas, 64.7% of them include at least one of the translations. To be more specific, 21.9% of answers contain the term "file" and no other translations, 3.7% include both "file" and "io" term, 0.6% include "file", "io" and "ioexception" in their bodies and so forth. The answers including the term "file" are forming 44.8% of all answers associated with "io" tag. Whereas, only 5.2% of answers are covered by "io" term and no other translations. This exemplifies the deep gap between users' information need (i.e. queries) and the textual representation of queries which can be bridged using translation terms.

While Fig. 2 indicates the vocabulary gap for three translations of "io" (i.e. "file", "read" and "ioexception"), Fig. 3 indicates the same information for 10 translations. According to this figure, surprisingly, 57.3% of the "io" related answers have not included the term "io" itself in their body while covered at least one of the translation terms. This observation justifies the effectiveness of translation approach to improve the retrieval performance. Moreover, two sectors (i.e. sectors 2 and 3) of the pie chart indicate the cases

in which translation approaches cannot improve the retrieval performance directly. Specifically, sector 2 indicates 28.6% of the answers, related to "io" cannot be retrieved using neither "io" nor the related translations. In other words, they cannot be retrieved using our proposed translation models at all. The third sector demonstrates the set of answers which include both the main query (i.e. "io") and at least one of the translation terms. The answers in this sector can be directly retrieved without using translation models. Finally, the last sector illustrates the answers which include only the main query (i.e. "io") and none of the top ten translations. Practically, the translation models cannot be used for this subset of answers, however, they form a very small portion of the whole related answers (i.e. only 3.1%).

To sum up, both Figures 3 and 2 imply that the vocabulary gap in CQA networks, which exist between the user's query and the terms which are used in the answers body, is remarkable. Thus, translation models should be utilized in order to improve expert finding. Each of these translations retrieves new documents which should be blended together to find the final ranking of the candidates.

It is worth mentioning that documents (i.e. answers) have not the equal quality in StackOverflow [18]. Therefore, it is necessary to take the quality of answers into consideration in scoring step. A simple approach to measure the quality of the answers is *Voteshare*. To be more specific, for every question asked in StackOverflow, a competition is formed among answerer to get more votes. It is obvious that better answers of a specific question receive a higher share of votes compared to the other answers. Since better answers are expected to be provided by expert users, it can be inferred intuitively that these answers have a higher Voteshare. Voteshare, as it implies, is the share of an answer's votes to the summation of all answers' votes in a single thread as shown in Eq. 1.[5]

$$Voteshare(a_i) = \frac{Vote(a_i)}{\sum_{j=1}^{j=n} Vote(a_j)} \tag{1}$$

where n is the total number of answers in a thread. Fig. 4 indicates the relation between Voteshare and the quality of answers. In this figure, the an-

[5]We assumed that answers with zero/negative votes have not any Voteshare.

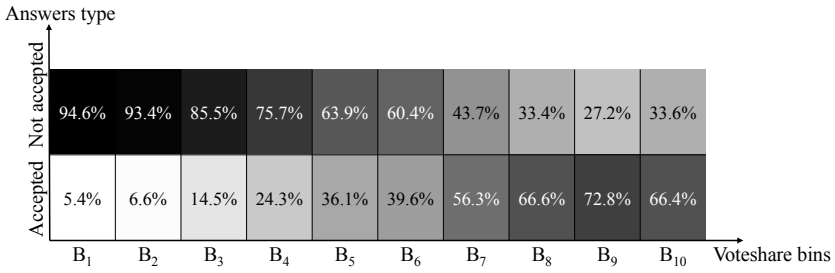


Figure 4: Distribution of Voteshare on high and low quality answers

swers are clustered into 10 bins with regard to their Voteshare values. To be more specific, B_1 includes the decile of answers with lowest Voteshare values, accordingly, B_{10} is decile of answers having the highest value of Voteshare. Additionally, we categorized answers into two groups including accepted answers (i.e. high-quality answers) and not accepted answers [6] (i.e. low-quality answers). According to Fig. 4, answers with slightly inferior value of Voteshare have lower chance of being accepted and accordingly they have less quality. In contrast, answers with superior value of Voteshare (e.g. Bin 8 and above) are most probably high-quality answers. For instance, 66.4% of answers in B_{10} are accepted, whereas, only 5.4% of the answers are marked as accepted in B_1.

In section 6 we will propose four methods in order to score the candidates using the documents which are retrieved via translation models. Two of these approaches is based on the Voteshare concept.

4. Baselines

In this section, we will explain our examined baselines in detail. First of all, we will review expert finding task mathematically. Then, we will explain probabilistic language models for expert finding. Finally, we will describe topic modeling approach which is among successful approaches to address the vocabulary gap issue.

In order to estimate $P(ca|q)$, different methods have been proposed [13, 16] to address this task. Prior to explaining the approaches which are taken as our baselines, it would be favorable to investigate the expert finding task and its nature. The task of expert finding is defined as identification and

[6]The concept of accepted answers is described in section 2.

ranking of candidates who are expected to be expert with respect to a given query. Thus, in expert finding task, we tend to indicate $P(ca|q)$ and then rank the candidates with regard to this probability. Obviously, the higher a candidate has $P(ca|q)$, the more probable he is to be an expert candidate. $P(ca|q)$ can be approximated using Bayes' theorem as follows.

$$P(ca|q) = \frac{P(q|ca)P(ca)}{P(q)}, \tag{2}$$

in which, $P(ca)$ is the prior probability of candidate ca, $P(q)$ is the probability of query q and can be ignored. As it has a constant value for a given query, it does not affect ranking of experts. Therefore, the probability of candidate given a query (i.e. $P(ca|q)$) is directly proportional to the probability of a query given the candidate $P(q|ca)$ and weighted by a prior probability of candidate.

$$P(ca|q) \propto P(q|ca)P(ca). \tag{3}$$

4.1. Language Models for Expert Finding

Balog *et al.* [19] have proposed two generative probabilistic language models, known as *candidate-based* and *document-based* approaches. Each one models the expert finding from slightly distinctive perspective. They are defined as follows.

4.1.1. Candidate-based approach

The first model, which is known as candidate-based model and referred as Model 1, approximates the corresponding probability (i.e. $P(q|ca)$) from candidates' point of view. Indeed, it builds a multinomial language model θ_{ca} for each candidate over the terms which are used in their associated documents. Under the assumption that query terms are independently sampled, $P(q|ca)$ can be calculated by the production of terms of the query as follows.

$$P(q|ca) = P(q|\theta_{ca}) = \prod_{t \in q} P(t|\theta_{ca})^{n(t,q)} \tag{4}$$

where $n(t, q)$ is the number of times which term t appears in query q. In order to estimate $P(t|\theta_{ca})$, firstly, the probability of a term given a candidate $P(t|ca)$ must be estimated. It should be noted that some candidates may not use a specific term of a query and thus make the probability equal to

zero. Hence, in order to avoid zero probabilities due to data sparsity, $P(t|ca)$ should be smoothed with the background collection probabilities as shown in Eq. 5.

$$P(t|\theta_{ca}) = (1 - \lambda_{ca})P(t|ca) + \lambda_{ca}P(t) \tag{5}$$

where $P(t)$ is the probability of a term in the documents collection, $P(t|ca)$ is the likelihood that candidate ca would write about term t and is estimated using Eq. 6, and λ_e is the parameter of model.

