Toward Faster and Better Retrieval Models for Question Search

Guangyou Zhou, Yubo Chen, Daojian Zeng, and Jun Zhao
National Laboratory of Pattern Recognition,
Institute of Automation, Chinese Academy of Sciences
95 Zhongguancun East Road, Beijing 100190, China
{gyzhou, yubo.chen, djzeng, jzhao}@nlpr.ia.ac.cn

ABSTRACT

Community question answering (cQA) has become an important service due to the popularity of cQA archives on the web. This paper is concerned with the problem of question search. Question search in cQA aims to find the historical questions that are semantically equivalent or similar to the queried questions. In this paper, we propose a faster and better retrieval model for question search by leveraging user chosen category. After introducing the question category, we can filter certain amount of irrelevant historical questions under a wide range of leaf categories. Experimental results conducted on real cQA data set demonstrate that the proposed techniques are more effective and efficient than a variety of baseline methods.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Storage and Retrieval; H.3.5 [**Information Systems and Applications**]: On-line Information Services

General Terms

Algorithms, Experimentation, Performance

Keywords

Question Search, Question Category, Translation Model, Language Model

1. INTRODUCTION

Over the past few years, large-scale question and answer archives have become an important information resource on the web. To make use of the large-scale archives of question-answer pairs, it is critical to have functionality of helping users to retrieve previous answers [9]. Therefore, it is a meaningful task to retrieve the semantically equivalent or similar questions to the queried questions, and then these answers of the retrieved questions will be used to answer the queried questions.

Recently, question search in cQA has gained a wide interest in NLP and IR communities. A series of conferences (ACL, EMNLP, COLING, SIGIR, WWW, and CIKM) have advanced the question search techniques and proposed several different retrieval models, such as the vector space model (VSM) [7, 11], the Okapi BM25 model [7, 11], the language model (LM) [6, 7, 9, 11], the translation model (TR) [1, 7, 11, 13], the translation-based language model (TRLM) [26], phrase-based translation model [20, 28], and the statistical machine translation enriched model [29, 30]. Experimental results consistently reported that the translation-based language model (TRLM) achieved the state-of-the-art performance for question search [26]. However, all these approaches focus on how to improve the performance of question search while ignoring *the efficiency* (computational cost of average running time for a search query). Efficiency is important for question search since question answer archives are huge[1] and they keep growing [6]. We note that applying these existing methods for question search suffers from the following problem:

- **Irrelevant historical questions:** For a queried question, all historical questions in the archives are involved in computing the similarity, although certain amount of historical questions under a wide range of categories might be irrelevant to the queried question.

As a result, the irrelevant historical questions would increase the computational cost of running time and hinder the efficiency of question search, rather than contribute to the performance of question search. Consider an example shown in Table 1. The estimated similarity of d_1 is higher than that of $\mathbf{d}_2$ and $\mathbf{d}_3$ to the queried question $\mathbf{q}$ by using the existing methods (VSM, BM25, LM, TR and TRLM). However, $\mathbf{d}_1$ is irrelevant to $\mathbf{q}$.

Table 1: An example for question search

Queried question:
$\mathbf{q}$: How do you make a java chip? in "Food & Drink" $\rightarrow$ "Non-Alcoholic Drinks" category
Irrelevant:
$\mathbf{d}_1$: How do you make java do algebra? in "Computers & Internet" $\rightarrow$ "Programming & Design" category
Relevant:
$\mathbf{d}_2$: how to make java mint frappuccino? in "Food & Drink" $\rightarrow$ "Non-Alcoholic Drinks" category
$\mathbf{d}_3$: How to make Starbucks Java chip? in "Food & Drink" $\rightarrow$ "Cooking & Recipes" category

cQA usually organizes questions into a hierarchy of categories. When a user asks a question, the user is typically required to choose

[1]Yahoo! Answers has more than 1 billion resolved questions as of May 1, 2010.

CIKM'13, Oct. 27–Nov. 1, 2013, San Francisco, CA, USA.

a category for the question from a predefined hierarchy of categories. The available categories of queried questions can be used to filter irrelevant questions in the archives and thus improve the efficiency of question search. Moreover, we note that not all the relevant questions come from the same category with the category of the queried question. The relevant questions under the similar categories might be exploited for further improving the performance of question search. For example in Table 1, the category "Non-Alcoholic Drinks" of $\mathbf{q}$ will filter the irrelevant historical questions $\mathbf{d}_1$, and the relevant questions $\mathbf{d}_2$ and $\mathbf{d}_3$ under the same and similar categories will be obtained relative higher ranks.

Although it appears natural to exploit the existing category information for question search, we are aware of only three published studies [6, 7, 5] on utilizing category information for question search. Cao et al. [6] employed classifiers to compute the probability of a queried question belonging to different categories. The performance of question search highly depends on the accuracy of classifiers. However, question classification in cQA is challenged by the large-scale hierarchical classification problem[2], the classification error leads to the retrieval results improvement only slightly [7]. Cao et al. [7] proposed a category enhanced retrieval model and computed the global relevance score with regard to the entire collection of questions, which greatly hinders the efficiency of question search, as we will show in the experiments. Cai et al. [5] incorporated the category information into TRLM for further improving the performance while ignoring the efficiency.

In this paper, we aim to balance between effectiveness (better performance) and efficiency (lower computation cost) for question search by leveraging user chosen category information. Compared to [6, 7, 5], our proposed method is much faster and better. To the best of our knowledge, it is the first work to give a thorough analysis between effectiveness and efficiency in studies of question search in cQA.

The contribution of this paper is expected in the following two aspects:

- **Lower computational cost:** Computational cost of average running time for a search query will be substantially reduced because only a small number of relevant questions in the archives will be involved in.
- **Better performance:** We consider the relevant questions not only from the same category but also from the similar categories with the category of the queried question.

The rest of this paper is organized as follows. Section 2 describes the related work. In section 3, we give a brief a introduction of the existing retrieval models for question search. Section 4 presents our proposed faster and better retrieval models for question search by leveraging the user chosen question category information. Experimental results are presented in section 5. Finally, we conclude with ideas for future work in section 6.

[2]In cQA, there are more than 1,200 leaf categories organized into a hierarchical structure, the large-scale top-down hierarchical classification approach used in [6] suffers from the following problems as discussed in [25]: (1) misclassification at a parent or ancestor category may force a question to be excluded from the child categories; (2) the classification over high-level categories may fail easily since some of the categories are too general and thus harder to discriminative. Cao et al. [6] reported the Micro F_1-score of hierarchical question classification was only 45.59%. Therefore, the performance of question search is hindered by the classification error.

2. RELATED WORK

The research of question search has been further extended to the cQA data. The major challenge for question search in cQA is the word ambiguity and lexical gap problems. Jeon et al. [11] proposed a word-based translation model for automatically fixing the lexical gap problem. Xue et al. [26] proposed a word-based translation language model for question search. The results indicated that word-based translation language model further improved the retrieval results and obtained the state-of-the-art performance. Subsequent work on word-based translation models focused on providing suitable parallel data to learn the translation probabilities. Lee et al. [13] tried to further improve the translation probabilities based on question-answer pairs by selecting the most important terms to build compact translation models. Bernhard and Gurevych [1] proposed to use as a parallel training data set the definitions and glosses provided for the same term by different lexical semantic resources. Cao et al. [7] explored the category information into the word-based translation model for question search.

