# A Generalized Framework of Exploring Category Information for Question Retrieval in Community Question Answer Archives

Xin Cao[1], Gao Cong[1,2], Bin Cui[3], Christian S. Jensen[1]
[1]Department of Computer Science, Aalborg University, Denmark
{xcao, gaocong, csj}@cs.aau.dk
[2]School of Computer Engineering, Nanyang Technological University, Singapore
[3]Dept. of Computer Science & Key Lab of High Confidence Software Technologies (MOE), Peking University, China
{bin.cui}@pku.edu.cn

## ABSTRACT

Community Question Answering (CQA) has emerged as a popular type of service where users ask and answer questions and access historical question-answer pairs. CQA archives contain very large volumes of questions organized into a hierarchy of categories. As an essential function of CQA services, question retrieval in a CQA archive aims to retrieve historical question-answer pairs that are relevant to a query question. In this paper, we present a new approach to exploiting category information of questions for improving the performance of question retrieval, and we apply the approach to existing question retrieval models, including a state-of-the-art question retrieval model. Experiments conducted on real CQA data demonstrate that the proposed techniques are capable of outperforming a variety of baseline methods significantly.

## 1. INTRODUCTION

Community Question Answering (CQA) services are Internet services that enable users to ask and answer questions, as well as to search through historical question-answer pairs. Examples of such community-driven knowledge services include Yahoo! Answers (answers.yahoo.com), Naver (www.naver.com), Baidu Zhidao (zhidao.baidu.com), and WikiAnswers (wiki.answers.com).

CQA services can directly return answers to the query questions of users, instead of a long list of ranked URLs, thus providing an effective alterative to web search [1, 19]. Question retrieval can also be considered as an alternative solution to the traditional TREC Question Answering (QA) task. In TREC QA, the research has largely focused on extracting concise answers to questions of a limited type, e.g., factoid questions. In contrast, the answers to questions in a CQA archive are generated by users, and these real-world questions are usually more complex and often presented in an informal way. Hence, the TREC QA task of extracting a correct answer is transformed to a Question retrieval task [18, 19].

Although CQA services emerged only recently[1], they have built up very large archives of historical questions and their answers. Retrieving relevant historical questions that best match a user's query question is an essential component of a CQA service. Additionally, when a user asks a new question in a CQA service, the service would typically search automatically for historical question-answer pairs that match the new question. If good matches are found, the lag time incurred by waiting for a human to respond can be avoided, thus improving user satisfaction.

Work on question retrieval in CQA data [3, 7, 9, 10, 19], employs different retrieval models, including the vector space model [7, 9, 10], the language model [7, 10], the Okapi model [10], and the translation model [2, 10, 14, 19]. The experimental studies in previous work consistently show that the translation model usually achieves the best performance [10, 19], or is able to improve other methods [7]. Other recent proposals for question retrieval include the use of learning-to-rank techniques [3], and syntactic structures of questions [18]. However, it is not clear of whether they are competitive with other methods, such as the language model and the translation model, due to a lack of empirical comparisons.

A distinctive feature of question-answer archives in CAQ services is that CQA services usually organize questions into a hierarchy of categories. For example, the subcategory "Health.Dental" is a child category of "Health" in Yahoo! Answers. When a user asks a question, the user is typically required to choose a category label for the question from a predefined hierarchy of categories. Hence, each question in a CQA archive has a category label. The questions in the same category or subcategory usually relate to the same general topic. For example, the questions in the subcategory "Travel.Europe.Denmark" mainly relate to travel to or in the country of Denmark.

The utility of category information in question retrieval can be exemplified by the following query question ($\mathbf{q}$): "can you recommend sightseeing opportunities for senior citizens in Boston?" The user is interested in sightseeing specifically in Boston, not in other places. Hence, the question ($\mathbf{d}$) "can you recommend sightseeing opportunities for senior citizens in China?" is not relevant to the user's question ($\mathbf{q}$), although the two questions are syntactically very similar. If one could establish a connection between $\mathbf{q}$ and the category "Travel.United States.Boston," the ranking of questions in that category could be promoted, thus perhaps improving the question retrieval performance.

Although it appears natural to attempt to exploit the available category information for question retrieval, we are aware of only one published study [5] on utilizing category information for question retrieval. That study considers two approaches to using categories for enhancing the performance of language-model based question retrieval. The first approach employs classifiers to compute the probability of a query question belonging to different categories; this probability is then used to adjust the ranking returned

[1]E.g., Yahoo! Answers was launched in December 2005.

Copyright is held by the International World Wide Web Conference Committee (IW3C2). Distribution of these papers is limited to classroom use, and personal use by others.
*WWW 2010*, April 26–30, 2010, Raleigh, North Carolina, USA.
ACM 978-1-60558-799-8/10/04.

by the language model. However, the experimental results [5] show that it can improve the performance of the language model only slightly.

The second approach is based on a two-level smoothing model. A category language model is computed and then smoothed with the whole question collection. The question language model is subsequently smoothed with the category model. The study offers a theoretical justification for the approach, which is also shown to significantly improve the performance of the language model for question retrieval. However, this second approach of utilizing category information is tightly coupled with the language model and is not applicable to other question retrieval models.

The empirical studies [5] provide evidence that category information can be utilized to improve the performance of the language model for question retrieval. This suggests an open problem: how ca category information be utilized effectively in the context of other popular question retrieval models. In this paper, we present a new and general approach to exploiting category information that can be applied to any question retrieval model.

The proposed method is called the category enhanced retrieval model. The intuition behind is that the more related a category is to a query $\mathbf{q}$, the more likely it is that the category contains questions relevant to $\mathbf{q}$. The model ranks a historical question $\mathbf{d}$ based on an interpolation of two relevance scores: the one is a global relevance score between query $\mathbf{q}$ and the *category* containing $\mathbf{d}$, and the other is a local relevance score between query $\mathbf{q}$ and question $\mathbf{d}$.

A subtle feature of this approach is that the words play different roles in computing global relevance and local relevance. For example, in the example query $\mathbf{q}$, the word "Boston" is discriminative when computing the global relevance scores across categories because it makes the category "Travel.United States.Boston" highly relevant to the query; however, it is not effective at distinguishing one question from another within "Travel.United States.Boston," since many questions in that category contain this word. The global relevance score is computed with regard to the entire collection of questions while the local relevance score is computed with regard to the category of a historical question.

The approach can be combined with any existing question retrieval approach. We combine it with the vector space model [10, 11], the language model [10], the Okapi model [10], the translation model [2, 10, 14], and the translation-based language model [19], one of the state-of-the-art methods.

The paper's contributions are twofold. First, we propose a simple yet effective approach to utilizing category information to enhance the performance of question retrieval. This approach combines the global relevance (the relevance of a query to a category) and the local relevance (the relevance of a query to a question in the category). We also exploit several methods to compute global and local relevance. The approach combines well with existing question retrieval methods since it can use any question retrieval method to compute the local relevance.

Second, we conduct extensive experiments on a data set extracted from Yahoo! Answers consisting of more than 3 million questions. We find that the proposed technique using category information is capable of significantly improving existing question retrieval models, including one of the state-of-the-art models [19]. We also empirically evaluate the performance of the alternative methods of computing global relevance. Finally, we compare with the two existing approaches [5] to exploiting category information.

The paper is organized as follows. Section 2 covers existing question retrieval models. Section 3 details the proposed techniques. Section 4 covers the experimental study. Section 5 reviews related work. Section 6 concludes and offers research directions.

# 2. PRELIMINARIES

We cover the retrieval models that we use for question retrieval.