$$P(t|ca) = \sum_{d \in D_{ca}} P(t|d, ca).P(d|ca) \tag{6}$$

Assuming that document and the candidate are conditionally independent, $P(t|d, ca)$ can be reduced to $P(t|d)$ in which is the occurrence probability of term t in document d. $P(t|d)$ can also be approximated using P_{MLE} (i.e. maximum-likelihood probability). Candidate model is obtained by combining the Eqs. 4-6 as shown in the following equation.

$$P(q|ca) = \prod_{t \in q} \Big\{ (1 - \lambda_{ca}).\Big(\sum_{d \in D_{ca}} P(t|d).P(d|ca) \Big) + \lambda_{ca}.P(t) \Big\}^{n(t,q)} \tag{7}$$

4.1.2. Document-based approach

The second approach of expert finding adopted by Balog et al. acts somewhat differently to estimate $P(q|ca)$. It is known as document-centric model (referred as Model 2) and in spite of candidate-based model which found candidates directly, it considers the documents in a collection as a bridge which links the given query to candidates and evidence their author's expertise. In this case, the problem of expert finding can be defined as follows. Given a collection of documents which are ranked according to the given query, the authors of the relevant documents to the query should be retrieved and ranked. This model can be set off by taking the sum over entire documents $d \in D_{ca}$ as expressed in Eq. 8.

$$P(q|ca) = \sum_{d \in D_{ca}} P(q|d, ca).P(d|ca) \tag{8}$$

where $P(q|d, ca)$ is the likelihood of generating query q according to document d and candidate ca, and $P(d|ca)$ denotes the binary association of document

d and candidate ca. Under the assumption that query terms are occurred independently, $P(q|d, ca)$ can be estimated as follows.

$$P(q|d, ca) = \prod_{t \in q} P(t|d, ca)^{n(t,q)} \tag{9}$$

Having substituted Eq. 9 into Eq. 8, we result the following equation.

$$P(q|ca) = \sum_{d \in D_{ca}} \prod_{t \in q} P(t|d, ca)^{n(t,q)}.P(d|ca) \tag{10}$$

In order to estimate $P(t|d, ca)$, we can assume conditional independence between the query q and the candidate ca i.e. $P(t|d, ca) \approx P(t|\theta_d)$ in which θ_d is the document language model which is inferred from document d. Therefore, the probability of a term t given document model θ_d can be calculated using Eq. 11.

$$P(t|\theta_d) = (1 - \lambda_d).P(t|d) + \lambda_d.P(t) \tag{11}$$

By putting $P(t|\theta_d)$ instead of $P(t|d, ca)$ in Eq. 10, final approximation of document-based model is yielded as follows.

$$P(q|ca) = \sum_{d \in D_{ca}} \prod_{t \in q} \Big\{(1 - \lambda_d).P(t|d) + \lambda_d.P(t)\Big\}^{n(t,q)}.P(d|ca) \tag{12}$$

4.2. Topic Modeling for Expert Finding

Topic modeling is the other baseline to determine the probability of a candidate to be an expert with regard to a given query (i.e. $P(ca|q)$). Momtazi and Naumann [16] have proposed the model for the task of expert finding. The model approaches the document-based approach with this difference that it utilizes topics extracted from a document repository (i.e. collection) rather than documents. Indeed, the extracted topics are acted as a bridge to connect candidates to a given query. It is worth mentioning that topic modeling approach is among vigorous approaches to overcome the vocabulary gap issue and also is expected to outperform state-of-the-art approaches (i.e. candidate-based and document-based). The process of expert finding using topic modeling includes two main phases. The first phase is involved with

extracting topics using Latent Dirichlet Allocation (LDA) and mostly performed off-line. In the next phase, the extracted topics are used to determine $P(q|ca)$ as follows.

$$P(q|ca) = \sum_{z \in Z} P(q|z, ca) P(z|ca) \tag{13}$$

in which Z symbolizes the extracted topics, and $P(q|z, ca)$ is the likelihood of generating query q given topic z and candidate ca. Under the conditional independence assumption between q and ca, $P(q|z, ca)$ can be reduced to $P(q|z)$. The probability of query given topic (i.e. $P(q|z)$) is calculated by taking the product over the terms of query q as expressed in Eq. 14.

$$P(q|z) = \prod_{t \in q} P(t|z)^{n(t,q)} \tag{14}$$

where t denotes query term, and $n(t, q)$ is the number of times that t appears in q.

We can also estimate the probability of topic z given candidate ca (i.e. $P(z|ca)$) using Bayes' theorem as follows.

$$P(z|ca) \propto P(ca|z) P(z) \tag{15}$$

in which $P(z)$ is the prior probability of selecting topic z and generally it is considered to be uniform. Substituting Eqs. 15 and 14 into Eq. 13 leads to the following equation.

$$P(q|ca) = \sum_{z \in Z} \Big[\prod_{t \in q} P(t|z)^{n(t,q)} \Big] P(ca|z) P(z) \tag{16}$$

where $P(ca|z)$ is the probability that topic z would be talked by candidate ca and calculated using LDA algorithm. To avoid zero probabilities, Jelinek-Mercer smoothing is employed. So in this way, a background probability is interpolated with the original probability to ensure that there are not any zero probability (the probability is always non-zero). By applying Jelinek-Mercer smoothing to Eq. 16, the final estimation of topic modeling approach is resulted as follows.

$$P(q|ca) = \\ \sum_{z \in Z} \prod_{t \in q} \Big\{ (1 - \lambda) P(t|z) + \lambda P(t) \Big\}^{n(t,q)} P(ca|z) P(z) \tag{17}$$

5. Translation Approaches

As mentioned in the previous section, the translation approach can be beneficial to reduce the gap between query and the terms occurred in documents. Here, each query represents a *skill area* that can be used to retrieve relevant candidates. In the rest of this paper, we demonstrate the expert finding query by *sa* notation[7]. In this section, we explain two methods of skill area translation which are Mutual Information approach (i.e. MI) and Word Embedding approach (i.e. WE), respectively.

5.1. Mutual Information Based Approach (MI)

Assuming each skill area as a class label, the set of answers in StackOverflow can be partitioned into two disjoint subsets. The first subset contains answers tagged by a given skill area, and the second one includes other answers. In our problem, we can use the mutual information (MI) to determine how much information the presence or absence of a term contributes to making the correct classification decision [20]. For each pair of word w and skill area sa, the MI can be calculated using the following equation.

$$MI(sa, w) = \sum_{A_{sa}=0,1} \sum_{A_w=0,1} p(A_{sa}, A_w) \log \frac{p(A_{sa}, A_w)}{p(A_{sa})p(A_w)} \tag{18}$$

in which A_{sa} and A_w denote binary variables indicating the occurrence event of skill area sa and word w in an answer. The probabilities indicated in Eq. 18 can be estimated using the following equations:

$$\begin{aligned}
p(A_{sa} = 1) &= \frac{c(A_{sa} = 1)}{N} \\
p(A_{sa} = 0) &= 1 - p(A_{sa} = 1) \\
p(A_w = 1) &= \frac{c(A_w = 1)}{N} \\
p(A_w = 0) &= 1 - p(A_w = 1) \\
p(A_{sa} = 1, A_w = 1) &= \frac{c(A_{sa} = 1, A_w = 1)}{N} \\
p(A_{sa} = 1, A_w = 0) &= \frac{c(A_{sa} = 1) - c(A_{sa} = 1, A_w = 1)}{N}
\end{aligned}$$

[7]The *tag*, *sa* and *q* refer to the same concept in this paper.