In order to improve the word-based translation model with some contextual information, Riezler et al. [20] and Zhou et al. [28] proposed a phrase-based translation model for question and answer retrieval. The phrase-based translation model can capture some contextual information in modeling the translation of phrases as a whole, thus the word ambiguity and lexical gap problems are somewhat alleviated. Singh [21] addressed the lexical gap issues by extending the lexical word-based translation model to incorporate semantic information (entities).

Recently, Zhou et al. [29] argued that the effectiveness of the above translation models were highly dependent on the availability of quality parallel monolingual corpora (e.g., question-answer pairs) in the absence of which they were troubled by noise issue. Therefore, Zhou et al. [29] proposed an alternative way to address the word ambiguity and word mismatch problems by taking advantage of potentially rich semantic information drawn from other languages. Furthermore, Zhou et al. [30] integrated the semantic knowledge drawn from other languages with matrix factorization in order to solve the data sparseness and noised introduced by statistical machine translation.

Besides, some other studies model the semantic relationship between the queried questions and the candidate answers with deep question analysis or a learning-to-ranking strategy. Duan et al. [9] proposed to conduct question search by identifying question topic and question focus. Surdeanu et al. [19] proposed an approach to rank the answers retrieved by Yahoo! Answers with multiple features. Wang et al. [24] aimed to rank the candidate answers with only word information instead of the combination of different kinds of features.

However, all these existing approaches focus on how to improve the performance of question search while ignoring *the efficiency* (computational cost of average running time for a search query). In this paper, we aim to balance between effectiveness (better performance) and efficiency (lower computation cost) for question search by leveraging user chosen category information. Although some studies utilized category information for question search [6, 7, 5], they focused on improving the performance of question search while hindering the efficiency. To the best of our knowledge, it is the first work to give a thorough analysis between effectiveness and efficiency in studies of question search in cQA.

3. RETRIEVAL MODELS

In this section, we describe some existing retrieval models for question search.

3.1 Vector Space Model

The vector space model (VSM) has been widely used for question search [11, 7]. Given a queried question $\mathbf{q}$ and a historical question $\mathbf{d}$ in the archives, the similarity score function can be computed as follows:

$$P(\mathbf{q}|\mathbf{d}) = \frac{\sum_{t\in\mathbf{q}\cap\mathbf{d}} w_{\mathbf{q},t} \times w_{\mathbf{d},t}}{\sqrt{\sum_t w_{\mathbf{q},t}^2}\sqrt{\sum_t w_{\mathbf{d},t}^2}} \quad (1)$$

$$w_{\mathbf{q},t} = \ln(1 + \frac{N}{f_t}), w_{\mathbf{d},t} = 1 + \ln(tf_{\mathbf{d},t})$$

where $w_{\mathbf{q},t}$ denotes the IDF (inverse document frequency) of term t in the collection, and $w_{\mathbf{d},t}$ denote the TF (term frequency) of term t in $\mathbf{d}$. N is the number of questions in the whole collection, f_t is the number of questions containing the term t, and $tf_{\mathbf{d},t}$ is the frequency of term t in $\mathbf{d}$.

3.2 BM25 Model

BM25 model takes into account the question length to overcome the shortcoming of VSM [16]. Following [7], the similarity score function between a queried question $\mathbf{q}$ and a historical question $\mathbf{d}$ in the archives can be computed as follows:

$$P(\mathbf{q}|\mathbf{d}) = \sum_{t\in\mathbf{q}\cap\mathbf{d}} w_{\mathbf{q},t} \times w_{\mathbf{d},t} \quad (2)$$

$$w_{\mathbf{q},t} = \ln(\frac{N - f_t + 0.5}{f_t + 0.5})\frac{(k_2+1)tf_{\mathbf{q},t}}{k_2 + tf_{\mathbf{q},t}}$$

$$w_{\mathbf{d},t} = \frac{(k_1+1)tf_{\mathbf{d},t}}{K_{\mathbf{d}} + tf_{\mathbf{d},t}}, K_{\mathbf{d}} = k_1((1-b) + b\frac{|\mathbf{d}|}{W_A})$$

where k_1, b, and k_2 are parameters that are set to 1.2, 0.75, and ∞, respectively. $|\mathbf{d}|$ is the question length of $\mathbf{d}$ and W_A is the average question length in the collection.

3.3 Language Model

The unigram language model (LM) is often used and assumes that each term is generated independently. It concerns only the probabilities of sampling a single word by the maximum likelihood. To avoid zero probability, we use Jelinek-Mercer smoothing [27] due to its good performance and cheap computational cost. So the ranking function for the query likelihood language model with Jelinek-Mercer smoothing can be written as:

$$P(\mathbf{q}|\mathbf{d}) = \prod_{w\in\mathbf{q}} P_{LM}(w|\mathbf{d}) \quad (3)$$

$$P_{LM}(w|\mathbf{d}) = (1-\lambda)P_{ml}(w|\mathbf{d}) + \lambda P_{ml}(w|\mathcal{C}) \quad (4)$$

$$P_{ml}(w|\mathbf{d}) = \frac{\#(w,\mathbf{d})}{|\mathbf{d}|}, \quad P_{ml}(w|\mathcal{C}) = \frac{\#(w,\mathcal{C})}{|\mathcal{C}|} \quad (5)$$

where $\mathcal{C}$ is background collection, λ is smoothing parameter. $\#(w,\mathbf{d})$ is the frequency of term w in $\mathbf{d}$, $|\mathbf{d}|$ and $|\mathcal{C}|$ denote the length of $\mathbf{d}$ and $\mathcal{C}$, respectively.

3.4 Translation Model

Previous work [7, 11, 26] consistently reported that the translation model (TR) yielded superior performance for question search. This model exploits the word translation probabilities in a language modeling framework. Following [11, 26], the ranking function can be written as:

$$P(\mathbf{q}|\mathbf{d}) = \prod_{w\in\mathbf{q}} P_{TR}(w|\mathbf{d}) \quad (6)$$

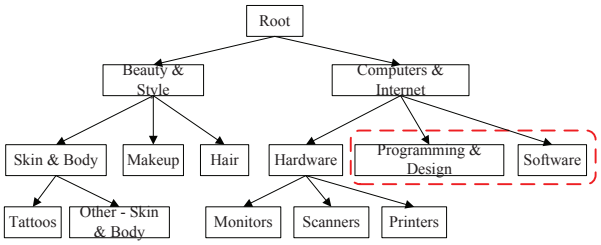

Figure 1: An example of category hierarchy in Yahoo! Answers.

$$P_{TR}(w|\mathbf{d}) = (1-\lambda)\Big[\sum_{t\in\mathbf{d}} P(w|t)P_{ml}(t|\mathbf{d})\Big] + \lambda P_{ml}(w|\mathcal{C}) \quad (7)$$

$$P_{ml}(t|\mathbf{d}) = \frac{\#(t,\mathbf{d})}{|\mathbf{d}|} \quad (8)$$

where $P(w|t)$ denotes the translation probability from word t to word w. Jeon et al. [11] assume that the probability of self-translation is 1, meaning that $P(t|t) = 1$.

