Approaches to Exploring Category Information for Question Retrieval in Community Question-Answer Archives

Xin Cao, Nanyang Technological University, Singapore
Gao Cong, Nanyang Technological University, Singapore
Bin Cui, Peking University, China
Christian S. Jensen, Aarhus University, Denmark
Quan Yuan, Nanyang Technological University, Singapore

Community Question Answering (CQA) is a popular type of service where users ask questions and where answers are obtained from other users or from historical question-answer pairs. CQA archives contain 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. This paper presents several new approaches to exploiting the category information of questions for improving the performance of question retrieval, and it applies these approaches 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 effective and efficient and are capable of outperforming a variety of baseline methods significantly.

Categories and Subject Descriptors: H.2.4 [**Database Management**]: Systems—*textual databases*; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; H.3.4 [**Information Storage and Retrieval**]: Systems and Software; H.3.5 [**Information Storage and Retrieval**]: Online Information Services—*web-based services*

General Terms: Algorithms, Experimentation, Performance

Additional Key Words and Phrases: question-answering services, question search, categorization, cluster-based retrieval

1. INTRODUCTION

1.1. Background and Motivation

Community Question Answering (CQA) services are Internet services that enable users to ask and to receive answers to questions. Answers are either contributed by other users or are retrieved from historical question-answer pairs. Examples of such community-driven knowledge market services include Yahoo! Answers (answers.yahoo.com)Naver (www.naver.com), Baidu Zhidao (zhidao.baidu.com), and WikiAnswers (wiki.answers.com).

CQA services can return actual answers to the query questions of users, instead of long lists of ranked URL. They thus provide an effective alternative to web search [Agichtein et al. 2008; Xue et al. 2008]. Question retrieval can also be considered as an alternative to the traditional TREC Question Answering (QA) task. In the context of TREC QA, 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 such real-world questions are usually more

Portions of this work appeared in Cao et al. [2009, 2010].

Authors' addresses: Xin Cao, Gao Cong^{+} (Contact author, gaocong@ntu.edu.sg), and Quan Yuan, School of Computing Engineering, Nanyang Technological University; Bin Cui, Department of Computer Science and Key Laboratory of High Confidence Software Technologies (Ministry of Education), Peking University; Christian S. Jensen, Department of Computer Science, Aarhus University;

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

complex and are often presented in an informal way. Hence, the TREC QA task of extracting a correct answer is transformed into a question retrieval task [Xue et al. 2008; Wang et al. 2009].

Although CQA services emerged only recently , they have built up large archives of historical questions and associated 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 having to wait for a human to respond can be avoided, thus improving user satisfaction. It is important that a search service offers relevant results efficiently. This paper's focus is to improve question search for CQA services.

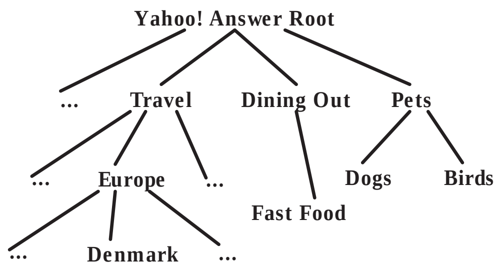

Fig. 1. The Category Structure of Yahoo! Answers

Works on question retrieval in CQA data [Bian et al. 2008; Duan et al. 2008; Jeon et al. 2005a,b; Xue et al. 2008] employ different retrieval models, including the vector space model [Duan et al. 2008; Jeon et al. 2005a,b], the language model [Duan et al. 2008; Jeon et al. 2005b], the Okapi model [Jeon et al. 2005b], and the translation model [Berger et al. 2000; Riezler et al. 2007; Jeon et al. 2005b; Xue et al. 2008]. The experimental studies in previous works show that the translation model usually achieves the best performance [Jeon et al. 2005b; Xue et al. 2008], or is able to improve on other methods [Duan et al. 2008]. Other recent proposals for question retrieval include the use of learning-to-rank techniques [Bian et al. 2008] and the exploitation of syntactic structures of questions [Wang et al. 2009]. However, it is not clear whether these recent proposals are competitive with other methods, such as the language model and the translation model, as empirical comparisons are absent.

A distinctive feature of question-answer archives is that CQA services usually organize questions according to a hierarchy of categories. Figure 1 shows a small part of the hierarchy of 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.

Although recent work has been done on question search in CQA data, we are not aware of work that aims to exploit the available categorizations of questions for question search. To exemplify how a categorization of questions may be exploited, consider a user who enters the following query question ($\mathbf{q}$): "Can you recommend sightseeing opportunities for senior citizens in Denmark?" The user is interested in sightseeing specifically in Denmark, not in other countries. Hence, the historical question ($\mathbf{d}$) "Can you recommend sightseeing opportunities for senior citizens in China?" and its answers are not relevant to the user's question although the two questions are syntactically very similar, making it likely that existing question-answer search approaches will rank question $\mathbf{d}$ highly among the list of returned results. If one could establish a connection between $\mathbf{q}$ and the category “Travel.Europe.Denmark,” the ranking of questions in that category could be promoted, thus perhaps improving the question retrieval performance.

In this paper we propose several new approaches to exploiting categorization information for question retrieval, and these approaches are applied to several existing question retrieval models.

1.2. The First Idea

We observe that category specific frequent words (words that occur relatively frequently in a category, such as “Denmark” in category “Travel.Europe.Denmark”) can be utilized to improve the retrieval. On the one hand, such words are helpful in distinguishing questions across different categories. Questions containing such words tend to be contained in relevant categories, and they are more likely to be relevant to the query than those that do not contain such words. Thus, they deserve higher ranking scores. On the other hand, when comparing the relevancies of questions within a category, frequent words are not as useful as other words. For example, questions containing “Denmark” are quite likely to be contained in “Travel.Europe.Denmark,” and they are more likely to be relevant to our earlier query question $\mathbf{q}$. However , the word “Denmark” will be unimportant when we compare the relevancies of questions in “Travel.Europe.Denmark,” to $\mathbf{q}$, since nearly all questions in this category are about “Denmark.”