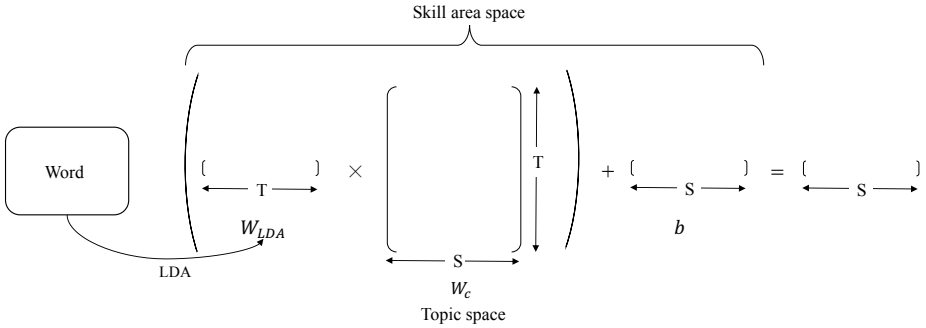


Figure 5: Schematic representation of the proposed word embedding model

$$p(A_{sa}=0,A_w=1)=\frac{c(A_w=1)-c(A_{sa}=1,A_w=1)}{N}$$
$$p(A_{sa}=0,A_w=0)=1-p(A_{sa}=1,A_w=1)$$
$$-p(A_{sa}=1,A_w=0)-p(A_{sa}=0,A_w=1)$$

where $c(A_{sa}=1)$ and $c(A_w=1)$ imply number of the answers associated with skill area sa, and denotes number of answers containing word w, respectively. N is also the number of all answers. To obtain translation probability, the MI score should be normalized using Eq. 19. For a given skill area sa, the most enlightening words can be sorted using $p_{MI}(w|sa)$ probability as follows.

$$p_{MI}(w|sa)=\frac{MI(sa,w)}{\sum_{w'}MI(sa,w')} \tag{19}$$

$p_{MI}(w|sa)$ gives the probability of translating skill area sa to word w. Intuitively, this probability will be higher, if the word w and skill area sa are likely to co-occur with each other.

5.2. Word Embedding Based Approach

As mentioned in section 4.2, the proposed model of Momtazi *et al.* [16] maps expert candidate, documents, and their used terms to a *topic space* and the matching between them is formulated in the corresponding space. Having reduced the vocabulary gap, the topic modeling approach improves the retrieval performance over the profile-based and document-based models introduced in section 4.1. Though, since terms representing skill areas (e.g. "hibernate", "orm" and etc.) rarely occurs in documents, it is necessary to embed document terms and skill areas i.e. queries into a single new space which is called *skill area space*.

Having embedded skill areas and document terms into the same space, in this section, we proposed a domain-aware translation method which maps a given skill area to the most relevant words occurred in documents (i.e. answers of StackOverflow in our problem).

As indicated in Fig. 5, we start translation process by applying topic modeling approach to the given set of documents to obtain a low-dimensional i.e. *topic space* representation of each word in our dataset. Then, a mapping function is designed from the *topics space* to the *skill areas space*. Using this function, the words of documents and the tags, which represent skill areas, are embedded into a single low-dimensional space. For notational convenience, we write $P(\boldsymbol{sa}|.)$ for the probability distribution over skill areas, which is the result of vector arithmetic. The relevance probability of the skill areas given a word w can be expressed by the following equation.

$$P_{WE}(\boldsymbol{sa}|w) = \frac{1}{z} e^{W_{LDA}.W_C+b} \tag{20}$$

where W_{LDA} is a $1 \times T$ vector expressing word w in topic space (T is the number of topics), W_C is a $T \times S$ matrix which maps the topic space representation of word w to skill area space (S symbolizes the number of skill areas), b is a $1 \times S$ vector which represents the prior relevance probability of skill areas to a given word, and finally z is a normalization factor which is calculated as follows: $z = \sum_{j=1}^{|sa|} [e^{W_{LDA}.W_C+b}]_j$.

In this model, the matrix W_C and vector b are denoting unknown parameters and should be learned during training. They are estimated using error back-propagation algorithm. During training step, for a set of given documents (i.e. training data) with known tags i.e. skill areas, we estimate the observed occurrence probability of each word in the skill areas as follows.[8]

$$P_{observed}(\boldsymbol{sa}|w) = \frac{tf(\boldsymbol{sa}, w)}{tf(w)} \tag{21}$$

in which $tf(sa, w)$ is the term frequency of w in the documents which are tagged by sa, and $tf(w)$ denotes the term frequency of w in training set. During model construction, we optimize the cross entropy of $H(P_{WE}, P_{observed})$

[8]In real scenarios, most of the times, it is not possible to compute $P_{observed}$, accordingly, we estimated it observing only a subset of data (i.e. training data).

using batch gradient descent as shown in Eq. 22.

$$L(W_C, b) = \frac{1}{m}\sum_{i=1}^{m} H(P_{WE}, P_{observed}) + \frac{\lambda}{2m}\left(\sum_{i,j} W_{C_{i,j}}^2\right) \tag{22}$$

$$= -\frac{1}{m}\sum_{i=1}^{m}\left(\sum_{j=1}^{|sa|} P_{observed}(sa_j|w_i)\log P_{WE}(sa_j|w_i)\right) + \frac{\lambda}{2m}\left(\sum_{i,j} W_{C_{i,j}}^2\right)$$

where $L(W_C, b)$ gives us the loss function, m is the size of a training batch, and λ is a weight regularization parameter. The update rule for a particular parameter $\theta(W_C, b)$ given a single batch of size m is:

$$\theta^{(t+a)} = \theta^{(t)} - \alpha^{(t)} \odot \frac{\partial L(W_C^{(t)}, b^{(t)})}{\partial \theta} \tag{23}$$

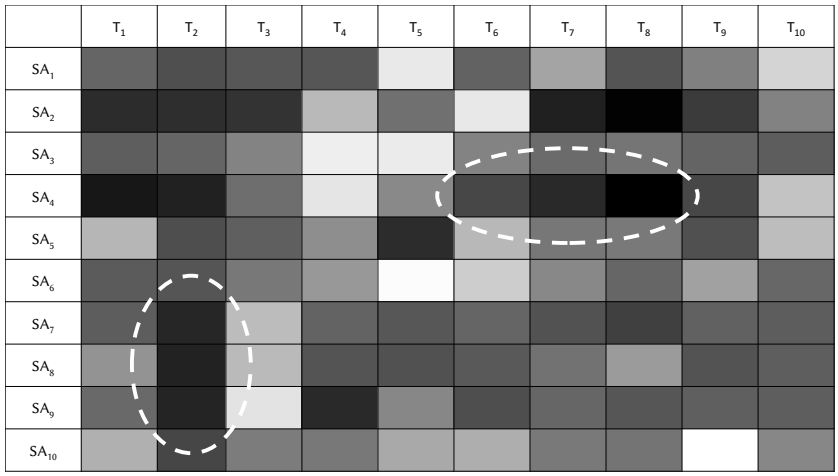


Figure 6: The heat-map of a subset of trained matrix W_c

Suppose that word w is related to topic t_i and t_j and numerously appeared in the answers associated with the skill areas sa_k and sa_m. During training, the weights of the matrix W_C and vector b will be updated such that the representation of word w in skill areas space be placed near the representation of sa_k and sa_m. In addition, by applying the update rule, other words which are related to t_i and t_j will also get closer to s_k and s_m in skill areas space.