3.5 Translation-Based Language Model

Xue et al. [26] proposed to linearly mix two different estimations by combining the language model with the translation model into a unified framework, called translation-based language model (TRLM). It is shown that this model gains better performance than both the language model and the translation model [26]. The model can be written as:

$$P(\mathbf{q}|\mathbf{d}) = \prod_{w\in\mathbf{q}} P_{TRLM}(w|\mathbf{d}) \quad (9)$$

$$\begin{aligned} P_{TRLM}(w|\mathbf{d}) =& (1-\lambda)\Big[\alpha\sum_{t\in\mathbf{d}} P(w|t)P_{ml}(t|\mathbf{d}) \\ &+(1-\alpha)P_{ml}(w|\mathbf{d})\Big] + \lambda P_{ml}(w|\mathcal{C}) \end{aligned} \quad (10)$$

where parameter α controls the impact of translation component.

4. THE PROPOSED APPROACHES

4.1 Basic Category-Sensitive Retrieval Model

As discussed above, the previous works on question search focus on how to improve the performance while ignore the efficiency (computational cost of running time) of question search. For a given queried question $\mathbf{q}$, all historical questions in the archives are involved in similarity computation, although certain amount of historical questions under a wide range of categories irrelevant to $\mathbf{q}$. These irrelevant historical questions will increase the computational cost rather than improve the performance of question search. To solve the above problem, we propose a faster and better retrieval model by leveraging question category information to filter the irrelevant historical questions for question search. In cQA, all questions are usually organized into a hierarchy of categories. Figure 1 shows an example of part of category hierarchy of Yahoo! Answers. When a user asks a question, the user typically required to choose a category label for the question from a predefined hierarchy of categories. Hence, each question in cQA has a category label. Let $C = \{c_1, c_2, \cdots, c_n\}$ denote all leaf categories. The basic category-sensitive (BCS) question search is defined as follows:

$$P_{BCS}(\mathbf{d}|\mathbf{q}, c_i) \propto P_{BCS}(\mathbf{q}, c_i|\mathbf{d})P(\mathbf{d}) \quad (11)$$

$$P_{BCS}(\mathbf{q}, c_i|\mathbf{d}) = P_{BCS}(\mathbf{q}|c_i, \mathbf{d})P(c_i|\mathbf{d}) \quad (12)$$

$$P(c_i|\mathbf{d}) = \begin{cases} 1 & \text{if } c_i = c(\mathbf{d}) \\ 0 & \text{otherwise} \end{cases} \quad (13)$$

where c_i is the category of queried question $\mathbf{q}$, and $c(\mathbf{d})$ is the category of historical question $\mathbf{d}$. $P_{BCS}(\mathbf{q}|c_i, \mathbf{d})$ is category-specific question search, which can be calculated using the existing methods (VSM, BM25, LM, TR and TRLM) with some minor modifications, as we will describe in subsection 3.3. By introducing the question category information, only a small number of the historical questions under the same leaf category with the category of a queried question are employed, we have

$$P_{BCS}(\mathbf{q}, c_i|\mathbf{d}) = P_{BCS}(\mathbf{q}|c(\mathbf{d}), \mathbf{d}) \quad (14)$$

According to equation (14), the computational cost of running time for a search query is substantially reduced and the efficiency of question search is greatly enhanced.

4.2 Related Category-Sensitive Retrieval Model

In subsection 4.1, the basic category-sensitive retrieval model in equations (11) and (12) is based on the **same leaf category** assumption, with potential relevant questions under the **similar leaf categories** being omitted. As shown in Figure 1, there exists several similar leaf categories under one main category. For example, "Programming & Design" and "Software" are two similar leaf categories under the main category "Computers & Internet". Questions under the leaf category "Programming & Design" may also be relevant with questions under the leaf category "Software". Based on these observations, we propose a related category-sensitive (RCS) retrieval model by taking into account the relevant questions under the similar leaf categories:

$$P_{RCS}(\mathbf{q}, c_i|\mathbf{d}) = \frac{1}{A}\Big\{\gamma P_{BCS}(\mathbf{q}, c_i|\mathbf{d}) + \sum_{c_j \in Related(c_i)} R(c_j \to c_i) P_{BCS}(\mathbf{q}, c_j|\mathbf{d})\Big\} \quad (15)$$

$$A = \gamma + \sum_{c_j \in Related(c_i)} R(c_j \to c_i) \quad (16)$$

where γ controls the relative impact between the original leaf category and other similar leaf categories. If we set a large value of γ, the importance of the original leaf category is emphasized. $Related(c_i)$ denotes the set of similar leaf categories which are related to (similar to) c_i, and $R(c_j \to c_i)$ represents the similar probability from category c_j to category c_i.[3]

4.2.1 Category Similarity Formulation

In this paper, we define $Related(c_i)$ as follows:

$$c_j \in Related(c_i) \quad \text{if} \quad R(c_j \to c_i) \geq \delta \quad (17)$$

where δ is a threshold between 0 and 1.

To estimate the similar probability between two categories, answerer-based and content-based methods used in [14] can be naturally employed. However, we observe that some leaf categories consist of only a small number of questions, which may lead to the data sparseness. In this paper, we propose to leverage topic models for inferring similar probability between two categories. The basic assumption is that two categories are similar because their probabilities of belonging to the same latent topic are similar. For example in Figure 1, leaf categories "Monitors", "Scanners" and "Printers" are similar because they all belong to the latent topic *Computer Hardware*.

[3]In cQA (e.g., Yahoo! Answers), when a user asks a new question, the user has to choose a particular category for the question. The cQA system allows the askers to choose only one leaf category for each question. So the most similar category is not allowed to manually assign to each category when adding it in cQA system. This motivates us to automatically calculate the category similarity.

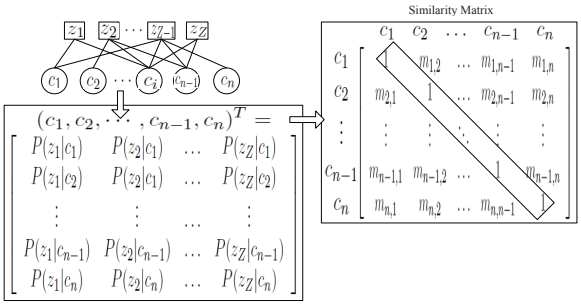

Figure 2: Category similarity matrix.