Approach 1: Leaf category smoothing enhancement

Based on this idea, we propose our first approach, which can be combined with models containing a smoothing component, i.e., the language model [Jeon et al. 2005b], the translation model [Berger et al. 2000; Riezler et al. 2007; Jeon et al. 2005b], and the translation-based language model [Xue et al. 2008], a state-of-the-art method. Taking the language model as an example, 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. An in-depth theoretical justification for the approach is offered in the paper. However, this approach is applicable only to models with smoothing, and it is not applicable to other question retrieval models. We call this approach LS.

1.3. The Second Idea

The more relevant a category is to a query, the more likely it is that the category contains questions relevant to the query. Based on this idea, the questions from relevant categories should be promoted in some way. For example, considering the query question $\mathbf{q}$, more relevant questions are expected to be found in category “Travel.Europe.Denmark” than in category “Travel.Asia.China”.

Combining this idea with the first idea, i.e., that frequent words in a category are less useful in distinguishing questions from the same category, we develop two different approaches to computing the relevance of a category to a query.

Approach 2: Category enhancement

We treat each category as one big pseudo-document when computing the relevance of a category to a query $\mathbf{q}$. Specially, we rank a historical question $\mathbf{d}$ based on an interpolation of two relevance scores: the global relevance score between query $\mathbf{q}$ and the *category* containing $\mathbf{d}$ (denoted as $Cat(\mathbf{d})$) and the local relevance score between query $\mathbf{q}$ and question $\mathbf{d}$ with regard to $Cat(\mathbf{d})$.

The global relevance score measures the relevance of a category to $\mathbf{q}$, and is used to distinguish among questions from different categories. All historical questions in a category are concatenated and form one big pseudo-document when computing this score. Questions contained in more relevant categories will have higher global relevance scores and will thus be favored in the retrieval.

The local relevance score measures the relevance of $\mathbf{d}$ to $\mathbf{q}$ with regard to $Cat(\mathbf{d})$, and is used to distinguish questions within the same category. The category is treated as the collection when computing this score. Thus, we compute document frequency with regard to the category instead of the whole collection. Category specific frequent words (such as “Denmark”) will result in low local relevance scores of questions, since they are less important to the retrieval when focusing on a category. This is a general approach, and it can be applied to any existing question retrieval model. We call this model CE.

Approach 3: Query classification enhancement

We use a classifier to estimate the relevance of each category to $\mathbf{q}$. We compute the probability of $\mathbf{q}$ belonging to each category, which measures the relevance of $\mathbf{q}$ to each category. Then the probability is used to adjust the ranking scores of a historical questions. Questions from categories with high classification probability scores are promoted. We call this approach as QC.

We show that, formally, this approach can be applied to models that use probabilities as ranking scores, i.e., the language model, the translation model, and the translation-based language model. We also apply the approach to other types of models, and the experimental results show that the performance is improved as well.

Additionally, we utilize the query classification results to prune the search space to improve efficiency. Remember that the classification probabilities measure the relevances of categories to the query. Therefore we can prune by considering only questions within categories with probability scores larger than a threshold.

1.4. The Third Idea

Because users may submit questions to non-relevant categories by mistake and because some categories in a hierarch may overlap, it is inappropriate to fully trust in the category labels of questions. For example, some categories in Yahoo! Answers overlap. Specifically, many categories have an "Other" sub- category that overlaps with their other sub-categories. We should try to map a historical question to its "truly relevant" categories, and then utilize this corrected category information in question retrieval.

Approach 4: Question classification enhancement

We compute the historical question classification probability scores and utilize these to avoid the negative effects caused by erroneously submitted questions and overlapping categories. We estimate the probability of each historical question $\mathbf{d}$ belonging to each category. If the category label of $\mathbf{d}$ is consistent with the classification results, we compute ranking scores as in the original model. Otherwise, we use the classification probability scores to adjust the ranking scores of $\mathbf{d}$ with regard to each relevant category as computed by some retrieval model. Theoretically, this approach can only be applied to models that use probabilities as ranking scores. It can be incorporated with the three approaches introduced above. We call this approach DC.

1.5. Contribution and Outline

In summary, the paper's contributions are twofold. First, we propose four approaches to enhancing question search with categorization information. Second, we conduct experiments with a large real data set consisting of more than 3 million questions from Yahoo! Answers to empirically elicit pertinent properties of the techniques that make up the proposed frameworks. Experimental results show that the proposed techniques using category information are capable of significantly improving existing question retrieval models, including a state-of-the-art model [Xue et al. 2008] in terms of both effectiveness and efficiency, i.e., category information is indeed useful for improving question search.

The paper is organized as follows. Section 2 covers existing question retrieval models. Section 3, 4, 5, and 6 detail the proposed techniques. Section 7 covers the experimental study. Section 8 reviews related work. Finally, Section 9 concludes and identifies research directions.

2. PRELIMINARIES

We proceed to offer an overview of existing question retrieval models.

2.1. Vector Space Model

The Vector Space Model (VSM) has been used widely in question retrieval [Jeon et al. 2005b; Jijkoun and de Rijke 2005]. We consider a popular variation of this model [Zobel and Moffat 2006]: 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:

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

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. Finally, $w_{\mathbf{q},t}$ 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 VSM favors short questions, the Okapi BM25 Model [Robertson et al. 1994a] (Okapi) takes into account the question length to overcome this problem. Okapi is used for question retrieval by Jeon et al. [2005b]. 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, f_t, and $tf_{t,\mathbf{d}}$ are as defined for VSM; k_1, b, and k_3 are parameters that are set to 1.2, 0.75, and ∞, respectively, thus following Robertson et al. [1994a] (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 (LM) is used in previous work [Cao et al. 2009; Duan et al. 2008; Jeon et al. 2005b] for question retrieval. The basic idea of LM 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 [Zhai and Lafferty 2004]. 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 λ is the smoothing parameter.