As pointed out before, the matrix W_C gives a mapping function from topics space to skill areas space. Fig. 6 depicts the heat-map of a subset of the matrix after training. The darker a cell is, the stronger association would be between the specific topic and skill area and vice versa. For example, skill area SA_4 is more associated with topics T_7, T_8 and T_9. While, T_2 is more associated with skill areas SA_7, SA_8 and SA_9. This figure illustrates a many-to-many relationship between topics and skill areas.

After the training process of matrix W_C and vector b, $p(sa|w)$ can be estimated for each pair of words w and skill area sa using Eq. 20. In order to find the most relevant translation terms for a given skill area sa, we use Bayes' theorem to estimate $p(w|sa) \approx p(w)p(sa|w)$, in which $p(w)$ denotes the prior probability of word w to be chosen as a robust translation. We estimate $p(w)$ using TF-IDF computed over the data collection in our experiments.

6. Scoring Approach

In this section, we propose our scoring approaches to estimate the final ranking of the candidates according to the translation terms extracted by the MI and WE methods. Suppose a given skill area sa is translated to $w_1, w_2, ..., w_n$, which includes the skill area word as well (i.e. self translation). In expert finding problem, the goal is to estimate the probability of $P(ca|sa)$, by assuming the prior probability of candidates (i.e. $P(ca)$) to be uniform and having in mind that $P(sa)$ can be ignored in ranking, we have $P(ca|sa) \approx P(sa|ca)$, which is estimated as shown in Eq. 24.

$$P(sa|ca) \overset{translate}{=\!=} P(w_1, ..., w_n|ca) \tag{24}$$

Each document d associated with candidate ca and contains one or more translation words can be considered as an evidence of the author's expertise on skill area sa. Therefore, following the idea of Model 2 proposed in [19], the corresponding probability can be estimated as follows.

$$P(w_1, ..., w_n|ca) \propto \sum_{d \in D_{ca}} P(w_1, ..., w_n|d, ca).P(d|ca) \tag{25}$$

in which D_{ca} indicates the set of documents associated with candidate ca. By applying Bayes' theorem, we have $P(d|ca) = \frac{P(ca|d).P(d)}{P(ca)}$. Since, in StackOverflow, each document (i.e. answer) has exactly one author. Therefore, we can assume that $P(ca|d) = 1$. Besides, $P(ca)$ is also assumed to be uniform.

In addition, assuming conditional independence between the words and the candidate, the probability of $P(w_1, ..., w_n|ca)$ can be rewritten as follows.

$$P(w_1, ..., w_n|ca) \propto \sum_{d \in D_{ca}} P(w_1, ..., w_n|d).P(d) \tag{26}$$

Where $P(w_1, ..., w_n|d)$ is *document relevancy score* and $P(d)$ is *document quality score* (prior probability of the document) of the documents. In this research, we have proposed two approaches for estimating each score which is described in the rest of this section.

6.1. Estimating document relevancy score

We have proposed two approaches to estimate $P(w_1, ..., w_n|d)$. In the first approach, we exploit the idea of language model to score each candidate. Following the Eq. 12 in section 4.1.2, the mentioned probability can be estimated as follows:

$$P(w_1, ..., w_n|d) \propto \prod_{w_i} \Big\{ (1 - \lambda_d).P(w_i|d) + \lambda_d.P(w_i) \Big\} \tag{27}$$

Where, $P(w_i|d)$ is calculated by maximum likelihood estimation and $P(w_i)$ indicates the collection probability of word w_i. We refer this approach as *Language Model Scoring* in the rest of this paper.

In the second approach to estimate $P(w_1, ..., w_n|d)$, instead of applying a probabilistic model (i.e. language model), we have focused on the number of expertise evidence occurred in each candidate's profile. Accordingly, we consider each document d associated with candidate ca which contains one or more translation words as an evidence of the author's expertise on skill area sa. This is formally demonstrated in Eq. 28. We refer this approach as *Binary Scoring* in the rest of this paper.

$$P(w_1, ..., w_n|d) = \begin{cases} 0, & \text{if } w_1, ..., w_n \notin d \\ 1, & \text{otherwise} \end{cases} \tag{28}$$

6.2. Estimating document quality score

In this section, we propose two estimation methods for document quality score (i.e. $P(d)$). In the first approach, we simply assume all documents have

the same quality and accordingly, $P(d)$ can be ignored in ranking process. As a result Eq. 26 can be rewritten as follows:

$$P(w_1, ..., w_n|ca) \propto \sum_{d \in D_{ca}} P(w_1, ..., w_n|d) \tag{29}$$

In the second method, in order to take the quality of documents into account, utilizing the concept of Voteshare introduced in section 3, it is possible to estimate $P(d)$ by the Voteshare of document d. Accordingly, Eq. 26 can be rewritten as shown in the Eq. 30. We refer this scoring approach as Voteshare based scoring in the rest parts of this paper.

$$P(w_1, ..., w_n|ca) \propto \sum_{d \in D_{ca}} P(w_1, ..., w_n|d).Voteshare(d) \tag{30}$$

In both aforementioned equations, $P(w_1, ..., w_n|d)$ can be estimated by either language model or binary scoring methods which will lead to four disparate approaches to score the candidates.

7. Experiments

In this section, a set of experiments are designed to address the following research questions:

- **RQ1:** Which models are more successful to overcome the vocabulary gap problem?
- **RQ2:** How the proposed scoring approaches can affect the overall performance of retrieval?
- **RQ3:** What is the effect of Voteshare in scoring step?
- **RQ4:** How many translations are enough to cover the vocabulary gap? How sensitive are the proposed approaches on the number of translations?
- **RQ5:** Is there any difference between translation provided by Mutual Information and Word Embedding approaches?

In the rest of this section, we first set forth the experimental setup and parameter setting and then present our experimental results to answer the aforementioned research questions.

7.1. Experimental Setup

In this section, we describe our datasets and parameter setting.

7.1.1. Data Collection

Our dataset is downloaded from StackOverflow[9]. It covers the period August 2008 until March 2015 and contains 24,120,523 posts.

Table 1: General statistics for "Java" and "PHP" data collections

DataSet	#Q	#A	#C	Avg Q-Rel
Java	810,071	1,510,812	206,397	44.75
PHP	714,476	1,298,107	191,060	91.64

We have selected questions and their associated answers tagged by "Java" and "PHP" as two separated data collection in our experiments. The statistic related to each data collection including number of questions (#Q), number of answers (#A), number of candidates (#C) and the average number of relevant candidates per query (Avg Q-Rel) is indicated in table 1.