## 2.1 Vector Space Model

The Vector Space Model has been used widely in question retrieval [10, 11]. We consider a popular variation of this model [21]: Given a query $\mathbf{q}$ and a question $\mathbf{d}$, the ranking score $S_{\mathbf{q},\mathbf{d}}$ of the question $\mathbf{d}$ can be computed as follows:

$$\begin{aligned} S_{\mathbf{q},\mathbf{d}} &= \frac{\sum_{t\in\mathbf{q}\bigcap\mathbf{d}} w_{\mathbf{q},t}w_{\mathbf{d},t}}{W_{\mathbf{q}}W_{\mathbf{d}}}, \text{ where} \\ w_{\mathbf{q},t} &= \ln(1+\frac{N}{f_t}),\ w_{\mathbf{d},t} = 1+\ln(tf_{t,\mathbf{d}}) \\ W_{\mathbf{q}} &= \sqrt{\sum_t w_{\mathbf{q},t}^2},\ W_{\mathbf{d}} = \sqrt{\sum_t w_{\mathbf{d},t}^2} \end{aligned} \tag{1}$$

Here $N$ is the number of questions in the whole collection, $f_t$ is the number of questions containing the term $t$, and $tf_{t,\mathbf{d}}$ is the frequency of term $t$ in $\mathbf{d}$. The term $W_{\mathbf{q}}$ can be neglected as it is a constant for a given query and does not affect the rankings of historical questions. Note that $w_{\mathbf{q},t}$ captures the IDF (inverse document frequency) of term $t$ in the collection, and $w_{\mathbf{d},t}$ captures the TF (term frequency) of term $t$ in $\mathbf{d}$.

## 2.2 Okapi BM25 Model

While the Vector Space Model favors short questions, the Okapi BM25 Model [15] takes into account the question length to overcome this problem. The Okapi Model is used for question retrieval by Jeon et al. [10]. Given a query $\mathbf{q}$ and a question $\mathbf{d}$, the ranking score $S_{\mathbf{q},\mathbf{d}}$ is computed as follows:

$$\begin{aligned} S_{\mathbf{q},\mathbf{d}} &= \sum_{t\in\mathbf{q}\bigcap\mathbf{d}} w_{\mathbf{q},t}w_{\mathbf{d},t}, \text{ where} \\ w_{\mathbf{q},t} &= \ln(\frac{N-f_t+0.5}{f_t+0.5})\frac{(k_3+1)tf_{t,\mathbf{q}}}{k_3+tf_{t,\mathbf{q}}} \\ w_{\mathbf{d},t} &= \frac{(k_1+1)tf_{t,\mathbf{d}}}{K_{\mathbf{d}}+tf_{t,\mathbf{d}}} \\ K_{\mathbf{d}} &= k_1((1-b)+b\frac{W_{\mathbf{d}}}{W_A}) \end{aligned} \tag{2}$$

Here $N$ is the number of questions in the collection; $f_t$ is the number of questions containing the term $t$; $tf_{t,\mathbf{d}}$ is the frequency of term $t$ in $\mathbf{d}$; $k_1$, $b$, and $k_3$ are parameters that are set to 1.2, 0.75, and $\infty$, respectively, thus following Robertson et al. [15] (the expression $\frac{(k_3+1)tf_{t,\mathbf{q}}}{k_3+tf_{t,\mathbf{q}}}$ is then equivalent to $tf_{t,\mathbf{q}}$ ); and $W_{\mathbf{d}}$ is the question length of $\mathbf{d}$ and $W_A$ is the average question length in the collection.

## 2.3 Language Model

The Language Model is used in previous work [5, 7, 10] for question retrieval. The basic idea of the Language Model is to estimate a language model for each question, and then rank questions by the likelihood of the query according to the estimated model for questions. We use Jelinek-Mercer smoothing [20]. Given a query $\mathbf{q}$ and a question $\mathbf{d}$, the ranking score $S_{\mathbf{q},\mathbf{d}}$ is computed as follows:

$$\begin{aligned} S_{\mathbf{q},\mathbf{d}} &= \prod_{t\in\mathbf{q}}((1-\lambda)P_{ml}(t|\mathbf{d})+\lambda P_{ml}(t|\mathbf{Coll})), \text{ where} \\ P_{ml}(t|\mathbf{d}) &= \frac{tf_{t,\mathbf{d}}}{\sum_{t'\in\mathbf{d}} tf_{t',\mathbf{d}}} \\ P_{ml}(t|\mathbf{Coll}) &= \frac{tf_{t,\mathbf{Coll}}}{\sum_{t'\in\mathbf{Coll}} tf_{t',\mathbf{Coll}}} \end{aligned} \tag{3}$$

Here $P_{ml}(t|\mathbf{d})$ is the maximum likelihood estimate of word $t$ in $\mathbf{d}$; $P_{ml}(t|\mathbf{Coll})$ is the maximum likelihood estimate of word $t$ in the collection $\mathbf{Coll}$; and $\lambda$ is the smoothing parameter.

## 2.4 Translation Model

Previous work [2,7,11,14,19] consistently reports that the Translation Model yields superior performance for question retrieval. The Translation Model exploits word translation probabilities in the language modeling framework. We follow the translation model approach of Jeon et al. [10]. Given a query $\mathbf{q}$ and a question $\mathbf{d}$, the ranking score $S_{\mathbf{q},\mathbf{d}}$ is computed as follows:

$$S_{\mathbf{q},\mathbf{d}} = \prod_{t \in \mathbf{q}} ((1-\lambda) \sum_{w \in \mathbf{d}} T(t|w) P_{ml}(w|\mathbf{d}) + \lambda P_{ml}(t|\mathbf{Coll})), \tag{4}$$

where $P_{ml}(w|\mathbf{d})$ and $P_{ml}(t|\mathbf{Coll})$ can be computed similarly as in Equation 3 for the Language Model and $T(t|w)$ denotes the probability that word $w$ is the translation of word $t$. Jeon et al. [10] assume that the probability of self-translation is 1, meaning that $T(w|w) = 1$.

## 2.5 Translation-Based Language Model

A recent approach [19] to question retrieval, denoted as TRLM, combines the Language Model and the Translation Model. It is shown [19] that this model gains better performance than both the Language Model and the Translation Model. Given a query $\mathbf{q}$ and a question $\mathbf{d}$, the ranking score $S_{\mathbf{q},\mathbf{d}}$ is computed as follows:

$$\begin{aligned} S_{\mathbf{q},\mathbf{d}} = \prod_{t \in \mathbf{q}} ((1-\lambda)(\beta \sum_{w \in \mathbf{d}} T(t|w) P_{ml}(w|\mathbf{d}) \\ + (1-\beta) P_{ml}(t|\mathbf{d})) + \lambda P_{ml}(t|\mathbf{Coll})), \end{aligned} \tag{5}$$

where parameter $\beta$ controls the translation component's impact.

# 3. EXPLOITING CATEGORIES IN QUESTION RETRIEVAL

Each question in Yahoo! Answers belongs to a unique leaf category. We present an approach to exploiting such category information to enhance question retrieval. This approach can be easily applied to existing question retrieval models.

## 3.1 Category Enhanced Retrieval Model

Intuitively, the more related a category is to a query $\mathbf{q}$, the more likely it is that the category contains questions relevant to $\mathbf{q}$. Hence, we incorporate the relevance of the query to the category of a historical question into the question retrieval model. The idea is to compute the ranking score of a historical question $\mathbf{d}$ based on an interpolation of two relevance scores: one is a global relevance score between query $\mathbf{q}$ and the category $cat(\mathbf{d})$ that contains $\mathbf{d}$, and the other is a local relevance score between $\mathbf{q}$ and $\mathbf{d}$ considering only $cat(\mathbf{d})$.