2.4. Translation Model

Previous work [Berger et al. 2000; Jijkoun and de Rijke 2005; Riezler et al. 2007; Duan et al. 2008; Xue et al. 2008] consistently reports that the Translation Model yields superior performance for question retrieval. The Translation Model (TR) exploits word translation probabilities in the language modeling framework. We follow the translation model approach of Jeon et al. [2005b]. 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. [2005b] assume that the probability of self-translation is 1, meaning that $T(w|w) = 1$.

2.5. Translation-based Language Model

A recent approach [Xue et al. 2008] to question retrieval (TRLM) combines the LM and TR. It is shown [Xue et al. 2008] that this model gains better performance than both the LM and TR. Given a query $\mathbf{q}$ and a question $\mathbf{d}$, the ranking score $S_{\mathbf{q},\mathbf{d}}$ is computed as follows:

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

where parameter eta controls the translation component's impact and the other parameters are as defined earlier.

3. APPROACH 1: LEAF CATEGORY SMOOTHING ENHANCEMENT

The smoothing in LM is used to adjust the maximum likelihood estimator so as to correct the inaccuracy due to data sparseness [Fang et al. 2004]. It is also used in TR and TRLM and plays the same role as in LM. In this section, we introduce an approach that incorporates smoothing on leaf categories into the three models to enhance the question retrieval.

3.1. Language Model with Leaf Category Smoothing

Each question in Yahoo! Answers belongs to a leaf category. To realize the first idea presented in the introduction, this approach utilizes category information of historical queries such that category-specific frequent words assume an important role in comparing the relevancies of historical questions across categories to a query, while category-specific frequent words are less important than category-specific infrequent words in comparing the relevancies of questions within a category.

This idea can be realized by performing two levels of smoothing. The category language model is first smoothed with the whole question collection, and then the question language model is smoothed with the category model. We next present the two levels of smoothing and then show why this smoothing meets the requirements.

Given a query question $\mathbf{q}$ and a candidate historical question $\mathbf{d}$ (from a QA archive), we compute the probability $P(\mathbf{q}|\mathbf{d})$ of how likely $\mathbf{q}$ could have been generated from $\mathbf{d}$. $P(w|\mathbf{d})$ is used as the retrieval model to measure how relevant a historical question $\mathbf{d}$ is to query question $\mathbf{q}$. To compute $P(\mathbf{q}|\mathbf{d})$, we then need to estimate language model $P(w|\mathbf{d})$. In this approach, Equation 3 is modified as follows.

$$\begin{aligned} P(\mathbf{q}|\mathbf{d}) &= \prod_{w \in \mathbf{q}} P(w|\mathbf{d}) \\ P(w|\mathbf{d}) &= (1-\lambda) P_{ml}(w|\mathbf{d}) + \lambda[(1-\beta) P_{ml}(w|Cat(\mathbf{d})) + \beta P_{ml}(w|Coll)], \end{aligned} \tag{6}$$

where w is a word in query $\mathbf{q}$, λ and β are two different smoothing parameters, $Cat(\mathbf{d})$ denotes the category of historical question $\mathbf{d}$, $P_{ml}(w|\mathbf{d})$ is the maximum likelihood estimate of word w in question $\mathbf{d}$ and can be computed by Equation 3, and $P_{ml}(w|Cat(\mathbf{d})) = \frac{tf(w,Cat(\mathbf{d}))}{\sum_{w' \in Cat(\mathbf{d})} tf(w',Cat(\mathbf{d}))}$ is the maximum likelihood estimate of word w in the $Cat(\mathbf{d})$, and $P_{ml}(w|Coll) = \frac{tf(w,Coll)}{\sum_{w' \in Coll} tf(w',Coll)}$ is the maximal likelihood estimate of word w in the Collection. The ranking score for candidate question $\mathbf{d}$ using the query likelihood language model can be computed using Equation 6.

3.2. Translation Model with Leaf Category Smoothing

The leaf category smoothing component can also be used in the Translation Model. The new model is described as follows:

$$\begin{aligned} S_{\mathbf{q},\mathbf{d}} = \prod_{t \in \mathbf{q}} ((1-\lambda) \sum_{w \in \mathbf{d}} T(t|w) P_{ml}(w|\mathbf{d}) \\ + \lambda[(1-\beta)P_{ml}(w|Cat(\mathbf{d})) + \beta P_{ml}(w|Coll)]), \end{aligned} \tag{7}$$

where all parameters are as defined earlier.

3.3. Translation-based Language Model with Leaf Category Smoothing

Translation-based Language Model with leaf category smoothing can be described as follows:

$$\begin{aligned} S_{\mathbf{q},\mathbf{d}} = \prod_{t \in \mathbf{q}} ((1-\lambda)(\eta \sum_{w \in \mathbf{d}} T(t|w) P_{ml}(w|\mathbf{d}) + (1-\eta) P_{ml}(t|\mathbf{d})) \\ + \lambda[(1-\beta)P_{ml}(w|Cat(\mathbf{d})) + \beta P_{ml}(w|Coll)]), \end{aligned} \tag{8}$$

where all parameters are as defined earlier.