We mark users as experts on each tag (i.e. skill area) when two conditions are met. First, similar to the definition proposed in [2], the candidates should have ten or more[10] of their answers marked as accepted by the questioner. Second, following the idea proposed in [21], the acceptance ratio of their answers should be higher than the average acceptance ratio (i.e. 40%) in test collection. The first condition filters users with a low level of engagement and the second condition filters low-quality users. Moreover, we select 100 top most frequent tags which co-occurred with "Java" and "PHP" tags in our datasets as expert finding queries. Our queries and implementations for all approaches including baselines is publicly available[11].

7.1.2. Parameter Setting and Implementation Detail

In our baseline models described in section 4, we use $\lambda = 0.5$ as the smoothing parameter for both models in section 4.1. For topic modeling approach described in section 4.2, we tried different settings and finally set the number of topics to be 100.

[9]https://archive.org/details/stackexchange

[10]Since the number of question and answers in "PHP" dataset is less than "Java", we used 8 accepted answers as the threshold.

[11]http://tiny.cc/sofef

We translate each skill area to top 10 most relevant words using the MI and WE methods. In WE method, we restrict the size of vocabulary to top 2^{16} most frequent words. In order to optimize the loss function (i.e. Eq. 22), we use adadelta ($\rho = 0.95$, $\epsilon = 10^{-6}$) [22] with batch gradient descent and weight decay $\lambda = 0.01$.

Additionally, Tensorflow[12] has been used to calculate matrix operations on a Nvidia Titan X GPU. In order to index and search data and also implement the baselines and the proposed models, we have used Apache Lucene[13].

7.2. Experimental Results

In this section, we aim to answer the research questions mentioned earlier. Our first experiment is carried out to finding an answer to **RQ1**: Which models are successful to overcome the vocabulary gap problem? Our next experiment is concerned with analyzing proposed scoring methods to answer **RQ2:** How the proposed scoring approaches can affect the overall performance of retrieval? A comparison between Voteshare and basic scoring approach is done in order to answer the next research question **RQ3**: What is the effect of Voteshare in scoring step? The next experiment is performed in order to determine the effect of increasing translation counts on the results which sets forth the **RQ4**: How many translations are enough to cover the vocabulary gap? How sensitive are the proposed approaches on the number of translations? Finally, our last experiment is a comparison between the translations provided by our two proposed models and accounts for the **RQ5**: Is there any difference between translation provided by Mutual Information and Word Embedding approaches?

7.2.1. Analyzing the performance of models to overcome vocabulary gap problem

Table 2 indicates the result of Language Model (LM) 1 and 2, Topic Modeling (TM), Mutual Information (MI) and Word Embedding (WE) scored by the basic binary approach. According to this table, two main observations can be concluded. First, the performance of TM is significantly better than LM approaches. This observation indicates that TM approach can overcome the vocabulary gap problem to some extents. Second, the MI and WE approaches even with basic scoring approach outperform TM. As mentioned

[12]https://www.tensorflow.org
[13]https://lucene.apache.org/

Table 2: Comparison of the proposed models with baselines. Binary scoring results are reported.

	Method	MAP	P@1	P@5	P@10
Java	LM 1	0.377	0.560	0.500	0.440
	LM 2	0.362	0.540	0.482	0.425
	TM	0.434	0.550	0.530	0.488
	MI	0.478	**0.660**	0.604	0.529
	Δ LM1	26.8% *	17.9% *	20.8% *	20.2% *
	Δ TM	10.1% *	20.0% *	14.0% *	8.4%
	WE	**0.496**	0.650	**0.626**	**0.540**
	Δ LM1	31.6% *	16.1% *	25.2% *	22.7% *
	Δ TM	14.3% *	18.2% *	18.1% *	10.7% *
PHP	LM 1	0.335	0.570	0.524	0.479
	LM 2	0.309	0.520	0.478	0.437
	TM	0.401	0.530	0.550	0.491
	MI	0.458	0.590	0.612	0.561
	Δ LM1	36.7% *	3.5%	16.8% *	17.1% *
	Δ TM	14.3% *	11.3% *	11.3% *	14.3% *
	WE	**0.509**	**0.600**	**0.626**	**0.581**
	Δ LM1	52.0% *	5.3%	19.5% *	21.3% *
	Δ TM	27.0% *	13.2% *	13.8% *	18.3% *

** indicates that improvement is statistically significant on a two-tailed paired t-test ($\alpha = 0.05$)*

before, a skill area can be mapped to more than one topic and each conversely a topic can be mapped to more than one skill area in our problem. However, in the TM approach, each answer is mapped to a few number of topics but the relationship between skill areas and answers is not determined directly. In contrast, the MI and WE methods directly extract top relevant translations for a given skill area and accordingly they surpass in terms of both precision and recall. As a result, these methods improve precision@n (P@n) and mean average precision (MAP) measures significantly in comparison with TM method. This observation is consistent on both "Java" and "PHP" datasets. Another interesting observation is that the TM method decreases P@1 measure on both data collections which means although this method is successful in reducing the vocabulary gap, and accordingly improving the recall, it slightly decreases the precision measure in top level rankings. In contrast, our proposed models not only increase the recall but also marginally improve the precision on both data collections.

Table 3: Comparison of scoring methods with baselines based on the MAP measure

Method	Java	PHP
Language Model 1	0.377	0.335
Language Model 2	0.362	0.309
Topic Modeling	0.434	0.401
Basic Language Model	0.371	0.341
Voteshare based Language Model	0.419	0.366
Basic Binary Scoring	0.496	0.509
Voteshare Based Binary Scoring	0.660	0.562

7.2.2. Analyzing performance of scoring methods

The retrieval performance of baselines and scoring methods on WE translations are demonstrated in Table 3. As indicated in this table, performance of Language Model based Scoring (LMS) are significantly less than binary scoring approaches. In some cases, LMS methods' performance is even below the baseline methods. This observation can be illustrated by two explanations. First, In binary scoring methods each document containing at least one translation method is assumed as a single vote of expertise evidence. Whereas, In LMS approaches, abundance occurrence of a translation term in a document will lead to large score for its owner and significantly affect the amount of evidence for that document. However, publishing a single post including a large number of relevant terms cannot essentially indicates expertise of a candidate, accordingly, it should not over-affect the score of each candidate. Take, "io" skill area as an example, it could be translated into "stream", "file", etc. Since programming codes form up a large portion of posts in StackOverflow and "stream" is commonly used in programming codes, it is occurred in a large number of documents. Consequently, superior Term Frequency (TF) of this word would unfairly increase the score of these documents as expertise evidence. In other words, one reason behind the poor performance of LMS methods is considerable effect of TF on candidates scoring. Second, although expanding query (i.e. skill area) in LMS methods would marginally increase the recall of performance, the precision is significantly decreased as a result of concept drift [14] which will lead to inferior retrieval performance in comparison with other scoring approaches. Consequently, we will focus on binary scoring method in the next experiments.