4.2.2 Inferring Category Similarity Using LDA

To leverage topic model for inferring category similarity, we use the widely studied topic model −− Latent Dirichlet Allocation (LDA) [3] to identify the latent topic information from the large scale question-answer collection. LDA models each document as a mixture of underlying topics and generates each word from one topic. To identify the topics that each leaf category is about using LDA, we aggregate all questions under the same leaf category into a big document. Thus, each document essentially corresponds to a leaf category. After utilizing LDA, each leaf category c can be represented as a Z-dimension vector topic distribution $P(z|c)$, where Z is the topic number. Thus, the task of inferring category similarity probability is converted to calculate the distance between two leaf category vectors. In this paper, we propose to use normalized Kullback Leibler (KL) divergence [12], which is an asymmetric measure for measuring category similarity. The KL-divergence from c_j to c_i is computed by $P_{KL}(c_j||c_i) = \sum_z P(z|c_j)\log\frac{P(z|c_j)}{P(z|c_i)}$. Then we calculate the similarity between leaf categories c_j to c_i using Jensen Shannon divergence, which shows the superior performance than others. Thus, we have

$$R(c_j \to c_i) = \frac{1}{2}\Big\{P_{KL}(c_j||c_i) + P_{KL}(c_i||c_j)\Big\} \quad (18)$$

The larger $R(c_j \to c_i) \in [0, 1]$, the more similar c_j is for c_i. After calculating the similarity between each pair of leaf categories, we can obtain the similarity matrix $M_C = \{m_{ji} = R(c_j \to c_i)\}$, where $i, j \in [1, n]$, n is number of leaf category. Figure 2 shows an example of generating the category similarity matrix. From M_C, we can easily find the similarity between two leaf categories.

4.3 Category-Specific Retrieval Models

In subsection 4.1 and 4.2, we describe a faster and better retrieval model for question search by leveraging question category information. Now we focus on how to calculate the category-specific retrieval model $P_{BCS}(\mathbf{q}|c(\mathbf{d}), \mathbf{d})$ in equation (14) by using the existing question search models.

4.3.1 Vector Space Model

Computing the category-specific question search with regard to only the category containing the historical question appears to be

preferable. This observation motivates us to compute the category-specific question search differently from the standard VSM. The category-specific retrieval model $P_{BCS}(\mathbf{q}|c(\mathbf{d}),\mathbf{d})$ is computed by using equation (1) with the following modifications: N is replaced by $N_{c(\mathbf{d})}$, the number of questions in category $c(\mathbf{d})$; f_t is replaced by $f_{t,c(\mathbf{d})}$, the number of questions containing the term t in category $c(\mathbf{d})$. We modify $w_{\mathbf{q},t}$ as follows:

$$w_{\mathbf{q},t} = \ln(1 + \frac{N_{c(\mathbf{d})}}{f_{t,c(\mathbf{d})}}) \quad (19)$$

4.3.2 BM25 model

Similar to VSM, the category-specific retrieval model $P_{BCS}(\mathbf{q}|c(\mathbf{d}),\mathbf{d})$ is computed by using equation (2) with the following modifications: N is replaced with $N_{c(\mathbf{d})}$; f_t and W_A are computed with regard to the category $c(\mathbf{d})$ and are replaced by $f_{t,c(\mathbf{d})}$ and $W_{A,c(\mathbf{d})}$, respectively, where $f_{t,c(\mathbf{d})}$ is the document frequency of t in category $c(\mathbf{d})$, and $W_{A,c(\mathbf{d})}$ is the average question length in the category $c(\mathbf{d})$. Specifically, compared to equation (2), the following modifications are made:

$$w_{\mathbf{q},t} = \ln(\frac{N_{c(\mathbf{d})} - f_{t,c(\mathbf{d})} + 0.5}{f_{t,c(\mathbf{d})} + 0.5}) \frac{(k_2+1)tf_{\mathbf{q},t}}{k_2 + tf_{\mathbf{q},t}} \quad (20)$$

$$K_{\mathbf{d}} = k_1((1-b) + b\frac{|\mathbf{d}|}{W_{A,c(\mathbf{d})}}) \quad (21)$$

4.3.3 LM, TR, and TRLM

The smoothing in LM, TR and TRLM plays an IDF-like role. Therefore, we compute the smoothing value with regard to the category rather than the whole collection, i.e., we use $P_{ml}(w|\mathcal{C}_{c_i})$ to do the smoothing instead of $P_{ml}(w|\mathcal{C})$ when computing the category-specific retrieval model $P_{BCS}(\mathbf{q}|c_i,\mathbf{d})$ in equations (4), (7), and (10), where $\mathcal{C}_{c_i}$ denotes all questions in category c_i.

4.4 Learning Word Translation Probabilities

The performance of the translation model and the translation-based language model will rely on the the word-to-word translation probabilities. In our experiments, question-answer pairs are used for training, and the GIZA++ toolkit is used to learn the IBM translation model 1. IBM model 1 is a widely used word alignment algorithm which does not require linguistic knowledge for two languages[4].

IBM model 1 assumes each translation pair should be of comparable length. However, an answer is usually much longer than the corresponding question. It will hurt the performance if we will fill the length-unbalanced pairs for training. To address the length-unbalanced problem, we propose a word sampling method for each answer to make it comparable to the length of the corresponding question. Suppose the lengths of an answer and the corresponding question are $|a|$ and $|q|$, respectively. For answer a, we first build a bag of words $\mathbf{b}_a = \{(w_i, e_i)_{i=1}^{W_a}\}$, where W_a is the number of unique words in a, and e_i is the weights of word w_i in a.

We use TF*IDF scores as the weights of words. Using $\mathbf{b}_a$, we sample words for $|q|$ times with replacement according to the weights of words, and finally form a new bag with $|q|$ words to represent answer a. In the sampling result, we keep the most important words in answer a. We can thus construct a question-answer pair with balance length.

[4]We have also employed more sophisticated algorithms such as IBM model 3 for this task. However, these methods do not achieve better performance than the simple IBM model 1. Therefore, we only demonstrate the experimental results using IBM model 1 in this paper.

Table 2: Number of questions in each first-level category

Category	#Size	Category	# Size
Arts & Humanities	86,744	Home & Garden	35,029
Business & Finance	105,453	Beauty & Style	37,350
Cars & Transportation	145,515	Pet	54,158
Education & Reference	80,782	Travel	305,283
Entertainment & Music	152,769	Health	132,716
Family & Relationships	34,743	Sports	214,317
Politics & Government	59,787	Social Science	46,415
Pregnancy & Parenting	43,103	Ding out	46,933
Science & Mathematics	89,856	Food & Drink	45,055
Computers & Internet	90,546	News & Events	20,300
Games & Recreation	53,458	Environment	21,276
Consumer Electronics	90,553	Local Businesses	51,551
Society & Culture	94,470	Yahoo! Products	150,445

Given the length-balanced question-answer pairs, IBM model 1 can be trained using Expectation-Maximization (EM) algorithm [8] in an unsupervised fashion. Using IBM model 1, we can obtain the translation probabilities of two language-sides, i.e., $P(t|w)$ and $P(w|t)$, where w is a word in answer side and t is a word in question side.