3.4. Analysis

We proceed to prove that leaf smoothing improves the retrieval in two aspects: first, it can distinguish questions across different categories; second, it can distinguish questions from the same category (the same role as the inverse document frequency (IDF) in retrieval). We use LM as an example in the analysis. The same findings can be obtained using TR and TRLM using the same analysis method.

We proceed to show that category-specific frequent words play an important role in determining the relevancies of questions across different categories in this model to a query $\mathbf{q}$. We define a model P_{cs} for "seen" words that occur in the category $Cat(\mathbf{d})$ (i.e., $tf(w, Cat(\mathbf{d})) > 0$), and P_{cu} for "unseen" words that do not occur in the category $Cat(\mathbf{d})$ (i.e., $tf(w, Cat(\mathbf{d})) = 0$). The probability of a query $\mathbf{q}$ being generated from $\mathbf{d}$ can be written as follows:

$$\begin{aligned} \log P(\mathbf{q}|\mathbf{d}) &= \sum_{w \in \mathbf{q}} \log P(w|\mathbf{d}) \\ &= \sum_{w \in \mathbf{q}}^{tf(w,Cat(\mathbf{d}))>0} \log P_{cs}(w|\mathbf{d}) + \sum_{w \in \mathbf{q}}^{tf(w,Cat(\mathbf{d}))=0} \log P_{cu}(w|\mathbf{d}) \\ &= \sum_{w \in \mathbf{q}}^{tf(w,Cat(\mathbf{d}))>0} \log \frac{P_{cs}(w|\mathbf{d})}{P_{cu}(w|\mathbf{d})} + \sum_{w \in \mathbf{q}} \log P_{cu}(w|\mathbf{d}) \end{aligned} \tag{9}$$

According to the leaf smoothing model, we have:

$$\begin{aligned} P_{cs}(w|\mathbf{d}) &= (1-\lambda)P_{ml}(w|\mathbf{d}) + \lambda[(1-\beta)P_{ml}(w|Cat(\mathbf{d})) + \beta P_{ml}(w|Coll)] \\ P_{cu}(w|\mathbf{d}) &= \lambda\beta P_{ml}(w|Coll) \end{aligned} \tag{10}$$

From Equation 9 and the two equations in Equation 10, we get:

$$\log P(\mathbf{q}|\mathbf{d}) = \sum_{w \in \mathbf{q}}^{tf(w,Cat(\mathbf{d}))>0} \log\left(\frac{(1-\lambda)P_{ml}(w|\mathbf{d}) + \lambda(1-\beta)P_{ml}(w|Cat(\mathbf{d}))}{\lambda\beta P_{ml}(w|Coll)} + 1\right) + \sum_{w \in \mathbf{q}} \log \lambda\beta P_{ml}(w|Coll) \tag{11}$$

The second term on the right hand of Equation 11 is independent of $\mathbf{d}$ and thus can be ignored in ranking. The first term on the right hand side shows that for questions from different categories, the larger $P_{ml}(w|Cat(\mathbf{d}))$ gets, the larger $P(\mathbf{q}|\mathbf{d})$ gets, i.e., the more frequently a word w in query $\mathbf{q}$ occurs in a category, the higher relevancy scores the questions in the category will get. Hence, the category smoothing plays a role in differentiating questions from different categories.

We next show that category-specific frequent words are less important in comparing the relevancy of questions within the same category. As do Zhai and Lafferty [2004], we define a model $P_s(w|\mathbf{d})$ used for "seen" words that occur in a pre-existing question $\mathbf{d}$ (i.e., $tf(w, \mathbf{d}) > 0$), and a model $P_u(w|\mathbf{d})$ is used for "unseen" words that do not (i.e., $tf(w, \mathbf{d}) = 0$). The probability of a query $\mathbf{q}$ can be written as follows:

$$\begin{aligned} \log P(\mathbf{q}|\mathbf{d}) &= \sum_{w \in \mathbf{q}} \log P(w|\mathbf{d}) \\ &= \sum_{w \in \mathbf{q}}^{tf(w,\mathbf{d}))>0} \log P_s(w|\mathbf{d}) + \sum_{w \in \mathbf{q}}^{tf(w,\mathbf{d}))=0} \log P_u(w|\mathbf{d}) \\ &= \sum_{w \in \mathbf{q}}^{tf(w,\mathbf{d}))>0} \log \frac{P_s(w|\mathbf{d})}{P_u(w|\mathbf{d})} + \sum_{w \in \mathbf{q}} \log P_u(w|\mathbf{d}) \end{aligned} \tag{12}$$

In the leaf smoothing model, from Equation 6 we know:

$$\begin{aligned} P_s(w|\mathbf{d}) &= (1-\lambda)P_{ml}(w|\mathbf{d}) + \lambda[(1-\beta)P_{ml}(w|Cat(\mathbf{d})) + \beta P_{ml}(w|Coll)] \\ P_u(w|\mathbf{d}) &= \lambda[(1-\beta)P_{ml}(w|Cat(\mathbf{d})) + \beta P_{ml}(w|Coll)] \end{aligned} \tag{13}$$

From Equation 12 and the two equations in Equation 13, we get:

$$\log P(\mathbf{q}|\mathbf{d}) = \sum_{w \in \mathbf{q}}^{tf(w,\mathbf{d}))>0} \log\left(\frac{(1-\lambda)P_{ml}(w|\mathbf{d})}{\lambda[(1-\beta)P_{ml}(w|Cat(\mathbf{d})) + \beta P_{ml}(w|Coll)} + 1\right) + \sum_{w \in \mathbf{q}} \log(\lambda[(1-\beta)P_{ml}(w|Cat(\mathbf{d})) + \beta P_{ml}(w|Coll)) \tag{14}$$

We can see that for questions in the same category $Cat(\mathbf{d})$, the second term on the right hand side of Equation 14 is the same and thus will not affect their relative ranking. In contrast the first term in the right hand side is inversely proportional to the maximum likelihood estimate of word w in question $Cat(\mathbf{d})$, $P_{ml}(w|Cat(\mathbf{d}))$. Hence, the leaf smoothing plays a similar role as the well known IDF for questions in the same category. The more frequent a word occurs in a specific category, the less important it is for searching relevant questions in that category in this model.