Various retrieval models can be used to compute the two scores, and the two scores may have very different domain ranges. We therefore normalize them into the same range before we employ a linear interpolation to combine them. The final ranking score $RS_{\mathbf{q},\mathbf{d}}$ is then computed as follows:

$$RS_{\mathbf{q},\mathbf{d}} = (1-\alpha) N(S_{\mathbf{q},\mathbf{d}}) + \alpha N(S_{\mathbf{q},cat(\mathbf{d})}), \tag{6}$$

where $S_{\mathbf{q},\mathbf{d}}$ is the local relevance score, $cat(\mathbf{d})$ represents the category containing question $\mathbf{d}$, $S_{\mathbf{q},cat(\mathbf{d})}$ is the global relevance score, $\alpha$ controls the linear interpolation, and $N()$ is the normalization function for the local and global relevance scores (discuss in Section 4.1).

The intuition behind the approach is simple: the global relevance is employed to differentiate the relevancies of different categories to the query, while the local relevance is mainly used to differentiate the relevancies of questions within the same category to the query.

However, subtle issues exist when applying the existing question retrieval methods to compute the global and local relevance scores, one of which is considered next. As the global relevance can distinguish questions across categories, we would like the local relevance to effectively rank questions within a specific category. We observe that some words may be important the global ranking of questions across categories, while they are unimportant when ranking questions within a specific category. For example, the word "Denmark" is important for distinguishing questions in the category "Travel.Europe.Denmark" from questions in other categories; however it is of little importance when ranking questions within the category "Travel.Europe.Denmark," since it appears in many questions in this category. We thus compute the local relevance with regard to a category, rather than the whole collection, since the importance of a word in a category is more meaningful than in the whole collection when ranking questions within a specific category.

Various retrieval models can be adopted to compute the local and global relevance scores, and then the final ranking score can be computed using Equation 6. We proceed to detail how to compute the global and local relevance scores using different retrieval models.

## 3.2 Retrieval Models—Global Relevance

Historical CQA questions are categorized, and the global relevance reveals the relationship between the questions in a category as a whole and a query. Assigning relevance scores to categories makes it possible to favor questions in relevant categories. We proceed to present how to compute the global relevance scores using the models introduced in Section 2.

### *3.2.1 Vector Space Model*

The problem here is how to adapt the Vector Space Model (VSM) to compute the relevance score between a question and a category. We develop two methods.

In the first, we view a category as a concatenation of the questions it contains, i.e., the category is represented by a single pseudo document. Thus, the whole question collection can be viewed as a collection of pseudo documents, each of which represents a category. The TF (term frequency) and IDF (inverse document frequency) need to be modified correspondingly. The TF of a word is computed with regard to a certain category, and the IDF is actually the "inverse category frequency," which is in inversely proportional to the number of categories containing the word.

A straightforward method is to compute the cosine similarity between a query question and the document representing a category as the global relevance. However, this straightforward method will yield poor performance for two reasons. First, the the category document length varies significantly from category to category. For example, on the data set used in our experiments, the length varies from 467 to 69,789. Second, the query question length is much shorter than the category document length, and a query and a category will share only very few words. If we compute the similarity according to Equation 1, the value will be dominated by the normalization value. Thus, categories with few questions will tend to have high global relevance scores.

Instead, we use the category document length to adjust the term frequency of a word, and only the query vector is used for normalization. More formally, given a query $\mathbf{q}$ and a question $\mathbf{d}$ with

category $cat(\mathbf{d})$, the global relevance $S_{\mathbf{q},cat(\mathbf{d})}$ in Equation 6 is computed as follows:

$$S_{\mathbf{q},cat(\mathbf{d})} = \frac{\sum_{t\in\mathbf{q}\cap cat(\mathbf{d})} w_{\mathbf{q},t} w_{cat(\mathbf{d}),t}}{W_{\mathbf{q}}}, \text{ where}$$
$$w_{\mathbf{q},t} = \ln(1+\frac{M}{fc_t}), w_{cat(\mathbf{d}),t} = 1 + \frac{1}{\ln(\frac{W_{cat(\mathbf{d})}}{tf_{t,cat(\mathbf{d})}})}$$

Here $M$ is the total number of leaf categories, $fc_t$ is the number of categories that contain the term $t$, $tf_{(t,cat(\mathbf{d}))}$ is the frequency of $t$ in the category $cat(\mathbf{d})$, $W_{cat(\mathbf{d})}$ is the length of $cat(\mathbf{d})$ (number of words contained in $cat(\mathbf{d})$), and $w_{\mathbf{q},t}$ captures the IDF of word $t$ with regard to categories.

We can see that the relevance score is only normalized by $W_{\mathbf{q}}$ and the length of a category is used when computing the TF for word $t$.

We note that the existing pivoted document length normalization method [16] does not normalize the documents to unit length and that the normalized score is skewed to account for the effect of document length on relevance. However, this method needs training data to learn the parameter of skewness. We do not have the training data and do not employ this method.

The second method for computing the similarity between a query question and a category employs a centroid vector of a category. The global relevance score is then computed as the cosine similarity between the query vector and the centroid vector. Specifically, the centroid vector of $cat(\mathbf{d})$, the category containing $\mathbf{d}$, is computed as:

$$\vec{c} = \frac{1}{|cat(\mathbf{d})|} \sum_{d_i \in cat(\mathbf{d})} \vec{v_{d_i}},$$

where $\vec{v_{d_i}}$ is the vector of a historic question $d_i$ contained in $cat(\mathbf{d})$, and each element in the vector is computed by $w_{\mathbf{d},t}$ in Equation 1.

Our experiments show that this method performs worse than the first method. A possible explanation is that this method violates the term frequency saturation heuristic required for effective retrieval [8]. This heuristic says that the score contribution from matching a term should grow sub-linearly as the frequency of the matched term increases. The intuition is that the information gain from the first occurrences of the term is higher than for subsequent occurrences (e.g., "the change in the score caused by increasing TF from 1 to 2 is larger than that caused by increasing TF from 100 to 101" [8]). When terms are weighted in each of the constituent questions of a category and then combined to form a representation for the category, a term occurring in multiple questions of a category would be weighted much higher than it would be if we were to weight it after aggregating over all the terms of the category (i.e., as in the first method). In other words, this method violates the heuristic since repeated occurrences of the same term are counted as "fresh occurrences."

### 3.2.2 Okapi BM25 Model

The centroid representation of a category performs worse than the question concatenation representation in the Vector Space Model. The reasons for this also apply to the Okapi Model. Hence in the Okapi Model, we only consider concatenating the questions in a category into a pseudo document. As in the Vector Space Model, the TF and IDF also need to be modified correspondingly. They are computed with regard to categories. In addition, this model considers the lengths of documents—these lengths are modified to be the lengths of categories.

According to Equation 2, the global relevance score $S_{\mathbf{q},cat(\mathbf{d})}$ can be computed as follows:

$$\begin{aligned} S_{\mathbf{q},cat(\mathbf{d})} &= \sum_{t\subset\mathbf{q}\bigcap cat(\mathbf{d})} w_{\mathbf{q},t} w_{cat(\mathbf{d}),t}, \text{ where} \\ w_{\mathbf{q},t} &= \ln(\frac{M-fc_t+0.5}{fc_t+0.5})\frac{(k_3+1)tf_{t,\mathbf{q}}}{k_3+tf_{t,\mathbf{q}}} \\ w_{cat(\mathbf{d}),t} &= \frac{(k_1+1)tf_{t,cat(\mathbf{d})}}{K_{\mathbf{d}}+tf_{t,cat(\mathbf{d})}} \\ K_{\mathbf{d}} &= k_1((1-b)+b\frac{W_{cat(\mathbf{d})}}{W_{A(cat)}}) \end{aligned} \quad (7)$$

Here $M$ is the number of categories, $fc_t$ is the number of categories that contain the term $t$, $W_{cat(\mathbf{d})}$ is the length of the concatenated category document $cat(\mathbf{d})$, and $W_{A(cat)}$ is the average length of the concatenated category documents.