7.2.3. Voteshare vs. Basic scoring approach

As explained before, the MI and WE with basic scoring approach (i.e. without including Voteshare) improve the recall measure considerably (i.e. they can solve the vocabulary gap problem) in comparison with TM and LM models. In addition, they can slightly improve the precision of retrieval. As mentioned in section 3, all documents (i.e. answers) in StackOverflow do not have the same quality. The Voteshare scoring approach aims to solve this problem by exploiting high-quality answers in scoring step. Table 4 indicates the comparison of MI and WE translation models on basic and Voteshare based binary scoring approaches. In this table, MI (BS) and

Table 4: Comparison of scoring approaches for both MI and WE models. BS and VS indicate basic and Voteshare based binary scoring approaches respectively.

	Method	**MAP**	**P@1**	**P@5**	**P@10**
Java	MI (BS)	0.478	0.660	0.604	0.529
	WE (BS)	0.496	0.650	0.626	0.540
	MI (VS)	0.647	0.850	**0.736**	0.652
	Δ MI (BS)	35.3% *	28.8% *	21.9% *	23.3% *
	WE (VS)	**0.660**	**0.860**	0.728	**0.661**
	Δ WE (BS)	33.1% *	32.3% *	16.3% *	22.4% *
PHP	MI (BS)	0.458	0.590	0.612	0.561
	WE (BS)	0.509	0.600	0.626	0.581
	MI (VS)	**0.587**	**0.750**	**0.726**	**0.642**
	Δ MI (BS)	28.3% *	27.1% *	18.6% *	14.4% *
	WE (VS)	0.562	**0.750**	0.696	0.621
	Δ WE (BS)	10.4% *	25.0% *	11.2% *	6.9%

** indicates that improvement is statistically significant based on a two-tailed paired t-test ($\alpha = 0.05$)*

WE (BS) indicate corresponding translation models with Basic binary scoring (BS) approach. MI (VS), and WE (VS) indicate translation models with Voteshare based binary scoring approach (VS). Three important observations are noticed here: First, Voteshare scoring approach remarkably improve the precision at top levels of ranking independent of the translation model and the dataset. Second, in the majority of measures, the performance of WE method with basic scoring approach outperforms the MI model with the same scoring. The only exception here is P@1 on Java data collection which

MI performs better than WE but not noticeably. Third, the performance of WE and MI method with Voteshare scoring approach is almost the same on both test collections. This observation indicates that although WE and MI approaches overcome the vocabulary gap problem with different mechanisms, the Voteshare scoring method - Utilizing only high-quality documents - provides a consistent performance on both test collections independent of underlying translation models.

7.2.4. Sensitivity analysis on number of translations

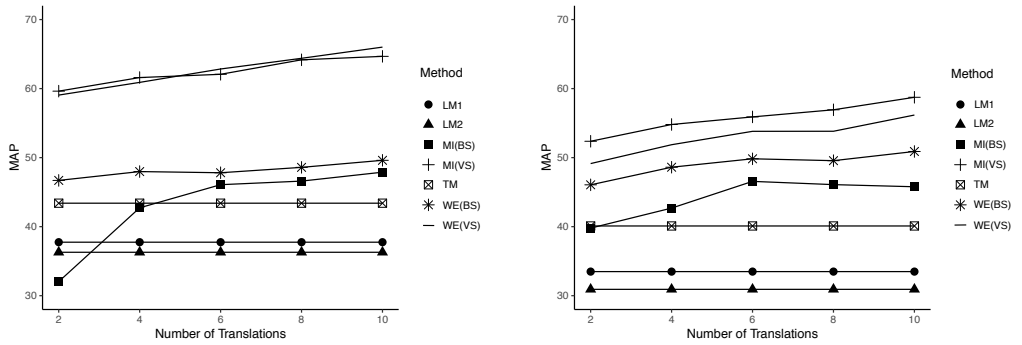
In this section, the proposed models are compared with the baselines in terms of number of translations for each original query which is the main parameter of the WE and MI models. Fig. 7a and Fig. 7b indicate the sensitivity of the proposed methods on the number of translation for "Java" and "PHP" data collections, respectively. Three important observations are marked here: First, the performance of WE and MI methods are almost ascending on both datasets independent of scoring approach. However, the performance of the proposed models does not change significantly after six translations. This observation is important because it means that the proposed models can be practically used in a retrieval engine without a significant overhead. Second, apparently, the MI method (with basic scoring) is more sensitive to the number of translation. In contrast, the WE method has a consistent performance even with only two translations on both test collections. Third, the translation models with Voteshare based scoring have almost the same performance (especially in "Java" test collection). This observation means that the Voteshare approach can work very well on top of both translation models.

7.2.5. Comparison of word embedding and mutual information translations

Table 5: Sample skill area translations using word embedding and mutual information methods

Method	Java				PHP			
	Skill Area	Translation 1	Translation 2	Translation 3	*Skill Area*	Translation 1	Translation 2	Translation 3
MI	*hibernate*	hibernate	entity	table	*SOAP*	xsd:string	s:element	soap:body
	swing	textsample	jframe	jpanel	*GD*	gd	image	imagecreatetruecolor
	selenium	__method.apply	selenium	webdriver	*JSON*	json	json_encode	json_decode
WE	*hibernate*	hibernate	entity	employee	*SOAP*	soap	wsdl	soapclient
	swing	jpanel	jbutton	jlabel	*GD*	image	img	alt
	selenium	tests	junit	test	*JSON*	json	data	json_encode

Table 5 indicates the top three translations for a few number of skill areas extracted by MI and WE models on both datasets. Interestingly, the MI



(a) Java dataset

(b) PHP dataset

Figure 7: The effect of varying number of translations on MAP measure for all proposed models

model usually translates the given skill area to more specific words while WE model selects more general words for the same topic. It seems that the MI model, which is basically a statistical translation model, is more sensitive to the co-occurrence of words and skill areas in documents. As a result, the MI model in most cases selects pieces of program codes (e.g. "__method.apply" for "selenium") which are the most frequent words found in StackOverflow answers. On the other hand, the WE model, as a semantic-aware translation model, provides more meaningful and human-friendly translations which can be used in ad-hoc tasks apart from expert finding. For instance, recruiters can use these translations to select outstanding questions about a skill area.

8. Related Work

In this section, firstly, we review expert finding task and its methods in different domains. Then, we discuss prior works on CQA platforms. Our proposed models are inspired by semantic matching and translation models whose related studies also have been investigated in the last part of this section.

8.1. Expert Finding

In the past few years, expert finding has been attracted a lot of attention in the Information Retrieval community. As mentioned earlier, the task of expert finding is to retrieve and rank the experts given a field of expertise as an input query.

This problem has been inquired in many environments such as organizations [19], bibliographic networks [23, 24, 25], social networks [26, 23], Wikipedia [27], LinkedIn [28], CQAs [29, 30] and even Instagram [31].

There have been many studies on generative probabilistic models for this task which rank candidates according to $P(ca|q)$ indicating the probability of a candidate ca being an expert in the topic q (i.e. query) [13]. These models are categorized into two groups including candidate generation models [32, 33] and topic generation models [19]. Most of these methods mainly use raw textual pieces of evidence, ignoring domain-specific information (e.g. document quality or structure). Nevertheless, there are various methods proposed to extend and enhance expertise retrieval in many ways. Deng *et al.* [6] proposed a query-sensitive method to model the authors' authorities based on the community citation networks and developed an adaptive ranking method to enhance expertise retrieval. Furthermore, Zhao *et al.* [34] proposed a ranking metric network learning framework for expert finding using both users' relative quality rank to given questions and their social relations. These methods are somewhat failed to address the issue of vocabulary gap in expertise retrieval. However, there have been some studies to bridge this gap. Momtazi and Naumann [16] proposed a topic modeling approach to extract the main topic of documents, then the extracted topics are acted as a bridge to find the probability of nominating each candidate as the expert for a given query. Additionally, Van Gysel *et al.* [35] introduced an unsupervised discriminative model for this task by exclusively employing textual evidence via learning distributed word representations in an unsupervised way.