To summarize, we find that on one hand, leaf category smoothing enables the questions in categories that are more relevant to the query to gain higher relevance scores; on the other hand, leaf category smoothing plays a similar role as IDF computed with regard to a leaf category.

This approach is inspired by the clustering based retrieval model $CBDM$ for document retrieval [Kurland and Lee 2004; Liu and Croft 2004]. However, previous work on cluster-based retrieval does not establish the above analysis that also provides insight into the cluster-based retrieval model.

4. APPROACH 2: CATEGORY ENHANCEMENT

We next present a general approach that can be easily applied to existing question retrieval models.

4.1. Overview

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 a 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: a global score capturing the relevance of the category $Cat(\mathbf{d})$ that contains $\mathbf{d}$ to query $\mathbf{q}$, and a local score of $\mathbf{d}$ to $\mathbf{q}$ 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 combine them using linear interpolation. The final ranking score $RS_{\mathbf{q},\mathbf{d}}$ is then computed as follows:

$$RS_{\mathbf{q},\mathbf{d}} = (1-\alpha)N(S_{\mathbf{q},\mathbf{d}}) + \alpha N(S_{\mathbf{q},Cat(\mathbf{d})}), \tag{15}$$

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, α controls the linear interpolation, and $N()$ is the normalization function for the local and global relevance scores (to be discussed in Section 7.1.7).

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. For example, a 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 are important for 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.

Existing retrieval models can be adopted to compute local and global relevance scores, upon which the final ranking score can be obtained using Equation 15. We proceed to detail how to compute the global and local relevance scores using different retrieval models.

4.2. Retrieval Models—Global Relevance

Historical CQA questions are categorized, and the global relevance reveals the relevance of the questions in a category as a whole to 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.

4.2.1. The Vector Space Model. The problem adapts the Vector Space Model (VSM) to compute the relevance score between a question and a category. We develop two methods.

First, we view a category as a concatenation of the questions it contains, i.e., the category is represented by one single pseudo document. Thus, the whole question collection is 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 accordingly. 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 method will perform poorly for two reasons. First, 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 15 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 $Cat(\mathbf{d})$), $w_{\mathbf{q},t}$ captures the IDF of word t with regard to categories, and $w_{Cat(\mathbf{d}),t}$ captures the TF (term frequency) of term t in $Cat(\mathbf{d})$..

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

We note that the existing pivoted document length normalization method [Singhal et al. 1996] 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 the notion of 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 $Cat(\mathbf{d})$, 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 question d_i contained in $Cat(\mathbf{d})$, and each element in the vector is computed by $w_{\mathbf{d},t}$ as stated 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 [Fang et al. 2004]. 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" [Fang et al. 2004]). 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 is weighted much higher than it is if we 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."

4.2.2. The 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 need to be modified to be 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:

$$\begin{aligned}
S_{\mathbf{q},Cat(\mathbf{d})} &= \sum_{t \subset \mathbf{q} \bigcap Cat(\mathbf{d})} w_{\mathbf{q},t} w_{Cat(\mathbf{d}),t}, \text{ where} \\
w_{\mathbf{q},t} &= \ln(\frac{M - fc_t + 0.5}{fc_t + 0.5})\frac{(k_3+1)tf_{t,\mathbf{q}}}{k_3 + tf_{t,\mathbf{q}}} \\
w_{Cat(\mathbf{d}),t} &= \frac{(k_1+1)tf_{t,Cat(\mathbf{d})}}{K_\mathbf{d} + tf_{t,Cat(\mathbf{d})}} \\
K_\mathbf{d} &= k_1((1-b) + b\frac{W_{Cat(\mathbf{d})}}{W_{A(cat)}})
\end{aligned} \tag{16}$$

Here M, fc_t, and $W_{Cat(\mathbf{d})}$ is as defined in Section 4.2.1, and $W_{A(cat)}$ is the average length of the concatenated category documents.

4.2.3. The Language, Translation, and Translation-Based Language Models. Since categories are viewed as pseudo documents, in these three models, we simply need to replace $\mathbf{d}$ with $Cat(\mathbf{d})$ to compute the global relevance scores using Equations 3, 7, and 8.

4.3. 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 by treating a category as the collection.

4.3.1. The Vector Space Model. As discussed briefly in Section 4.1, 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, which we call the 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:

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

4.3.2. The Okapi BM25 Model. Similar to the VSM, the Okapi Model also has an IDF component, i.e., $w_{\mathbf{q},t}$ as stated in Equation 2. We also compute the IDF with regard to the category containing the historical question, which we call the 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, the following modifications are made to Equation 2:

$$w_{\mathbf{q},t} = \ln(\frac{N_{Cat(\mathbf{d})} - f_{t,Cat(\mathbf{d})} + 0.5}{f_{t,Cat(\mathbf{d})} + 0.5})\frac{(k_3+1)tf_{t,\mathbf{q}}}{k_3 + tf_{t,\mathbf{q}}}$$
$$K_{\mathbf{d}} = k_1((1-b) + b\frac{W_{\mathbf{d}}}{W_{A,Cat(\mathbf{d})}})$$

4.3.3. The Language, Translation, and Translation-based Language Models. Unlike VSM and Okapi, the Language Model does not have an IDF component. However, the smoothing in the Language Model plays an IDF-like role [Zhai and Lafferty 2004]. 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, 7, and 8.