IBM model 1 will produce one-to-many alignments from one side to another side, and the trained model is thus asymmetric. Hence, we can train two different translation models by assigning translation pairs in two directions, i.e., (answer $\rightarrow$ question) and (question $\rightarrow$ answer). We denote the former model P_{a2q} and the latter as P_{q2a}. We define the final translation probability $P(t|w)$ as the harmonic mean of the two models:

$$P(t|w) \propto \Big(\frac{\beta}{P_{a2q}(t|w)} + \frac{1-\beta}{P_{q2a}(t|w)}\Big)^{-1} \quad (22)$$

where β is the harmonic factor to combine the two models. When $\beta = 1.0$ or $\beta = 0.0$, it simply uses model P_{a2q} or P_{q2a}, correspondingly.

5. EXPERIMENTS

5.1 Data Set

We collect the questions from Yahoo! Answers and use the *getByCategory* function provided in Yahoo! Answers API[5] to obtain cQA threads from the Yahoo! site. More specifically, we utilize the *resolved* questions and the resulting question repository that we use for question search contains 2,288,607 questions. Each resolved question consists of four parts: "question title", "question description", "question answers" and "question category". For question search, we only use the "question title" part. It is assumed that the titles of the questions already provide enough semantic information for understanding the users' information needs [9]. There are 26 categories at the first level and 1,262 categories at the leaf level. Each question belongs to a unique leaf category. Table 2 shows the distribution across first-level categories of the questions in the archives. To learn the word-to-word translation probabilities, we use the GIZA++ alignment toolkit[6] trained on one million question-answer pairs from another data set.[7]

We use the same test set in previous work [6, 7]. This set contains 252 queried questions and can be freely downloaded for research

[5]http://developer.yahoo.com/answers

[6]http://www-i6.informatik.rwth-aachen.de/Colleagues/och/software/GIZA++.html

[7]The Yahoo! Webscope dataset Yahoo answers comprehensive questions and answers version 1.0, available at http://reseach.yahoo.com/Academic_Relations.

communities.[8] For each method, the top 20 retrieval results are kept. Given a returned result for each queried question, an annotator is asked to label it with "relevant" or "irrelevant". If a returned result is considered semantically equivalent to the queried question, the annotator will label it as "relevant"; otherwise, the annotator will label it as "irrelevant". Two annotators are involved in the annotation process. If a conflict happens, a third person will make judgement for the final result. In the process of manually judging questions, the annotators are presented only the questions.

5.2 Evaluation Metrics

We evaluate the performance of our approach using the following metrics: **Mean Average Precision** (MAP) and **Precision@N** (P@N), as they are widely used in evaluation the performance of retrieval result [15].

- **MAP:** For a set of queried questions Q, MAP measures the mean of the average precision for each queried question $\mathbf{q}$ for a method $\mathcal{M}$:

$$\text{MAP} = \frac{\sum_{\mathbf{q}\in Q} \text{Avg}P(\mathbf{q})}{|Q|} \tag{23}$$

$$\text{Avg}P(\mathbf{q}) = \frac{1}{N_{\mathcal{M}_\mathbf{q}}} \sum_{j=1}^{|\mathcal{M}_\mathbf{q}|} \frac{N_{\mathcal{M}_{\mathbf{q},j}}}{j} \mathbf{1}(\mathcal{M}_{\mathbf{q},j}) \tag{24}$$

where $\mathbf{1}(\mathcal{S})$ is an indicator function which returns 1 when $\mathcal{M}_{\mathbf{q},j}$ is relevant based on the test collection we constructed. $N_{\mathcal{M}_{\mathbf{q},j}}$ denotes the number of relevant questions among the top j ranked list returned by $\mathcal{M}$ for queried question $\mathbf{q}$, and $N_{\mathcal{M}_\mathbf{q}}$ denotes the total number of relevant questions of queried question $\mathbf{q}$ returned by a method $\mathcal{M}$, and $\mathcal{M}_{\mathbf{q},j}$ is the j-th question generated by method $\mathcal{M}$ for queried question $\mathbf{q}$. MAP rewards methods that return relevant questions early and also rewards correct ranking of the results.

- **P@N:** For a set of queried questions Q, P@10 is the fraction of the top n retrieved questions that are relevant to the queried questions for a method $\mathcal{M}$:

$$P@n = \frac{1}{|Q|} \sum_{\mathbf{q}\in Q} \frac{N_{\mathcal{M}_{\mathbf{q},N}}}{N} \tag{25}$$

where $N_{\mathcal{M}_{\mathbf{q},N}}$ denotes the number of relevant questions among the top N ranked list returned by a method $\mathcal{M}$ for queried question $\mathbf{q}$.

5.3 Parameter Selection

The experiments use several parameters. The first two are smoothing parameters λ and α; the third γ, controls the relative importance between the original leaf category and other similar leaf categories; the fourth parameter δ, is a threshold between 0 and 1; the last parameter β, controls the translation directions.

In the traditional LM in equation (4), the traditional translation model (TR) in equation (7), and the traditional translation-based language model (TRLM) in equation (7), we follow the previous work presented in [6, 7, 11, 26, 27] to select the parameters and finally empirically set $\lambda = 0.2$ and $\alpha = 0.8$.

For parameters λ and α used in the category-specific retrieval models, different models need different combination values. We do an experiment on a small development set of 50 questions to determine the optimal values among 0.1, 0.2, $\cdots$, 0.9 in terms of MAP for each category-specific retrieval model. This set is also extracted from the Yahoo! Answers data set, and it is not included in the test set. As a result, we set $\lambda = 0.3$ and $\alpha = 0.7$ for category-specific LM, TR, and TRLM, respectively.

[8]The data set is available at http://homepages.inf.ed.ac.uk/gcong/qa/

Table 3: Computational cost of different methods by using the category information to filter the irrelevant questions, where running time (in seconds) is on a PC with 4G of memory and a 2.5Ghz CPU with Java programming language.

#	Methods	#Num	Time
1	VSM	2,288,607	0.157
2	BCS_VSM	2,413 (↓99.89%)	0.026 (↓83.44%)
3	RCS_VSM	101,466 (↓95.57%)	0.035 (↓77.71%)
4	BM25	2,288,607	0.096
5	BCS_BM25	2,413 (↓99.89%)	0.012 (↓87.50%)
6	RCS_BM25	101,466 (↓95.57%)	0.020 (↓79.17%)
7	LM	2,288,607	0.351
8	BCS_LM	2,413 (↓99.89%)	0.058 (↓83.48%)
9	RCS_LM	101,466 (↓95.57%)	0.064 (↓81.77%)
10	TR	2,288,607	0.655
11	BCS_TR	2,413 (↓99.89%)	0.072 (↓89.01%)
12	RCS_TR	101,466 (↓95.57%)	0.075 (↓88.55%)
13	TRLM	2,288,607	0.672
14	BCS_TRLM	2,413 (↓99.89%)	0.077 (↓88.54%)
15	RCS_TRLM	101,466 (↓95.57%)	0.083 (↓87.66%)

For parameter γ, we do an experiment on a small development set of 50 questions to determine the best value among 1, 2, $\cdots$, 9 in terms of MAP. As a result, we set $\gamma = 4$ in the experiments empirically as this setting yields the best performance.