### 3.2.3 Language Model, Translation Model, and Translation-Based Language Model

Since categories are viewed as pseudo documents, in these three models, we just need to replace $\mathbf{d}$ with $cat(\mathbf{d})$ to compute the global relevance scores using Equations 3, 4 and 5.

## 3.3 Retrieval Models—Local Relevance

Each model introduced in Section 2 can be employed without modification for computing the local relevance scores between a query question and a historical question. However, as the global relevance score captures the relevance of a category to a query, the local relevance score should mainly capture the relevance of a question within a category to a query. To achieve this, we adapt the models to compute more effective local relevance scores. The idea is to compute the local relevance scores treating a category as the collection.

### 3.3.1 Vector Space Model

As discussed briefly in Section 3.1, computing the local relevance score with regard to only the category containing the historical question appears to be preferable. This observation motivates us to compute the local relevance differently from the standard VSM.

Specifically, we compute the IDF corresponding to the category, and we call it local IDF. In the standard, or baseline, model the IDF is computed on the whole collection, and we call it global IDF. The local IDF reveals the importance of a word in a query better than does the global IDF when ranking the historical questions within the same category.

The local relevance is computed by Equation 1 with the following modifications: $N$ is replaced by $N_{cat(\mathbf{d})}$, the number of questions in the category $cat(\mathbf{d})$; $f_t$ is replaced by the number of questions in category $cat(\mathbf{d})$ containing term $t$, denoted as $f_{t,cat(\mathbf{d})}$. We modify $w_{\mathbf{q},t}$ as follows:

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

### 3.3.2 Okapi BM25 Model

Similar to the Vector Space Model, the Okapi Model also has an IDF component, i.e., $w_{\mathbf{q},t}$ in Equation 2. We also compute the IDF with regard to the category containing the historical question, which we call local IDF.

The local relevance is computed using Equation 2 with the following modifications: $N$ is replaced with $N_{cat(\mathbf{d})}$; and $f_t$ and $W_A$ are computed with regard to the category $cat(\mathbf{d})$ and are replaced

by $f_{t,cat(\mathbf{d})}$ and $W_{A,cat(\mathbf{d})}$, respectively, where $f_{t,cat(\mathbf{d})}$ is the document frequency of $t$ in category $cat(\mathbf{d})$, and $W_{A,cat(\mathbf{d})}$ is the average length of questions in the category $cat(\mathbf{d})$. Specifically, compared to Equation 2, the following modifications are made:

$$w_{\mathbf{q},t} = \ln(\frac{N_{cat(\mathbf{d})} - f_{t,cat(\mathbf{d})} + 0.5}{f_{t,cat(\mathbf{d})} + 0.5}) \frac{(k_3+1)tf_{t,\mathbf{q}}}{k_3 + tf_{t,\mathbf{q}}}$$

$$K_{\mathbf{d}} = k_1((1-b) + b\frac{W_{\mathbf{d}}}{W_{A,cat(\mathbf{d})}})$$

#### 3.3.3 *Language Model, Translation Model, and Translation-Based Language Model*

Unlike the Vector Space Model and the Okapi Model, the Language Model does not have an IDF component. However, the smoothing in the Language Model plays an IDF-like role [20]. Similarly, the smoothing in the Translation Model and in the Translation-Based Language Model also plays an IDF-like role. Thus, we compute the smoothing value with regard to the category rather than the whole collection, i.e., we use $P_{ml}(t|cat(\mathbf{d}))$ to do the smoothing instead of $P_{ml}(t|\mathbf{Coll})$ when computing the local relevance scores in Equations 3, 4, and 5.

Note that when compared to an existing approach [5], the category smoothing is used differently and for a different purpose. The existing approach uses the category smoothing together with standard collection smoothing to distinguish questions from both the same category and different categories. In contrast, in local relevance scoring, smoothing is used alone, to distinguish questions within the same category.

### 3.4 Learning Word Translation Probabilities

The performance of the Translation Model and the Translation-Based Language Model will rely on the quality of the word-to-word translation probabilities. We follow the approach of Xue et al. [19] to learn the word translation probabilities. In our experiments, question-description pairs are used for training, and the GIZA++[2] toolkit is used to learn the IBM translation model 1. There are two ways of accomplishing the training: the word translation probability can be calculated through setting the questions (resp. the descriptions) as the source and the descriptions (resp. the questions) as the target.

We employ $P(D|Q)$ to denote the word-to-word translation probability with the question $Q$ as the source and the description $D$ as the target, and $P(Q|D)$ denotes the opposite configuration. A simple method is to linearly combine the two translation probabilities for a source word and a target word as the final translation probability. Xue et al. find that a better method is to combine the question-description pairs used for training $P(D|Q)$ with the description-question pairs used for training $P(Q|D)$, and to then use this combined set of pairs for learning the word-to-word translation probabilities.

## 4. EVALUATION

We study the abilities of the proposed technique to improve the performance of question retrieval models.

### 4.1 Experimental Setup

**Question Search Methods:** We denote the five question retrieval models from Section 2 as VSM (Vector Space Model), Okapi (Okapi BM25 Model), LM (Language Model), TR (Translation Model). and TRLM (Translation-Based Language Model). They are used as baseline methods. In our category-enhanced model (denoted as CE, Section 3.1), these five models can be used to compute both the global relevance score and the local relevance score. Thus, we obtain 25 combinations.

Each combination is denoted by the name of the model used for computing the global relevance score plus the name of the model used for computing the local relevance score. For example, if the global relevance score is computed by the Language Model and the local relevance score is computed by the Vector Space Model, we denote the resulting model as LM+VSM.

As a reference, we also give the performance of the method searching in the category of the query, denoted as OptC (the query set is collected from the Yahoo! Answers, so the categories of queries are known).

We also compare with two existing methods [5]. The first utilizes question classification to compute the probabilities of a query belonging to the categories. We extend this method to all baseline models used in this paper and denoted this as QC. The second method is coupled with the Language Model, and we call it LM+L [5].

| Category | Number of Questions |
|---|---|
| Arts & Humanities | 114737 |
| Beauty & Style | 49532 |
| Business & Finance | 154714 |
| Cars & Transportation | 208363 |
| Computers & Internet | 129472 |
| Consumer Electronics | 126253 |
| Dining Out | 58980 |
| Education & Reference | 107337 |
| Entertainment & Music | 196100 |
| Environment | 28476 |
| Family & Relationships | 53687 |
| Food & Drink | 55955 |
| Games & Recreation | 72634 |
| Health | 183181 |
| Home & Garden | 50773 |
| Local Businesses | 69581 |
| News & Events | 27884 |
| Pets | 72265 |
| Politics & Government | 85392 |
| Pregnancy & Parenting | 63228 |
| Science & Mathematics | 116047 |
| Social Science | 61011 |
| Society & Culture | 122358 |
| Sports | 275893 |
| Travel | 403926 |
| Yahoo! Products | 228368 |

**Table 6: Number of questions in each first-level category**

**Data Set:** We collect questions from all categories in Yahoo! Answers. We use the getByCategory function from the Yahoo! Answers API[3] to obtain QA threads from the Yahoo! site. The resulting *question repository* that we use for question retrieval contains 3,116,147 questions. We use 1 million question-description pairs from another data set[4] for training the word translation probabilities.