Note that when compared to the leaf category smoothing enhancement introduced in Section 3, the category smoothing is used differently and for a different purpose. Our first 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 solely, to distinguish questions within the same category.

5. APPROACH 3: QUERY CLASSIFICATION ENHANCEMENT

5.1. Query Classification Enhancement

One straightforward method of leveraging the classification of a query question $\mathbf{q}$ is as follows: first determine the category of the query using a classification model; then rank questions in this category using a certain retrieval model (such as the Language Model) to retrieve relevant questions. We can build a classification model using historical questions to classify a query question, and we can compute the probability $P(\mathbf{q}|\mathbf{d})$ using Equation 17. This probability score captures the relevance of a historical question $\mathbf{d}$ to query $\mathbf{q}$.

$$P(\mathbf{q}|\mathbf{d}) = \begin{cases} P(\mathbf{q}|\mathbf{d}, Cat(\mathbf{d})), CLS(\mathbf{q}) = Cat(\mathbf{d}) \\ 0, CLS(\mathbf{q}) \neq Cat(\mathbf{d}) \end{cases} \tag{17}$$

where $P(\mathbf{q}|\mathbf{d}, Cat(\mathbf{d}))$ represents how likely it is that query $\mathbf{q}$ was generated from the historical question $\mathbf{d}$ with regard to the category $Cat(\mathbf{d})$, and $CLS(\mathbf{q})$ denotes the category of the query $\mathbf{q}$ determined by the classifier.

This simple approach may greatly improve the efficiency of question search since it can prune the search space by limiting search to a single category. The number of questions in a leaf category usually does not exceed 5% of all questions; thus searching in a category will be much more efficient than searching in the whole collection. However, as will be shown in Section 7.2.2, this approach is not good in terms of effectiveness even if we assume that perfect classification results can be achieved. This is because not all the relevant questions come from the same category . In addition, the effectiveness of question search will depend highly on the accuracy of the classifier used: if the query question is not classified correctly, then the retrieval results will be poor since a wrong category is searched.

To alleviate the aforementioned problems, we exploit the same idea as for category enhancement in Section 4.1. We compute the probability of query $\mathbf{q}$ belonging to the category $Cat(\mathbf{d})$ of a historical question $\mathbf{d}$, which measures the relevance of $Cat(\mathbf{d})$ to $\mathbf{q}$. The probability is denoted by $P(Cat(\mathbf{d})|\mathbf{q})$. According to Equation 17, under the condition $CLS(\mathbf{q}) = Cat(\mathbf{d})$ we have $P(\mathbf{q}|\mathbf{d}) = P(\mathbf{q}|\mathbf{d}, Cat(\mathbf{d}))$, and under the condition $CLS(\mathbf{q}) \neq Cat(\mathbf{d})$ we have $P(\mathbf{q}|\mathbf{d}) = 0$. In fact, $P(Cat(\mathbf{d})|\mathbf{q})$ represents the probability of $CLS(\mathbf{q}) = Cat(\mathbf{d})$. Finally, according to the total probability theorem, we estimate $P(\mathbf{q}|\mathbf{d})$ as follows:

$$\begin{aligned} P(\mathbf{q}|\mathbf{d}) &= P(\mathbf{q}|\mathbf{d}, Cat(\mathbf{d})) \cdot P(CLS(\mathbf{q}) = Cat(\mathbf{d})) + 0 \cdot P(CLS(\mathbf{q}) \neq Cat(\mathbf{d})) \\ &= P(\mathbf{q}|\mathbf{d}, Cat(\mathbf{d}))P(Cat(\mathbf{d})|\mathbf{q}) \end{aligned} \tag{18}$$

where $P(Cat(\mathbf{d})|\mathbf{q})$ is computed by a classification model (to be discussed in Section 5.3). For a query question, the classification model can return the probability of the query belonging to a category.

Equation 18 suggests a way to rank questions by incorporating the query classification probability into the models that compute probabilistic scores, i.e., LM, TR, and TRLM. The ranking of a historical question $\mathbf{d}$ will be promoted if the probability of the query question $\mathbf{q}$ belonging to the category $Cat(\mathbf{d})$ is high.

We also apply this approach to VSM and Okapi, which do not compute probabilistic scores. In these cases, the ranking scores can be viewed as the weighted averages of the two situations, $CLS(\mathbf{q}) = Cat(\mathbf{d})$ and $CLS(\mathbf{q}) \neq Cat(\mathbf{d})$. As shown in Section 7.2.2, the performance is also better. However, the improvement is not as large as when applying this enhancement to the three probabilistic models.

5.2. Searching Space Pruning Using Query Classification

Efficiency is important in question search since CQA archives contain large and growing volumes of question-answer pairs. [1]

The results of query question classification are used to distinguish historical questions across different categories to improve the performance in the query classification enhanced models. Query classification can also help prune the question search space and thus improve runtime efficiency. To achieve this, we introduce a threshold ξ and prune the categories if the probability of $\mathbf{q}$ belonging to them is below ξ.

We note that this pruning might decrease the effectiveness of question search while improving the runtime. The trade-off between effectiveness and efficiency is evaluated in Section 7.2.3.

5.3. Classification Model

Hierarchical classification has been shown to be more efficient and usually more effective than flat classification [Dumais and Chen 2000]. We thus use a top-down hierarchical classification approach [Dumais and Chen 2000] to build a hierarchical classification model for classifying new questions. The top-down approach first learns to distinguish among categories at the top level, upon which lower level distinctions are learned within the appropriate top level of the tree. Given a query $\mathbf{q}$ to be classified, the approach traverses a classification tree, starting from the root. At each node, it check whether the probability of $\mathbf{q}$ belonging to the category at the node is larger than a threshold ζ: if so, it assigns a probability to each category (child node) in the current node; otherwise, the subtree rooted at the node is not traversed.