The key success is how to balance the effectiveness and efficiency in the best way. In this paper the balance is controlled by parameter δ, we do an experiment on this development set to determine the best value among 0.05, 0.1, $\cdots$, 0.45 in terms of the computational cost and the performance (MAP) of question search. Figure 3(a) shows the influence of δ on the computational cost of average running time for a search query, while Figure 3(b) shows the influence of δ on the performance (MAP) of question search. Finally, we set $\delta = 0.25$ in the experiments as this setting yields the best way to balance between the the effectiveness and efficiency.

For parameter β, we do an experiment on a small development set of 50 questions to determine the best value among 0.1, 0.2, $\cdots$, 0.9 in terms of MAP. As a result, we set $\beta = 0.7$ in the experiments empirically as this setting yields the best performance.

Besides, we use LDA to infer the category similarity probability. In this paper, we set Dirichlet priors $\alpha' = 50/Z$, and $\beta' = 0.05$ as Griffiths and Steyvers [10]. We run LDA with 200 iterations of Gibbs sampling. After trying a few different number of topics, we empirically set $Z = 150$. We choose these parameter settings because they give coherent and meaningful topics for our data set.

5.4 Experimental Results

5.4.1 Lower Computational Cost

Efficiency is important for question search since historical questions in the archives are huge and they keep growing. By introducing the category information, we can filter certain amount of historical questions under a wide range of leaf categories, which are less likely to contain relevant questions to the queried questions. In other words, the different retrieval models will search questions only in the categories which are the same or similar to categories of the queried questions. This can greatly reduce the computational cost and save running time for a search query.

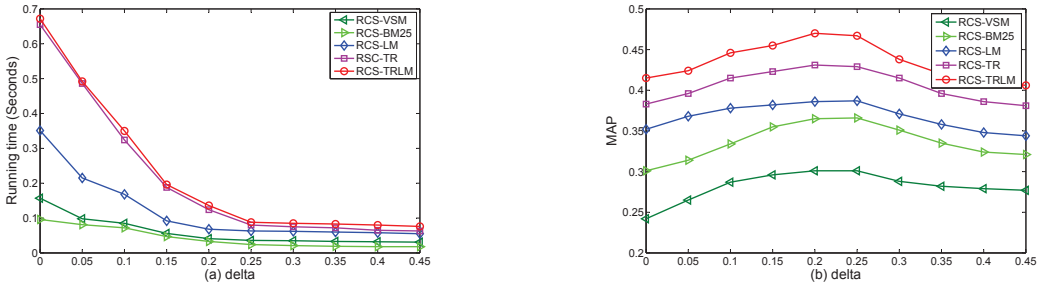

Figure 3: The influence of δ on (a) the computational cost of average running time for a search query and (b) the performance (MAP) of question search.

We first look into how well the proposed method benefits the efficiency of question search. Table 3 shows the impact of average number of historical questions and the running time for a search query by using the category information to filter the irrelevant questions. The experiments are conducted on a PC with 4G of memory and a 2.5Ghz CPU with Java programming language. When question search is based on the same leaf category assumption (queried questions and the historical questions come from the same leaf category), the average number of historical questions for a search query is reduced by 99.89%, with a significant decrease of search space by limiting search in a specific leaf category (row 1 vs. row 2; row 4 vs. row 5; row 7 vs. row 8; row 10 vs. row 11; row 13 vs. row 14).

A leaf category usually consists of only a small number of questions, thus search in a leaf category will be much more efficient than in the whole collection.[9] However, this simple assumption is not good in terms of performance. We note that not all relevant questions come from the same leaf categories with the categories of the queried questions. So we also consider the relevant questions under the similar leaf categories with the categories of the queried questions. This improved filtering strategy prunes the search space by limiting the search in a several similar leaf categories, with the average number of historical questions being reduced by 95.57% (row 1 vs. row 3; row 4 vs. row 6; row 7 vs. row 9; row 10 vs. row 12; row 13 vs. row 15).

Turn to the computational cost of average running time for a search query, we find that our proposed methods are more time efficient than the traditional methods and thus make question search faster (saving more than 80% time). Also, we find that BCS_TRLM (e.g., BCS_VSM, BCS_BM25, BCS_LM, BCS_TR) spends less time than RCS_TRLM (e.g., RCS_VSM, RCS_BM25, RCS_LM, RCS_TR) because the latter considers the historical questions under the similar leaf categories with categories of the queried questions. *Although the latter is more time consuming than the former, it is possible to reduce the running time by using the parallel computing since the category similarity calculation between the leaf categories is independent with each other*. We will leave it for future work.

5.4.2 Better Performance

There are some clear trends in Table 4 showing the performance of question search. First, note that the different methods using the question category information to filter irrelevant questions under a wide range of categories consistently outperform the baseline methods. Some of these improvements can be quite large; for example, the MAP of BCS_VSM and RCS_VSM increase that of VSM by 16.53% and 26.03% (row 1 vs. row2 and row 3). Second, when considering the relevant questions under the similar leaf categories with the categories of queried questions, the performance is further improved (row 2 vs. row 3; row 5 vs. row 6; row 8 vs. row 9; row 11 vs. row 12; row 14 vs. row 15).[10] To sum up, the results in Table 4 demonstrate the effectiveness of exploiting user chosen category information (the same or the similar leaf category information) for question search.

Table 4: The performance of question search by leveraging the question category information performed on a PC with 4G of memory and a 2.5Ghz CPU with Java programming language. Improvements over baseline methods are shown in parentheses.

#	Methods	MAP	P@10
1	VSM	0.242	0.176
2	BCS_VSM	0.282 (↑16.53%)	0.194 (↑10.23%)
3	RCS_VSM	0.305 (↑26.03%)	0.213 (↑21.02%)
4	BM25	0.301	0.208
5	BCS_BM25	0.347 (↑15.28%)	0.231 (↑11.06%)
6	RCS_BM25	0.368 (↑21.61%)	0.237 (↑13.94%)
7	LM	0.382	0.240
8	BCS_LM	0.425 (↑11.26%)	0.248 (↑3.33%)
9	RCS_LM	0.453 (↑18.59%)	0.267 (↑11.25%)
10	TR	0.396	0.245
11	BCS_TR	0.454 (↑14.65%)	0.268 (↑9.39%)
12	RCS_TR	0.471 (↑18.94%)	0.273 (↑11.43%)
13	TRLM	0.430	0.257
14	BCS_TRLM	0.463 (↑7.67%)	0.270 (↑5.06%)
15	RCS_TRLM	0.482 (↑12.09%)	0.275 (↑7.00%)

In order to have a better understanding why the proposed method can significantly outperforms the traditional methods, we manually check each queried question in the test set shown in Figure 4, where X axes represents the number of the similar leaf categories that the relevant questions come from, and Y axes represents the proportion of the relevant questions relative to the number of the similar leaf categories.

The results in Figure 4 show that the relevant questions come from the one leaf category only 42%, that is to say, more than half percentage questions come from the similar leaf categories with the category of queried questions. The figure validates why our proposed related category-sensitive methods significantly outperforms the basic category-sensitive methods (BCS_VSM vs. RCS_VSM; BCS_BM25 vs. RCS_BM25; BCS_LM vs. RCS_LM; BCS_TR vs. RCS_TR; BCS_TRLM vs. RCS_TRLM).