Note that the question repository is much larger than those used in previous work on question retrieval; a large real data set is expected to better reflect real application scenarios of CQA services. The *question repository* is also used as the training set for query classification. There are 26 categories at the first level and 1263 categories at the leaf level. Each question belongs to a unique leaf category. Table 6 shows the distribution across first-level categories

[2]http://www.fjoch.com/GIZA++.html

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

[4]The Yahoo! Webscope dataset ydata-ymusic-user-artist-ratings-v1_0, available at http://research.yahoo.com/Academic_Relations.

| | VSM | OptC | QC | VSM+VSM | %chg | Okapi+VSM | %chg | LM+VSM | %chg | TR+VSM | %chg | TRLM+VSM | %chg |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MAP | 0.2407 | 0.2414 | 0.2779 | **0.3711** | 54.2* | 0.3299 | 37.1* | 0.3632 | 50.9* | 0.3629 | 50.8* | 0.3628 | 50.7* |
| MRR | 0.4453 | 0.4534 | 0.4752 | **0.5637** | 26.6* | 0.5314 | 19.3* | 0.5596 | 25.7* | 0.5569 | 25.1* | 0.5585 | 25.4* |
| R-Prec | 0.2311 | 0.2298 | 0.2568 | **0.3419** | 48.0* | 0.3094 | 33.9* | 0.3366 | 45.7* | 0.3346 | 44.8* | 0.3357 | 45.3* |
| P@5 | 0.2222 | 0.2289 | 0.2436 | **0.2789** | 25.5* | 0.2559 | 15.2* | 0.2746 | 23.6* | 0.2746 | 23.6* | 0.2753 | 23.9* |

**Table 1: VSM vs. CE with VSM for computing local relevance (%chg denotes the performance improvement in percent of each model in CE; * indicates a statistically significant improvement over the baseline using the t-test, p-value $< 0.05$)**

| | Okapi | OptC | QC | VSM+Okapi | %chg | Okapi+Okapi | %chg | LM+Okapi | %chg | TR+Okapi | %chg | TRLM+Okapi | %chg |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MAP | 0.3401 | 0.2862 | 0.3622 | 0.4007 | 17.8* | 0.3977 | 16.9* | **0.4138** | 21.7* | 0.4082 | 20.0* | 0.4132 | 21.5* |
| MRR | 0.5406 | 0.4887 | 0.5713 | 0.6131 | 13.4* | 0.5884 | 8.8 | 0.6214 | 15.0* | 0.6172 | 14.2* | **0.6215** | 15.0* |
| R-Prec | 0.3178 | 0.2625 | 0.3345 | 0.3648 | 14.8* | 0.3613 | 13.7* | 0.3758 | 18.3* | 0.3677 | 15.7* | **0.3762** | 18.4* |
| P@5 | 0.2857 | 0.2824 | 0.2998 | 0.3140 | 9.9* | **0.3176** | 11.2* | 0.3161 | 10.6* | 0.3111 | 8.8 | 0.3147 | 10.2* |

**Table 2: Okapi vs. CE with Okapi for computing local relevance (%chg denotes the performance improvement in percent of each model in CE; * indicates a statistically significant improvement over the baseline using the t-test, p-value $< 0.05$)**

| | LM | OptC | QC [5] | LM+L [5] | VSM+LM | %chg | Okapi+LM | %chg | LM+LM | %chg | TR+LM | %chg | TRLM+LM | %chg |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MAP | 0.3821 | 0.3402 | 0.4083 | 0.4586 | **0.4620** | 20.9* | 0.4599 | 20.4* | 0.4609 | 20.6* | 0.4603 | 20.5* | 0.4616 | 20.8* |
| MRR | 0.5945 | 0.5219 | 0.6083 | 0.6620 | 0.6630 | 11.5* | 0.6651 | 11.9* | 0.6622 | 11.4* | 0.6633 | 11.6* | **0.6667** | 12.1* |
| R-Prec | 0.3404 | 0.3129 | 0.3624 | 0.4072 | **0.4101** | 20.5* | 0.4079 | 19.8* | 0.4087 | 20.1* | 0.4087 | 20.1* | 0.4100 | 20.4* |
| P@5 | 0.3040 | 0.2810 | 0.3230 | 0.3460 | 0.3512 | 15.5* | 0.3498 | 15.1* | **0.3519** | 15.8* | 0.3512 | 15.5* | 0.3513 | 15.6* |

**Table 3: LM vs. CE with LM for computing local relevance (%chg denotes the performance improvement in percent of each model in CE; * indicates a statistically significant improvement over the baseline using the t-test, p-value $< 0.05$)**

| | TR | OptC | QC | VSM+TR | %chg | Okapi+TR | %chg | LM+TR | %chg | TR+TR | %chg | TRLM+TR | %chg |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MAP | 0.4010 | 0.3417 | 0.4125 | **0.4592** | 14.5* | 0.4528 | 12.9* | 0.4507 | 12.4* | 0.4526 | 12.9* | 0.4522 | 12.8* |
| MRR | 0.6084 | 0.5392 | 0.6178 | **0.6607** | 8.6 | 0.6532 | 7.4 | 0.6527 | 7.3 | 0.6552 | 7.7 | 0.6540 | 7.5 |
| R-Prec | 0.3717 | 0.3175 | 0.3853 | **0.4153** | 11.7* | 0.4079 | 9.7* | 0.4054 | 9.1 | 0.4071 | 9.5* | 0.4058 | 9.2 |
| P@5 | 0.3168 | 0.2670 | 0.3280 | 0.3505 | 10.6* | **0.3519** | 11.1* | 0.3505 | 10.6* | 0.3497 | 10.4* | 0.3490 | 10.2* |

**Table 4: TR vs. CE with TR for computing local relevance (%chg denotes the performance improvement in percent of each model in CE; * indicates a statistically significant improvement over the baseline using the t-test, p-value $< 0.05$)**

| | TRLM | OptC | QC | VSM+TRLM | %chg | Okapi+TRLM | %chg | LM+TRLM | %chg | TR+TRLM | %chg | TRLM+TRLM | %chg |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MAP | 0.4369 | 0.3645 | 0.4570 | **0.4937** | 13.0* | 0.4823 | 10.4* | 0.4836 | 10.7* | 0.4876 | 11.6* | 0.4886 | 11.8* |
| MRR | 0.6316 | 0.5506 | 0.6551 | **0.6704** | 6.1 | 0.6652 | 5.3 | 0.6675 | 5.6 | 0.6685 | 5.8 | 0.6678 | 5.7 |
| R-Prec | 0.4008 | 0.3474 | 0.4196 | **0.4407** | 10.0 | 0.4349 | 8.5 | 0.4319 | 7.8 | 0.4331 | 8.1 | 0.4343 | 8.4 |
| P@5 | 0.3398 | 0.2910 | 0.3487 | **0.3570** | 5.1 | 0.3556 | 4.6 | 0.3541 | 4.2 | 0.3548 | 4.4 | 0.3527 | 3.8 |

**Table 5: TRLM vs. CE with TRLM for computing local relevance (%chg denotes the performance improvement in percent of each model in CE; * indicates a statistically significant improvement over the baseline using the t-test, p-value $< 0.05$)**

of the questions in the training data set.

We use the same *query set* as in previous work [5]. This set contains 252 queries and is available at http://homepages.inf.ed.ac.uk/gcong/qa. We remove the stop words from each question. For each method, the top 20 retrieval results are kept. For each query question, all the results from the different methods are combined and used for annotation. Thus annotators do not know which results are from which methods. An annotator is asked to label each returned question with "relevant" or "irrelevant." Two annotators are involved in the. If conflicts happen, a third person will make a decision for the final result.

**Metrics:** We evaluate the performance of our approaches using Mean Average Precision (MAP), Mean Reciprocal Rank (MRR), Precision@n, and R-Precision. MAP rewards approaches that return relevant questions early and also rewards correct ranking of the results. MRR gives us an idea of how far down in a ranked list we must look to find a relevant question. Precision@n reports the fraction of the top-$n$ questions retrieved that are relevant. R-Precision is the precision after $R$ questions have been retrieved, where $R$ is the number of relevant questions for the query. Note that the recall base for a query question consists of the relevant questions in the top 20 results from all approaches. The recall base is needed to compute some of measures that we use.

**Parameter Selection:** The experiments use three parameters. The first is the smoothing parameter $\lambda$ in the Language Model; the second, $\beta$, controls the self-translation impact in the Translation-Based Language Model; and the last parameter, $\alpha$, is used for linear interpolation in the CE model. Following the literature, we set $\lambda$ to 0.2 [20] and $\beta$ to 0.8 [19].