Given a query $\mathbf{q}$ and a category Cat, the probability of $\mathbf{q}$ belonging to category Cat is defined as $P(Cat|\mathbf{q})$, and it is computed as follows:

$$P(Cat|\mathbf{q}) = \prod_{c_i \in Path(Cat)} P(\mathbf{q}|c_i),$$

where $Path(Cat)$ refers to the path from category Cat to the root in the classification tree, which satisfies $P(\mathbf{q}|c_i) > \zeta$, for all c_i belonging to $Path(Cat)$.

[1]The Community QA in Baidu ZhiDao has more than 121 million resolved questions as of Feb. 28 2011

Some leaf categories might not be traversed due to the threshold ζ. In the query classification enhanced models, we assign untraversed categories the probability of their nearest ancestor that has been traversed.

The classification task is essentially a text classification problem, and using bags of words as features works well in text classification [Sebastiani 2002]. Hence, we treat each question as a bag of words. Substantial work on question classification in Question Answering has been reported, e.g., the work of Voorhees [2001], which classifies questions (mainly factoid questions) into a number of categories mainly based on expected answer types, e.g., a number category (the expected answer is a number). These approaches could also be used for the classification task in our work. However, the classification is not the topic of this paper.

6. APPROACH 4: QUESTION CLASSIFICATION ENHANCEMENT

6.1. Question Classification Enhancement Retrieval

In the previous approaches, when computing the ranking scores with regard to a category, we consider only $Cat(\mathbf{d})$, i.e., the category containing the question $\mathbf{d}$. This means that we trust the question categorization totally. However, it is quite possible that a user submits a question to a non-relevant category by mistake. Another problem is that some categories overlap from a semantic perspective. For example, the category "Travel.United States.Los Angeles" may contain questions about restaurant recommendations, which is the main topic of the category "Dining Out.United States.Los Angeles." If we only use the category containing $\mathbf{d}$, the retrieval models might not perform well when a relevant question is submitted to a non-relevant category or when a relevant question may belong to several semantically overlapping categories.

In order to avoid the negative effects of the two aforementioned problems, we propose to incorporate historical question classification probability scores into existing retrieval models. We use the same classifier as in Section 5.3 to compute the question classification probability scores. Specifically, given a historical question $\mathbf{d}$, we compute the probability $P(cat_i|\mathbf{d})$ where cat_i is a leaf category and i ranges from 1 to the number of leaf categories.

According to the total probability theorem, we can compute the probability $P(\mathbf{q}|\mathbf{d})$ as follows:

$$P(\mathbf{q}|\mathbf{d}) = \sum_i P(\mathbf{q}|\mathbf{d}, cat_i)P(cat_i|\mathbf{d}), \tag{19}$$

where $P(cat_i|\mathbf{d})$ is the probability returned by the classifier and $P(\mathbf{q}|\mathbf{d}, cat_i)$ is the probabilistic ranking score computed with regard to the category cat_i. To be specific, in LM, TR, and TRLM, we do the smoothing on the category cat_i instead of the whole collection, i.e., replacing $P_{ml}(w|Coll)$ with $P_{ml}(w|cat_i)$ in Equations 3, 4, and 5.

The effectiveness of this approach is adversely affected by the classification errors. The classifier is also not totally reliable. Therefore, we make a compromise between the decision of users and the classifier. Thus, we use the following method to compute the ranking score of $\mathbf{d}$:

$$P(\mathbf{q}|\mathbf{d}) = \begin{cases} P(\mathbf{q}|\mathbf{d}, Cat(\mathbf{d})), & CLS(\mathbf{d}) = Cat(\mathbf{d}) \\ \tau P(\mathbf{q}|\mathbf{d}, Cat(\mathbf{d})) + (1-\tau)\dfrac{1}{\sum_{i=1}^{k} P(cat_i|\mathbf{d})} & \\ \qquad\qquad \displaystyle\sum_{i=1}^{k} P(\mathbf{q}|\mathbf{d}, cat_i)P(cat_i|\mathbf{d}), & CLS(\mathbf{d}) \neq Cat(\mathbf{d}) \end{cases} \tag{20}$$

If the category label of a question $\mathbf{d}$ is consistent with the classification result, we believe that it is submitted correctly, and we use the probability $P(\mathbf{q}|\mathbf{d}, Cat(\mathbf{d}))$ as the ranking score. Otherwise, we use the categories specified by both the users and the classifier to compute the final ranking score. $CLS(\mathbf{d})$ denotes the top-1 category returned by the classifier. Parameter τ is used to tune the relative importance of the score computed with regard to the category given by the user and the score computed with regard to the categories returned by the classifier. In fact, parameter τ represents how much we believe in the category label of Yahoo! Answers when it contradicts the classifier. Finally, parameter k is the number of categories returned by the classifier that will be used. It makes sense to use only categories with relative large probability scores.

This approach is based on the total probability theorem. Therefore, it can only be applied to the models that compute probability scores, i.e., LM, TR, and TRLM.

6.2. Incorporation of Leaf Category Smoothing Enhancement

Leaf category smoothing enhancement can also take advantage of question classification enhancement. Specifically, in Equation 20, we compute both $P(\mathbf{q}|\mathbf{d}, Cat(\mathbf{d}))$ and $P(\mathbf{q}|\mathbf{d}, cat_i)$ using leaf category smoothing as applied to LM, TR, and TRLM with regard to $Cat(\mathbf{d})$ and cat_i, respectively. Taking LM with leaf category smoothing as an example, they are computed as follows:

$$P(\mathbf{q}|\mathbf{d}, Cat(\mathbf{d})) = \prod_{t\in\mathbf{q}}(1-\lambda)P_{ml}(w|\mathbf{d}) + \lambda[(1-\beta)P_{ml}(w|Cat(\mathbf{d})) + \beta P_{ml}(w|Coll)]$$

$$P(\mathbf{q}|\mathbf{d}, cat_i) = \prod_{t\in\mathbf{q}}(1-\lambda)P_{ml}(w|\mathbf{d}) + \lambda[(1-\beta)P_{ml}(w|cat_i) + \beta P_{ml}(w|Coll)]$$

Then, we compute the final ranking score for a question $\mathbf{d}$ as Equation 20.