[9]In our data set, the number of questions in a leaf category is usually not exceeding 1% of the whole collection.

[10]These comparisons are statistically significant at $p < 0.05$

Table 5: Retrieval results for "How do i find out what kind of bird i have by looking on the internet?".

Models	Rank	Questions	Categories
TRLM	1st	Where can I find digital bird songs on the Internet?	Internet
	4th	**Were can i find out what kind of parrot i have?**	Birds
	15th	**How do you find out if abird is a girl or a boy?I don't know what kind of bird i have.?**	Birds
BCS_TRLM	1st	**Were can i find out what kind of parrot i have?**	Birds
	4th	I have a bird but I don't know what kind it is?	Birds
	5th	**How do you find out if abird is a girl or a boy?I don't know what kind of bird i have.?**	Birds
RCS_TRLM	1st	**Were can i find out what kind of parrot i have?**	Birds
	5th	**How do you find out if abird is a girl or a boy?I don't know what kind of bird i have.?**	Birds
	6th	**Does anyone know what kind of bird this was?**	Other-Pets

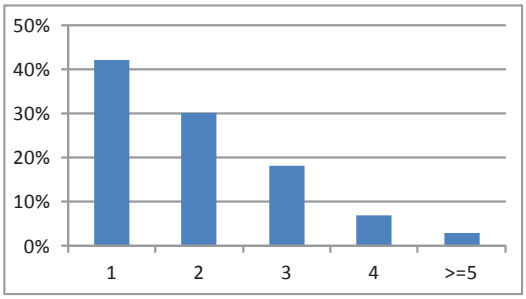

Figure 4: The proportion of the relevant questions relative to the number of the similar leaf categories.

Table 6: Comparison with Cao's work. All these comparisons are based on the same setting (a PC with 4G of memory and a 2.5Ghz CPU with Java programming language), where † indicates the statistical significance over the baseline LM + QC with $p < 0.05$, $*$ indicates the statistical significance over the baseline VSM + TRLM with $p < 0.08$.

#	Methods	Time (in Seconds)	MAP	P@10
1	LM + QC	0.071	0.408	0.247
2	**RCS_LM**	**0.064**	$\mathbf{0.453}^{\dagger}$	$\mathbf{0.267}^{\dagger}$
3	VSM + TRLM	0.445	0.456	0.269
4	**RCS_TRLM**	**0.083**	$\mathbf{0.482}^{*}$	$\mathbf{0.275}^{*}$

Example: To get a better understanding the effectiveness of our proposed method, Table 5 gives part of the results of an example queried question "How do i find out what kind of bird i have by looking on the internet?" that is originally in the category "Pets → Birds". The questions in bold are labeled as "relevant". After using the user chosen category, we can filter the irrelevant question under the category "Internet". Moreover, the ranks of the relevant questions under the specific category can be promoted (from 4th to 1st and 15th to 5th, respectively). Finally, when considering the relevant questions under the similar leaf categories (e.g., Other-Pets), some relevant questions under the the similar leaf categories can be found, thus the overall performance can be further improved. This example validates the effectiveness of our proposed method.

5.4.3 *Comparison with Cao's work*

We are aware of only two published studies [6] and [7] on utilizing category information for question search. In this subsection, we compare with two studies in term of computational cost of average running time for a search query and the performance of question search.

Cao et al. [6] employed classifiers to compute the probability of a queried question belonging to different categories, and then incorporated the classified categories into language model for question search. We denote this method as LM + QC shown in row 1 in Table 6. Cao et al. [7] computed the global relevance score with regard to the entire collection of questions, and then computed the local relevance with regard to each category of the historical questions. Cao et al. [7] introduced the different combinations to compute the global relevance and local relevance, the combination VSM + TRLM showed the superior performance than others. In this paper, we compare the proposed method with the combination VSM + TRLM shown in row 3 in Table 6 according to two measures MAP and P@10.

From Table 6, we can see that the proposed method is much faster and better than Cao et al. [6] (row 1 vs. row 2) and Cao et al. [7] (row 3 vs. row 4). To investigate why Cao et al. [6] fails to give the satisfactory results, we check the hierarchical classification model and some wrong examples. We find that the Micro F_1-score of the classifier is only about 45%, if the queried question is not correctly classified, then the retrieval results are poor since we search in a wrong category. For [7], the global relevance and local relevance are computed in a pipeline way, so it is difficult to employ the algorithm (e.g., parallel computing) to reduce the computational cost of running time. In this paper, we balance the effectiveness and the efficiency of question search by leveraging user chosen question category, which is simpler, faster and better than Cao's works.

Table 7: Effect of length-balanced translation pair by using two measures MAP and P@10.

Model	#	Methods	MAP	P@10
TRLM	1	LU	0.415	0.246
	2	**LB**	**0.430**	**0.257**
RCS_TRLM	3	LU	0.472	0.272
	4	**LB**	**0.482**	**0.275**

5.4.4 *Effect of Length-Balanced Translation Pair*

In this paper, question-answer pairs collected from Yahoo! Answers are used as a type of parallel corpus. IBM alignment tool assumes that each translation pair should be of comparable length. However, the answer part is usually much longer than the question part. It will hurt the performance if we fill the length-unbalanced pairs for learning the translation probabilities. In this subsection, we look into how much the length-balanced translation pair benefits the question routing. We introduce the baseline method (denoted as LU) to denote the translation probability with the length-unbalanced question-answer pairs, which is similar to [26]. Our proposed length-balanced question-answer pairs (denoted as LB) is also used for comparison.

Table 8: Sample word translation probabilities using the length-unbalanced pairs (left) and the length-balanced pairs (right). Note that words are stemmed.

$t =$ car	
w	$P(w\|t)$
insurance	1.0
pittman	0.855464
trabant	0.714575
shutter	0.456294
warhead	0.422596
jet	0.307796
gas	0.191602
drive	0.131121
car	0.0935975
buy	0.0891022

$t =$ car	
w	$P(w\|t)$
car	0.795577
insurance	0.648186
auto	0.472068
drive	0.307796
shutter	0.196918
speed	0.100494
pittman	0.0856549
buy	0.0627145
stop	0.0576182
mile	0.0452815

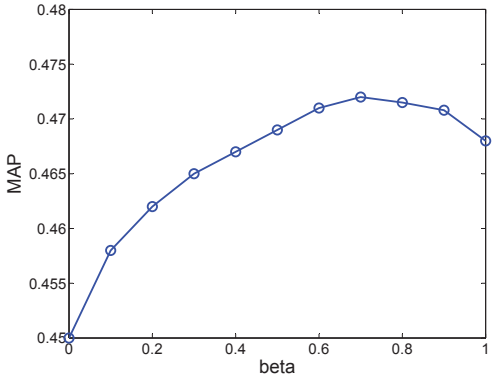

Figure 5: The effect of translation directions with length-balanced translation pair for question search by using the measure MAP.