8.2. Community Question Answering

Over the recent years, many studies have been done on detecting expert users in CQAs [36, 37]. In these approaches, associated documents, social interactions, and the personal activities of each candidate are deemed as their expertise evidence. Nonetheless, CQA platforms are dynamic environments due to their immense daily posts, the rate of joining new users, changing in their activities and interests, emerging new topics and upward or downward trend (i.e. novelty) of topics. Consequently, in these networks, experts should be detected not only by their textual pieces of evidence (i.e. documents), but also by using network structure and specific features of CQA [38, 2, 39].

Another aspect of research on CQAs is the automatic evaluation of the quality of user generated contents based on a defined measure (i.e. formula) [40, 41].

Neshati *et al.* [29] also have introduced a new problem of detecting users that can be potential experts in future. The proposed method relies on the expertise evidence of users (i.e. documents) in current time and then according to these pieces of evidence, the authors have suggested a method to predict the best ranking of experts in future.

8.3. Semantic Matching

In recent years, a great deal of studies have been conducted in semantic matching as there are many approaches proposed in this task such as Query Reformulation [42], Translation Models [15], Topic Modeling [43] and etc. It should be noted that only Translation Models and Topic Modeling approaches are in the scope of this research study.

Statistical machine translation (SMT) refers to statistical learning methods for translating texts from one language to another or the same language[14]. To clarify, suppose "CA" as the main query. It is known that it can match "California" with a high degree of precision. In our problem, queries can be regarded as a single word (i.e. skill area), and documents are texts built from other words which have been used in that skill area. SMT technologies aim to deal with the mismatch between query and document in expert finding. As an instance, skill area "java-ee" (i.e. query) can be translated to *application, web, spring, bean, service, http, session, request, controller and ejb* which are crucial aspects of "java-ee" in StackOverflow.

The primary idea of SMT methods is to estimate the probability of translating a document to a query. As a term can be translated to a set of other terms with a certain probability, SMT methods can address the vocabulary gap problem. Karimzadehgan and Zhai [15] have adopted a method to estimate statistical translation models (SMT) based on mutual information in which first off, the mutual information scores for each pair of words is calculated, and then the score is normalized to obtain a translation probability.

As mentioned before, another method of semantic matching is topic modeling. Given a collection of documents, topic modeling techniques aim to discover the topics in the collection as well as the topic representations of the documents [14]. One of the most popular methods for this approach is Latent Dirichlet Allocation (LDA) introduced by Blei *et al.* [43]. It is by far the most widely used method in many machine learning, natural language processing, and information retrieval applications [16]. Wei and Croft [44] applied this model to language model based information retrieval and compared it with probabilistic latent semantic indexing and cluster-based

retrieval. Momtazi and Naumann [16] have adopted this method for expert finding.

Over the past few years, there has been a growing trend towards Word Embedding (WE) approaches [45]. These models learn continuous-valued distributed representations of words known as embeddings [46, 47] in order to reduce the high dimensionality of words representations in contexts and increase generalization by introducing the expectation that similar word vectors signify semantically or syntactically similar words. WE has been used in many domains such as expert finding, [35], product search [48] and etc. In [35], the authors have proposed an unsupervised semantic matching method for expert finding. They directly utilize the words as the features for expert finding task while in our WE approach we consider each skill area as a query and words are acting like a bridge between skill areas and candidates. Besides, we observed that query words are not commonly used in the body of posts. Therefore, we first find important words in each skill area (translation step), then we exploit this words to score candidates. Whereas, in [35] words are directly used for expert finding which would not be able to alleviate the vocabulary gap problem. Finally, In [35] the words are presented in one-hot representation which makes it challenging to run in domains with extensive vocabulary size, such as StackOverflow. To overcome this problem, we have utilized the LDA algorithm for word representation.

9. Conclusion and Future work

In this paper, we studied the problem of vocabulary gap between the expert finding query and terms which candidates use in their documents in StackOverflow. We first illustrated that by utilizing appropriate translations, we can overcome the mentioned gap. Additionally, a concept was defined in this study. Voteshare which exploits the way of identifying high-quality answers in the community. We used this concept to improve the precision of expert finding task. Then we proposed two translation models based on statistical co-occurrence of words and the word embedding approach. The main finding in this paper is that utilizing both the translation models and considering the quality of documents simultaneously can significantly improve the quality of expert finding in StackOverflow. Future work may target the diversification aspect of translation to select a diverse set of words for each query. Additionally, the effectiveness of the proposed models in other domains rather than programming CQAs could be analyzed in future works.

Furthermore, models capable of translating each skill area into phrasal or multi-term words, are left for future work.

References

[1] S. Sotudeh Gharebagh, P. Rostami, M. Neshati, T-shaped mining: A novel approach to talent finding for agile software teams, in: Advances in Information Retrieval, Springer International Publishing, Cham, 2018, pp. 411–423.

[2] D. van Dijk, M. Tsagkias, M. de Rijke, Early detection of topical expertise in community question answering, in: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, Santiago, Chile, August 9-13, 2015, 2015, pp. 995–998.

[3] A. Dargahi Nobari, S. Sotudeh Gharebagh, M. Neshati, Skill translation models in expert finding, in: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '17, ACM, 2017, pp. 1057–1060.

[4] G. Zhou, J. Zhao, T. He, W. Wu, An empirical study of topic-sensitive probabilistic model for expert finding in question answer communities, Knowledge-Based Systems 66 (2014) 136 – 145.

[5] W. Wei, G. Cong, C. Miao, F. Zhu, G. Li, Learning to find topic experts in twitter via different relations, IEEE Transactions on Knowledge and Data Engineering 28 (7) (2016) 1764–1778. doi:10.1109/TKDE.2016.2539166.

[6] H. Deng, I. King, M. R. Lyu, Enhanced models for expertise retrieval using community-aware strategies, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 42 (1) (2012) 93–106. doi:10.1109/TSMCB.2011.2161980.

[7] M. Neshati, S. H. Hashemi, H. Beigy, Expertise finding in bibliographic network: Topic dominance learning approach, IEEE Transactions on Cybernetics 44 (12) (2014) 2646–2657.

[8] Stackoverflow candidate search, http://business.stackoverflow.com/careers/us/platform/candidate-search, accessed: 26-July-2017.

[9] Stackoverflow job listings, http://business.stackoverflow.com/careers/us/platform/job-listings, accessed: 26-July-2017.

[10] Z. Zhao, L. Zhang, X. He, W. Ng, Expert finding for question answering via graph regularized matrix completion, IEEE Transactions on Knowledge and Data Engineering 27 (4) (2015) 993–1004. doi:10.1109/TKDE.2014.2356461.