For the parameter $\alpha$, different combinations need different values. We do an experiment on a small development set of 20 queries to determine the best value among $0.1, 0.2, 0.3, ..., 0.9$ in terms of MAP for each combination in the CE model. This set is also extracted from the Yahoo! Answers data, and it is not included in the test *query set*. As a result, we use $\alpha = 0.1$ for the combinations using LM, TR, and TRLM to compute the local relevance. When computing the local relevance using VSM and Okapi, and computing the global relevance using Okapi, the value of $\alpha$ is set to 0.7 and 0.5, respectively; otherwise, $\alpha = 0.9$.

**Normalizing the Global and Local Relevance Scores:** The global and local relevance scores have different value ranges when they are computed by the five models introduced in Sections 3.2 and 3.3. We need to normalize them into approximately identical ranges before we can linearly combine the two for ranking. It is not easy to find the exact value range for each model. Making an estimation using the development data set with 20 queries, we find the approximate value range—shown in Table 7—for each model on this set.

We use the simple min-max normalization method to scale the values into the range [0,1]. We denote the original score as $S$, the

| | Global Relevance Score | Local Relevance Score |
|---|---|---|
| **VSM** | [0,3) | [0,1) |
| **Okapi** | [0,100) | [0,30) |
| **LM** | $(10^{-80}, 10^{-2})$ | $(10^{-50}, 10^{-2})$ |
| **TR** | $(10^{-80}, 10^{-2})$ | $(10^{-50}, 10^{-2})$ |
| **TRLM** | $(10^{-80}, 10^{-2})$ | $(10^{-50}, 10^{-2})$ |

**Table 7: Approximate value range for each model**

maximum value as $S_{max}$, the minimum value as $S_{min}$, and the normalized value as $N(S)$. We use the following equation for normalizing the global and local relevance scores (in LM, TR, and TRLM, the values are very small, and we use the logarithmic transformation of the original relevance values):

$$N(S) = \frac{S - S_{min}}{S_{max} - S_{min}}$$

**Training Word Translation Probability:** We use the GIZA++ toolkit for learning the IBM translation model 1 to get the word translation probabilities. After removing the stop words, we pool the question-description pairs and the description-question pairs together as the input to this toolkit [19]. The toolkit generates a word-to-word translation probability list after training.

## 4.2 Experimental Results

We present the experimental results in Section 4.2.1 and then analyze them in Section 4.2.2.

### *4.2.1 Results of CE Model*

We report experimental results for question retrieval using all the combinations in the CE model.

Table 1 shows the results of the five combinations that use VSM for local relevance computation (Section 3.3.1) and use the five models for global relevance computation (Section 3.2). The baseline method in this table is VSM. The results of VSM OptC (searching in the category containing the query question) and VSM QC (utilizing the query question classification) are also given. We can see that all the five CE models are able to significantly improve the performance of the baseline VSM. We will discuss the reasons for the improvement in the next section.

Similarly, Table 2 shows the results of five combinations that use Okapi for computing the local relevance (Section 3.3.2). The results of the baseline Okapi and of Okapi OptC and Okapi QC are also given in this table. Next, Table 3 shows the results of the baseline LM, of LM OptC and LM QC, and of the five combinations using LM for computing the local relevance (Section 3.3.3). Similarly, Table 4 shows the results of the baseline TR, of TR OptC and TR QC, and of the five combinations using TR for computing the local relevance (Section 3.3.3). Finally, Table 5 shows the results of the baseline TRLM, of TRLM OptC and TRLM QC, and of the five combinations using TRLM for computing the local relevance (Section 3.3.3).

Observations similar to those made for Table 1 can be made for Tables 2–5: Thus, all the combinations in the CE model consistently outperform the corresponding baseline methods. Incorporating the global relevance score computed by the five models all leads to a better performance, comparing with the corresponding baseline methods that do not exploit the category information. Most of these combinations can achieve statistically significant improvements in terms of all the measures. This clearly shows that the category information indeed can be utilized to improve question retrieval, and it shows that the CE model is effective.

We can see that among the five baseline models, TRLM performs the best and VSM performs the worst; TR, LM, and Okapi are in between. TRLM and TR have better performance since they are able to retrieve questions that do not share common words with the query, but are semantically similar to the query. Because TRLM solves the self-translation problem by combining TR and LM, it has the best performance. The results are consistent with those reported in previous work [10, 19].

Among the five methods for computing global relevance, we find that VSM and LM perform the best. TRLM does not outperform the other models, although it performs the best when used to compute the local relevance. The underlying reasons are analyzed in the next section.

The OptC models perform even worse than the baseline models. The reason is that for many queries, not all the relevant questions come from the same category as the query question. Table 8 shows that searching in only the top categories results in many relevant questions being missed.

The QC models perform better than the corresponding baseline models, which is consistent to result reported in previous work [5]. However, comparing to the results of the CE model the improvement is minor. In Table 3, we also give the results of the LM+L method [5]. We can see that its results are comparable to the results of the CE model when using LM. However, the CE model can be applied to other question retrieval models, rather than being restricted to LM.

### *4.2.2 Result Analysis*

To understand why the combinations in the CE model improve over the baseline models, let us recall the two components of the CE model, namely (1) the global relevance score of a query and the category containing a historical question, and (2) the local relevance score of a query and a historical question. Each component utilizes category information and contributes to the improvement.

**Global Relevance Score Analysis:** The CE model promotes the rankings of the questions in categories with larger global relevance scores. The introduction of a global relevance component in the CE model is based on the intuition that the more related a category is to a query question, the more likely the category is to contain questions relevant to the query. To verify this intuition, we analyze the relevant questions returned by the TRLM model, which is the best baseline.

For each query, we rank all the categories according to the global relevance scores of each category and the query, and we count the number of relevant questions in every rank. We then aggregate the number of relevant questions in each rank across all queries and compute the percentage of relevant questions in each rank.

Table 8 gives the results with regard to the five global relevance scores computed by the five models. It shows that most relevant questions come from the top-5 ranks, which is consistent with our intuition. Hence, it is helpful to promote the rankings of questions in the categories with higher global relevance scores. We also notice that some relevant questions in "Rest" can come from categories ranked at about 500.

| models of global relevance | Rank | | | | | |
|---|---|---|---|---|---|---|
| | **1** | **2** | **3** | **4** | **5** | **Rest** |
| **VSM** | 69.4 | 6.4 | 5.8 | 3.1 | 2.0 | 14.3 |
| **Okapi** | 61.7 | 11.0 | 6.3 | 3.2 | 2.8 | 15.0 |
| **LM** | 65.8 | 9.8 | 4.6 | 2.0 | 3.6 | 14.1 |
| **TR** | 61.8 | 11.0 | 5.3 | 2.8 | 3.0 | 16.1 |
| **TRLM** | 65.5 | 9.6 | 4.4 | 1.8 | 3.6 | 15.1 |

**Table 8: Distribution of relevant questions in different ranks of category in percentage (%, TRLM used for local relevance)**

**Comparing the models for computing global relevance:** All the five models improve the retrieval performance, as shown in Tables 1-5. However, TR and TRLM perform slightly worse than LM when used to compute the global relevance, although they have better results when used as baseline retrieval models. A possible reason why the two models perform better than LM when they are used as baselines is that the word translation probabilities help them find semantically similar, but lexically different, relevant questions.

However, category documents are usually much longer than query questions; when finding a relevant category, the category document usually shares the same words with the query, and the role of semantically similar, but lexically different words is less important. The word translation probabilities sometimes might even result in noise in this case. We can also see from the results that Okapi performs the worst when used to compute the global relevance.