Table 7 shows the comparison. From this Table, we see that using the length-balanced question-answer pairs significantly outperforms the method using the length-unbalanced translation pairs (row 1 vs. row 2; row 3 vs. row 4). Significant test using t-test show the difference between the two methods are statistically significant ($p < 0.05$).

To better understand the length-balanced and the length-unbalanced pairs, Table 8 shows a historical question word together with ten most probable queried question words that it will translate to by both LU and LB methods. The table shows that the related words for word "car" in case of the length-balanced question-answer pairs are more reasonable and specific than for words learned via the length-unbalanced pairs.

5.4.5 Effect of Translation Directions

The harmonic factor β controls the weight of the translation models trained in two directions, e.g., $P_{a2q}(t|w)$ and $P_{q2a}(t|w)$ as shown in equation (22). In this subsection, we look into the effect of the harmonic factor β for the performance.

In Figure 5, we show the MAP curve of the RCS_TRLM with the length-balanced question-answer pairs for question search when harmonic factor β ranges from 0.0 to 1.0 stepped by 0.1. From the figure, we observe that the best performance is obtained when $\beta = 0.7$, which indicates that "answer $\rightarrow$ question" is more important than "question $\rightarrow$ answer".

6. CONCLUSIONS AND FUTURE WORK

In this paper, we propose a faster and better retrieval model for question search by leveraging user chosen question category. The proposed method not only considers the relevant questions under the same leaf categories, but also considers the relevant questions under the similar leaf categories with the categories of queried questions. Experimental results conducted on large-scale real cQA data set demonstrate that the proposed techniques are more effective and efficient than a variety of baseline methods. To the best our knowledge, it is the first work to extensively address both the effectiveness and the efficiency of question search in cQA.

This work opens to several interesting directions for future work. First, it is necessary to include the related category-sensitive information into other studies (e.g., learning-to-rank techniques [2, 4, 18, 19], analogical reasoning-based approach [23], syntactic structures of questions [22], and phrase-based SMT [20, 28]) for question search. Second, question structures should be considered, so it would be interesting to combine the proposed method with other question search methods (e.g., Duan et al. [9]) to further improve the performance.

7. ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (No. 61303180, No. 61070106, No. 61272332 and No. 61202329), the National High Technology Development 863 Program of China (No. 2012AA011102), the National Basic Research Program of China (No. 2012CB316300), CCF Opening Project of Chinese Information Processing, and also Sponsored by CCF-Tencent Open Research Fund. We thank the anonymous reviewers for their insightful comments.

8. REFERENCES

[1] D. Bernhard and I. Gurevych. 2009. Combining lexical semantic resources with question & answer archives for translation-based answer finding. In *ACL*, pages 728-736.

[2] J. Bian, Y. Liu, E. Agichtein, and H. Zha. 2008. Finding the right facts in the crowd: factoid question answering over social media. In *WWW*, pages 467-476.

[3] D. Blei, A. Ng, and M. Jordan. 2003. Latent dirichlet allocation. Journal of Machine Learning Research, 3:993-1022.

[4] R. Bunescu and Y. Huang. 2010. Learning the relative usefulness of questions in community QA. In *EMNLP*.

[5] L. Cao, G. Zhou, K. Liu, and J. Zhao. 2011. Learning the latent topics for question retrieval in community QA. In *IJCNLP*.

[6] X. Cao, G. Cong, B. Cui, C. Jensen, and C. Zhang. 2009. The use of categorization information in language models for question retrieval. In *CIKM*, pages 265-274.

[7] X. Cao, G. Cong, B. Cui, and C. Jensen. 2010. A generalized framework of exploring category information for question retrieval in community question answer archives. In *WWW*.

[8] A. P. Dempster, N. M. Laird, D. B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1-38.

[9] H. Duan, Y. Cao, C. Lin, and Y. Yu. 2008. Searching questions by identifying questions topics and question focus. In *ACL*, pages 156-164.

[10] T. Griffiths and M. Steyvers. 2004. Finding scientific topics. The National Academy of Sciences, 101:5228-5235.

[11] J. Jeon, W. Croft, and J. Lee. 2005. Finding similar questions in large question and answer archives. In *CIKM*, pages 84-90.

[12] S. Kullback and R. A. Leibler. 1951. On information and sufficiency. Annals of Mathematical Statistics 22 (1): 79-86.

[13] J. Lee, S. Kim, Y. Song, and H. Rim. 2008. Bridging lexical gaps between queries and questions on large online Q&A collections with compact translation models. In *EMNLP*.

[14] B. Li, I. King, and M. Lyu. 2011. Question routing in community question answering: putting category in its place. In *CIKM*.

[15] C. Manning, P. Raghavan, and H. Schtze. 2008. Introduction to information retrieval. *Cambridge University Press*, New York, NY, USA.

[16] S. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. 1994. Okapi at trec-3. In *TREC*, pages 109-126.

[17] G. Salton, A. Wong, and C. S. Yang. 1975. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613-620.

[18] Y. -I. Song, C. -Y. Lin, Y. Cao, and H. -C. Rim. 2008. Question utility: a novel static ranking of question search. In *AAAI*, pages 1231-1236.

[19] M. Surdeanu, M. Ciaramita, and H. Zaragoza. 2008. Learning to rank answers on large online qa collections. In *ACL*, pages 719-727.

[20] S. Riezler, A. Vasserman, I. Tsochantaridis, V. Mittal, and Y. Liu. 2007. Statistical machine translation for query expansion in answer retrieval. In *ACL*, pages 464-471.

[21] A. Singh. 2012. Entity based q&a retrieval. In *EMNLP-CoNLL*, pages 1266-1277.

[22] K. Wang, Z. Ming, and T-S. Chua. 2009. A syntactic tree matching approach to finding similar questions in community-based qa services. In *SIGIR*, pages 187-194.

[23] X. Wang, X. Tu, D. Feng, and L. Zhang. 2009. Ranking community answers by modeling question-answer relationships via analogical reasoning. In *SIGIR*.

[24] B. Wang, X. Wang, C. Sun, B. Liu, and L. Sun. 2010. Modeling semantic relevance for question-answer pairs in web social communities. In *ACL*, pages 1230-1238.

[25] G. Xue, D. Xing, Q. Yang, and Y. Yu. 2008. Deep classification in large-scale text hierarchies. In *SIGIR*.

[26] X. Xue, J. Jeon, and W. Croft. 2008. Retrieval models for question and answer archives. In *SIGIR*, pages 475-482.

[27] C. Zhai and J. Lafferty. 2001. A study of smooth methods for language models applied to ad hoc information retrieval. In *SIGIR*, pages 334-342.

[28] G. Zhou, L. Cai, J. Zhao, and K. Liu. 2011. Phrase-based translation model for question retrieval in community question answer archives. In *ACL*, pages 653-662.

[29] G. Zhou, K. Liu, and J. Zhao. 2012. Exploiting bilingual translation for question retrieval in community-based question answering. In *COLING*, pages 3153-3170.

[30] G. Zhou, F. Liu, Y. Liu, S. He, and J. Zhao. 2013. Statistical Machine Translation Improves Question Retrieval in Community Question Answering via Matrix Factorization. In *ACL*, pages 852-861.