[11] M. Karimzadehgan, R. White, M. Richardson, Enhancing expert finding using organizational hierarchies, Advances in Information Retrieval (2009) 177–188.

[12] S. Ravi, B. Pang, V. Rastogi, R. Kumar, Great question! question quality in community q&a., in: ICWSM, 2014.

[13] K. Balog, Y. Fang, M. de Rijke, P. Serdyukov, L. Si, Expertise retrieval, Foundations and Trends in Information Retrieval 6 (2-3) (2012) 127–256. doi:10.1561/1500000024.

[14] H. Li, J. Xu, et al., Semantic matching in search, Foundations and Trends in Information Retrieval 7 (5) (2014) 343–469.

[15] M. Karimzadehgan, C. Zhai, Estimation of statistical translation models based on mutual information for ad hoc information retrieval, in: Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval, ACM, 2010, pp. 323–330.

[16] S. Momtazi, F. Naumann, Topic modeling for expert finding using latent dirichlet allocation., Wiley Interdisc. Rew.: Data Mining and Knowledge Discovery 3 (5) (2013) 346–353.

[17] Stackoverflow help center, https://stackoverflow.com/help/accepted-answer, accessed: 29-July-2017.

[18] M. Neshati, On early detection of high voted q&a on stack overflow, Inf. Process. Manage. 53 (4) (2017) 780–798.

[19] K. Balog, L. Azzopardi, M. de Rijke, A language modeling framework for expert finding, Information Processing & Management 45 (1) (2009) 1–19.

[20] C. D. Manning, P. Raghavan, H. Schütze, et al., Introduction to information retrieval, Vol. 1, Cambridge university press Cambridge, 2008.

[21] J. Yang, K. Tao, A. Bozzon, G. Houben, Sparrows and owls: Characterisation of expert behaviour in stackoverflow, in: User Modeling, Adaptation, and Personalization - 22nd International Conference, UMAP 2014, Aalborg, Denmark, July 7-11, 2014. Proceedings, 2014, pp. 266–277.

[22] M. D. Zeiler, ADADELTA: an adaptive learning rate method, CoRR abs/1212.5701.

[23] M. Neshati, H. Beigy, D. Hiemstra, Expert group formation using facility location analysis, Information Processing & Management 50 (2) (2014) 361 – 383.

[24] M. Neshati, H. Beigy, D. Hiemstra, Multi-aspect group formation using facility location analysis, in: Proceedings of the Seventeenth Australasian Document Computing Symposium, ADCS '12, 2012, pp. 62–71.

[25] A. Daud, J. Li, L. Zhou, F. Muhammad, Temporal expert finding through generalized time topic modeling, Knowledge-Based Systems 23 (6) (2010) 615 – 625.

[26] M. Neshati, D. Hiemstra, E. Asgari, H. Beigy, Integration of scientific and social networks, World Wide Web 17 (5) (2014) 1051–1079.

[27] H. Ziaimatin, T. Groza, G. Bordea, P. Buitelaar, J. Hunter, Expertise profiling in evolving knowledgecuration platforms, GSTF Journal on Computing (JoC) 2 (3).

[28] S. Budalakoti, R. Bekkerman, Bimodal invitation-navigation fair bets model for authority identification in a social network, in: Proceedings of the 21st International Conference on World Wide Web, WWW '12, ACM, New York, NY, USA, 2012, pp. 709–718.

[29] M. Neshati, Z. Fallahnejad, H. Beigy, On dynamicity of expert finding in community question answering, Information Processing & Management 53 (5) (2017) 1026 – 1042.

[30] P. Rostami, M. Neshati, T-shaped grouping: Expert finding models to agile software teams retrieval, Expert Systems with Applications 118 (2019) 231 – 245.

[31] A. Pal, A. Herdagdelen, S. Chatterji, S. Taank, D. Chakrabarti, Discovery of topical authorities in instagram, in: Proceedings of the 25th International Conference on World Wide Web, WWW '16, 2016, pp. 1203–1213.

[32] Y. Cao, J. Liu, S. Bao, H. Li, Research on expert search at enterprise track of trec 2005., in: TREC, 2005.

[33] H. Fang, C. Zhai, Probabilistic models for expert finding, Advances in Information Retrieval (2007) 418–430.

[34] Z. Zhao, Q. Yang, D. Cai, X. He, Y. Zhuang, Expert finding for community-based question answering via ranking metric network learning., in: IJCAI, 2016, pp. 3000–3006.

[35] C. Van Gysel, M. de Rijke, M. Worring, Unsupervised, efficient and semantic expertise retrieval, in: Proceedings of the 25th International Conference on World Wide Web, WWW '16, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 2016, pp. 1069–1079.

[36] A. Pal, Metrics and algorithms for routing questions to user communities, ACM Trans. Inf. Syst. 33 (3) (2015) 14:1–14:29.

[37] F. Riahi, Z. Zolaktaf, M. Shafiei, E. Milios, Finding expert users in community question answering, in: Proceedings of the 21st International Conference on World Wide Web, WWW '12 Companion, ACM, New York, NY, USA, 2012, pp. 791–798.

[38] A. Pal, R. Farzan, J. Konstan, R. Kraut, Early detection of potential experts in question answering communities, User Modeling, Adaption and Personalization (2011) 231–242.

[39] Z. Zhao, F. Wei, M. Zhou, W. Ng, Cold-start expert finding in community question answering via graph regularization, in: M. Renz, C. Shahabi, X. Zhou, M. A. Cheema (Eds.), Database Systems for Advanced Applications, Springer International Publishing, Cham, 2015, pp. 21–38.

[40] M. J. Blooma, D. H. Goh, A. Y. Chua, Predictors of high-quality answers, Online Information Review 36 (3) (2012) 383–400.

[41] L. Ponzanelli, A. Mocci, A. Bacchelli, M. Lanza, Understanding and classifying the quality of technical forum questions, in: 2014 14th International Conference on Quality Software, 2014, pp. 343–352. doi:10.1109/QSIC.2014.27.

[42] W. B. Croft, M. Bendersky, H. Li, G. Xu, Query representation and understanding workshop, SIGIR Forum 44 (2) (2011) 48–53.

[43] D. M. Blei, A. Y. Ng, M. I. Jordan, Latent dirichlet allocation, Journal of machine Learning research 3 (Jan) (2003) 993–1022.

[44] X. Wei, W. B. Croft, Lda-based document models for ad-hoc retrieval, in: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '06, ACM, New York, NY, USA, 2006, pp. 178–185.

[45] W. Y. Zou, R. Socher, D. M. Cer, C. D. Manning, Bilingual word embeddings for phrase-based machine translation., in: EMNLP, 2013, pp. 1393–1398.

[46] A. Mnih, K. Kavukcuoglu, Learning word embeddings efficiently with noise-contrastive estimation, in: Advances in neural information processing systems, 2013, pp. 2265–2273.

[47] J. Pennington, R. Socher, C. D. Manning, Glove: Global vectors for word representation., in: EMNLP, Vol. 14, 2014, pp. 1532–1543.

[48] C. Van Gysel, M. de Rijke, E. Kanoulas, Learning latent vector spaces for product search, in: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, CIKM '16, ACM, New York, NY, USA, 2016, pp. 165–174.