To illustrate the analysis, consider the query question "How can I stop my cat worrying/nibbling her own leg?" that belongs to the category "Pets.Cats." Using Okapi to compute the global relevance yields a score difference between "Pets.Cats" and "Pets.Dogs" of $0.0125$, after normalization; however, the difference is $0.2$ if we use VSM. With LM, the difference between the two categories is about $0.05$ after normalization, and with TR and TRLM it is only about $0.014$. Due to the word translation probabilities T(dog|cat) $= 0.005$ and T(dog|leg) $= 0.017$ as determined by TR and TRLM, "Pets.Dogs" is also very relevant to the query when using TR and TRLM. From Table 8, we can also see the trend that the top-1 category as determined by VSM contains more relevant questions than for the other models.

**Local Relevance Score Analysis:** In VSM and Okapi, the local relevance is computed using IDF with regard to the category of a historical question rather than the whole collection. In LM, TR, and TRLM, it is computed with regard to the category of a historical question for the smoothing.

To see the effects of the category-specific local IDF, we compare the results of the model VSM+VSM using local IDF (denoted as VSM+VSM.Local) and global IDF (VSM+VSM.Global). Table 9 shows the results of these two and VSM as the baseline. We can see that VSM+VSM.Local performs better than VSM+VSM.Global.

This is because the local IDF can better represent the importance of a word for a specific category when the local relevance is used to distinguish questions within a category. For example, the query "What is the ideal temperature for a ball python?" is contained in the category "Pets.Reptiles." In the full collection, 2,322 questions contain the word "temperature" and 401 questions contain "python"; but in the category "Pets.Reptiles," only 16 questions contain "temperature" whereas 332 questions contain "python." As a result, "temperature" is more important than "python" within the category, while "python" is more important within the full collection.

Using global IDF will rank questions containing "python" higher for the questions in the category "Pets.Reptiles"; however, questions in this category that contain "temperature" are more likely to be relevant. Using local IDF will be more suitable for the retrieval within a category. Similarly, in the Okapi Model, the local IDF also improves the retrieval performance.

In LM, TR, and TRLM, the smoothing plays a role similar to that of IDF [20]. The three models also improve in performance when the smoothing is done in the categories instead of on the whole collection. We compare the results of LM when using local smoothing, denoted as LM.Local, and when using global smoothing, denoted as LM.Global. Table 10 shows the MAP results when different models are used to compute the global relevance. We can see that the smoothing in a category performs better than smoothing in the whole collection, no matter which kind of model is used to compute the global relevance score. In TR and TRLM the local smoothing also leads to better performance, and we will not give details due to space limitation.

| | **MAP** | **MRR** | **R-Prec** | **P@5** |
|---|---|---|---|---|
| VSM+VSM.Local | 0.3711 | 0.5637 | 0.3419 | 0.2789 |
| VSM+VSM.Global | 0.2857 | 0.4843 | 0.2682 | 0.2487 |
| VSM | 0.2407 | 0.4453 | 0.2311 | 0.2222 |

**Table 9: Results of VSM+VSM.local, VSM+VSM.global and VSM baseline**

| | **VSM** | **Okapi** | **LM** | **TR** | **TRLM** |
|---|---|---|---|---|---|
| LM.Local | 0.4620 | 0.4599 | 0.4609 | 0.4603 | 0.4616 |
| LM.Global | 0.4305 | 0.4235 | 0.4275 | 0.4287 | 0.4291 |

**Table 10: MAP results of LM using local smoothing or global smoothing for computing local relevance score**

Tables 9 and 10 also show that the global relevance component improves the performance even if we use the global IDF or global smoothing to compute the local relevance score. This occurs because the rankings of questions from relevant categories are promoted.

**Comparing the two methods of using VSM to compute global relevance.** In Section 3.2.1, the second method of computing the global relevance score using VSM is to find and use the centroid vector of a category to compute the similarity. We denote this method as VSMCEN. Table 13 compares VSM and VSMCEN for computing global relevance when VSM is used to compute local relevance (the results are comparable when using other models to compute local relevance). The results show that VSMCEN can also improve the retrieval performance, but that VSM consistently performs the best. The reason is given in Section 3.2.1.

**Examples:** To better understand the CE model, Table 11 gives the top-3 results of the query question "Do guppies die after giving birth?" from the category "Pets.Fish," when using the combination VSM+VSM. The questions in bold are labeled as "relevant." First, we see that in the results of both VSM+VSM.Local and VSM+VSM.Global, the global relevance component promotes the question in category "Pets.Fish." This is because the category has a high relevance score for the query, while the relevance scores of the categories "Pregnancy & Parenting" and "Dream Interpretation" are lower. Second, VSM+VSM.Local using the local IDF yields a better ranking of questions within the same category. This is because in the method VSM+VSM.Local, the words "giving" and "birth" are more important than the word "guppies," while the opposite is true in the method VSM+VSM.Global.

The experimental results show that TRLM has superior performance when used alone as baseline or combined with models of global relevance. It gains from the word translation probabilities. Questions that do not share many common words with the query will have low rankings in other non-translation models. However, translation-based models are able to fill the lexical gap and find semantically similar questions. Table 12 gives the results of the query "Will marijuana leave my system faster if i drink alot of water?" from category "Health.Alternative." We can see that the three translation-based models rank the two relevant questions higher than LM. This is due to the two word translation probabilities: T(marijuana|thc) $= 0.128$ and T(marijuana|weed) $= 0.053$. In addition, VSM+TRLM performs better than TRLM because after normalization, the global relevance score between the query and the ".Beer, Wine & Spirits" category is as high as 0.91.

| Method | Top-3 Retrieval Results | Category |
|---|---|---|
| VSM | 1. What if i die while giving the birth? | Pregnancy & Parenting |
| | 2. Giving birth? | Dream Interpretation |
| | 3. Do you like guppies? | Fish |
| VSM+VSM.Global | 1. Do you like guppies? | Fish |
| | 2. How many guppies do I have? | Fish |
| | 3. **Is it at all normal for a guppy mother to die a day or so after giving birth?** | Fish |
| VSM+VSM.Local | 1. **Is it at all normal for a guppy mother to die a day or so after giving birth?** | Fish |
| | 2. Lifespan of platy females after giving birth? | Fish |
| | 3. Will my Guppies die if I get them to early? HELP!? | Fish |

**Table 11: Search results for "Do guppies die after giving birth?"**

| Method | Relevant Questions and their ranks | Category |
|---|---|---|
| LM | 10. **Will water flush thc from your system faster?** | Botany |
| TR | 1. **Will water flush thc from your system faster?** | Botany |
| | 4. **Does anyone know what helps take "weed" out of ur system quick? any drinks? lots of water?** | Beer, Wine & Spirits |
| TRLM | 1. **Will water flush thc from your system faster?** | Botany |
| | 15. **Does anyone know what helps take "weed" out of ur system quick? any drinks? lots of water?** | Beer, Wine & Spirits |
| VSM+TRLM | 1. **Will water flush thc from your system faster?** | Botany |
| | 10. **Does anyone know what helps take "weed" out of ur system quick? any drinks? lots of water?** | Beer, Wine & Spirits |

**Table 12: Search results for "Will marijuana leave my system faster if i drink alot of water?"**

| | MAP | MRR | R-Prec | P@5 |
|---|---|---|---|---|
| VSM+VSM | 0.3711 | 0.5637 | 0.3419 | 0.2789 |
| VSMcen+VSM | 0.2854 | 0.4791 | 0.2679 | 0.2401 |

**Table 13: Comparison (MAP) of VSM and VSMCEN**

| Deviation range | Improved queries (%) |
|---|---|
| $[0, 0.006)$ | 62.5 |
| $[0.006, 0.021)$ | 74.8 |
| $[0.021, 0.053]$ | 84.4 |

**Table 14: Percentage of queries with better performance for different ranges of global relevance score deviation in terms of MAP**

**Features of Questions that Benefit from CE:** For some queries, the CE model performs worse than the baseline methods. We find that this is mainly because the global relevance score promotes the rankings of questions in the non-relevant categories. If a query question does not have obvious category-specific features, the global relevance score will not help.

We would like to characterize the queries that can benefit from the CE model, although this is very difficult. The intuition behind the CE model indicates that if a query obviously belongs to certain categories, it is more likely that the global relevance will help; otherwise, if a query has similar category relevance scores for every category, category information is less likely to help.

Based on this, for each query we calculate the standard deviation of the global relevance scores computed by VSM between it and all the categories. The deviation measures the variability of the global relevance scores across categories. A large deviation means that a query has obvious category features. According to the deviation value of each query, we divide our test set into three subsets of equal size. Table 14 lists the percentage of queries that have better performance in each subset in terms of MAP when using VSM+VSM. It can be seen that with larger deviation, the CE model is more likely to perform better. We also analyzed the results from the perspectives of other features, such as the query length and category length, but did not find any useful correlations.

**Summary:** The experimental study shows that all the five methods of computing global relevance are capable of improving the performance of question retrieval models, and it shows that our variation of VSM is the best at computing the global relevance. The study also show that the strategy of computing the local relevance score with regard to categories is effective. Over all the combinations, the model VSM+TRLM achieves the best performance.

# 5. RELATED WORK

The work on question retrieval can be traced back to finding similar questions in Frequently Asked Questions (FAQ) archives. Most of the existing work focuses on addressing the word mismatching problem between user questions and the questions in a Question Answering (QA) archive. The word mismatching problem is especially important for QA retrieval, since question-answer pairs are usually short [19]. Burke et al. [4] combine lexical similarity and semantic similarity between questions to rank FAQs. Berger et al. [2] study several statistical approaches to bridge the lexical gap for FAQ retrieval, including a translation-based approach and a query expansion approach using a word mapping learned from a collection of question-answer pairs. Jijkoun and Rijke [11] use a vector space model for question-answer pair retrieval. Riezler et al. [14] provide a translation model for question retrieval; their translation model is trained on a large amount of data extracted from FAQ pages on the Web. Soricut and Brill [17] build an answer passage retrieval system.

Question retrieval has recently been revisited for Community Question Answering (CQA) data. Jeon et al. [9, 10] compare four different retrieval methods, i.e., the vector space model, the Okapi model, the language model, and the translation model, for question retrieval on CQA data, and the experimental results show that the translation model outperforms the other models. In subsequent work [19], they propose a translation-based language model that combines the translation model and the language model for question retrieval. The results reported in the work on CQA data are consistent with the results reported on other QA archives, e.g., FAQs: translation models usually help question retrieval since they can effectively address the word mismatch problem of questions. Additionally, they also explore answers in question retrieval.

Duan et al. [7] propose a solution that makes use of question structures for retrieval by building a structure tree for questions in a category of Yahoo! Answers, meaning that important phrases in questions are given higher weight in question matching. Wang et al. [18] employ a parser to build syntactic trees of questions, and questions are ranked based on the similarity between their syntactic trees and the syntactic tree of the query question. As observed by Wang et al. [18], current parsers are not well-trained to parse real-life questions, especially informally stated questions. They report that they outperform a Bag-of-Word baseline (that matches words between query and question) by 11.99% in terms of MAP. Bian et al. [3] propose an interesting learning framework for question retrieval. However, this approach needs training data (which would be difficult to get for general questions) and experiments are conducted on factoid questions.

In contrast to the previous work, our technique exploits the question categories in CQA data. The only work on utilizing category information for question retrieval is based on language models [5]. However, the effective category smoothing based approach [5] is tightly integrated with the langauge model and is difficult to apply to other retrieval models. In contrast, the technique proposed in this paper can be applied to any question retrieval method. The existing classification-based approach [5] can be applied to other question retrieval models, but can only slightly improve the performance, as shown in our experiments.

Chekuri and Raghavan [6] introduce the idea of utilizing classification for document retrieval. However, their focus is on the automatic classification of Web documents (no category information available) into high-level categories of the Yahoo! taxonomy; they do not investigate how to leverage the classification results for document retrieval. Similarly, Lam et al. [12] also focus on developing an automatic text categorization approach.

Finally, we note that other works exists on CQA services that consider category information, e.g., [1, 13]. However, these focus on other aspects of CQA, not on question retrieval.

# 6. CONCLUSIONS

Question retrieval is an essential component in Community Question Answer (CQA) services. In this paper, we propose a new approach that is capable of exploiting category information associated with questions in CQA archives for improving question retrieval. This approach can be applied easily to existing question retrieval models. Experiments conducted on a large CQA data set from Yahoo! Answers demonstrate the effectiveness of the proposed technique.

This work opens to several interesting directions for future work. First, it is of relevance to apply the proposed techniques to other question retrieval approaches (e.g., [7]). Second, it is interesting to include answers into our proposed technique for question retrieval. Third, hierarchical category structures may perhaps be exploited to further improve the performance of the CE model. Finally, it is of interest to explore other ways of combining global relevance scores and local relevance scores.

## Acknowledgments

C. S. Jensen is also an Adjunct Professor at University of Agder, Norway.

# 7. REFERENCES

[1] E. Agichtein, C. Castillo, D. Donato, A. Gionis, and G. Mishne. Finding high-quality content in social media. In *WSDM*, pp. 183–194, 2008.

[2] A. Berger, R. Caruana, D. Cohn, D. Freitag, and V. Mittal. Bridging the lexical chasm: statistical approaches to answer-finding. In *SIGIR*, pp. 192–199, 2000.

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

[4] R. D. Burke, K. J. Hammond, V. A. Kulyukin, S. L. Lytinen, N. Tomuro, and S. Schoenberg. Question answering from frequently asked question files: Experiences with the faq finder system. *AI Magazine*, 18(2):57–66, 1997.

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

[6] C. Chekuri, M. H. Goldwasser, P. Raghavan, and E. Upfal. Web search using automatic classification. In *WWW*, 1997.

[7] H. Duan, Y. Cao, C.-Y. Lin, and Y. Yu. Searching questions by identifying question topic and question focus. In *ACL-HLT*, pp. 156–164, 2008.

[8] H. Fang, T. Tao, and C. Zhai. A formal study of information retrieval heuristics. In *SIGIR*, pp. 49–56, 2004.

[9] J. Jeon, W. B. Croft, and J. H. Lee. Finding semantically similar questions based on their answers. In *SIGIR*, pp. 617–618, 2005.

[10] J. Jeon, W. B. Croft, and J. H. Lee. Finding similar questions in large question and answer archives. In *CIKM*, pp. 84–90, 2005.

[11] V. Jijkoun and M. de Rijke. Retrieving answers from frequently asked questions pages on the web. In *CIKM*, pp. 76–83, 2005.

[12] W. Lam, M. Ruiz, and P. Srinivasan. Automatic text categorization and its application to text retrieval. *IEEE TKDE*, 11(6):865–879, 1999.

[13] Y. Liu, J. Bian, and E. Agichtein. Predicting information seeker satisfaction in community question answering. In *SIGIR*, pp. 483–490, 2008.

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

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

[16] A. Singhal, C. Buckley, and M. Mitra. Pivoted document length normalization. In *SIGIR*, pp. 21–29, 1996.

[17] R. Soricut and E. Brill. Automatic question answering: Beyond the factoid. In *HLT-NAACL*, pp. 57–64, 2004.

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

[19] X. Xue, J. Jeon, and W. B. Croft. Retrieval models for question and answer archives. In *SIGIR*, pp. 475–482, 2008.

[20] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM TOIS*, 22(2):179–214, 2004.

[21] J. Zobel and A. Moffat. Inverted files for text search engines. *ACM Computing Surveys*, 38(2), 56 pages, 2006.