The experimental results show that when combining leaf category smoothing with the use of question classification probabilities, the performance is better than when simply using the leaf category smoothing.

6.3. Incorporation of Category Enhancement

The category enhancement is also able to benefit from question classification enhancement. Considering the category enhanced approach with both global and local relevance scores as computed by LM as an example, we have:

$$\begin{aligned} P(\mathbf{q}|\mathbf{d}, Cat(\mathbf{d})) = \ & (1-\alpha)\prod_{t\in\mathbf{q}}((1-\lambda)P_{ml}(w|\mathbf{d}) + \lambda P_{ml}(w|Cat(\mathbf{d}))) \\ & + \alpha\prod_{t\in\mathbf{q}}((1-\lambda)P_{ml}(w|Cat(\mathbf{d})) + \lambda P_{ml}(w|Coll)) \end{aligned}$$

$$\begin{aligned} P(\mathbf{q}|\mathbf{d}, cat_i) = \ & (1-\alpha)\prod_{t\in\mathbf{q}}((1-\lambda)P_{ml}(w|\mathbf{d}) + \lambda P_{ml}(w|cat_i)) \\ & + \alpha\prod_{t\in\mathbf{q}}((1-\lambda)P_{ml}(w|cat_i) + \lambda P_{ml}(w|Coll)) \end{aligned}$$

$P(\mathbf{q}|\mathbf{d}, Cat(\mathbf{d}))$ is computed as the category enhancement approach applied to model LM. $P(\mathbf{q}|\mathbf{d}, cat_i)$ is computed as the category enhancement approach applied to LM assuming that cat_i is the correct category $\mathbf{d}$ belongs to. They are used in Equation 20 to compute the final ranking scores.

The experimental results show that this combined approach is more effective than simply using category enhancement by itself.

6.4. Incorporation of Query Classification Enhancement

Finally, the query classification enhancement is able to benefit from question classification enhancement. In Equation 20, $P(\mathbf{q}|\mathbf{d}, Cat(\mathbf{d}))$ and $P(\mathbf{q}|\mathbf{d}, cat_i)$ are computed as the query classification enhancement applied to LM, TR, and TRLM with regard to $Cat(\mathbf{d}))$ and cat_i, respectively. Using again LM for computing local relevance scores as an example, we have:

$$P(\mathbf{q}|\mathbf{d}, Cat(\mathbf{d})) = P(Cat(\mathbf{d})|\mathbf{q}) \prod_{t \in \mathbf{q}} ((1-\lambda)P_{ml}(w|\mathbf{d}) + \lambda P_{ml}(w|Cat(\mathbf{d})))$$

$$P(\mathbf{q}|\mathbf{d}, cat_i) = P(cat_i|\mathbf{q}) \prod_{t \in \mathbf{q}} ((1-\lambda)P_{ml}(w|\mathbf{d}) + \lambda P_{ml}(w|cat_i))$$

All the parameters are as defined earlier.

7. EVALUATION

We study the abilities of the four proposed approaches utilizing category information, namely, LS(Section 3), CE(Section 4), QC(Section 5), and DC(Section 6), to improve question retrieval in this section. Section 7.1 describes the setup of experiments, Section 7.2 presents the experimental results, Section 7.3 presents the analysis of the experimental results, and Section 7.4 offers a summary of experimental results.

7.1. Experimental Setup

7.1.1. Classification Model. We employ the hierarchical top-down classification method using the approach of Dumais and Chen [2000], and set the threshold ζ to 0.01.

7.1.2. Question Search Models. We evaluate the following models:

— The five baseline models: the Vector Space Model (VSM), the Okapi Model (Okapi), the Language Model (LM), the Translation Model (TR) and the Translation-based Language Model (TRLM).
— LS: Leaf category smoothing enhancement applied to LM, TR, and TRLM, yielding LM@LS, TR@LS, and TRLM@LS, respectively.
— CE: Category enhancement. The five baseline models can be used to compute both the global and the local relevance scores, yielding 25 combinations. Each combination is given by the name of the model used for computing the global score "+" the name of the model used for computing the local score. For example, if the global score is computed by LM and the local score is computed by VSM, the combination is named LM+VSM.
— QC: Query classification enhancement applied to the five baseline models: VSM@QC, Okapi@QC, LM@QC, TR@QC, and TRLM@QC.
— DC: Question classification enhancement applied to three probabilistic baseline models: LM@DC, TR@DC, and TRLM@DC. We also extend the question classification enhanced approach with LS, CE and QC. Using LM as an example, the combined models are denoted as LM@LS@DC, LM+LM@DC and LM@QC@DC, as explained in Section 6.2, 6.3 and 6.4, respectively.
— The two models that are briefly mentioned in Section 5, namely search in the Top-1 category determined by the classifier (Top1C) and search in the correct category specified by the users of Yahoo! Answers (OptC; the query set is collected from Yahoo! Answers, and thus the categories of queries are known).

Note that the baseline models are reported as references and the main purpose of the study is to *gain detailed insight into the circumstances under which and the degrees to which the different category-based approaches are able to improve on the effectiveness and runtime performance of these approaches.*

Figure 2 describes how our approaches can be combined and applied to the five baseline models. LS, QC, and DC can be applied to models computing probabilistic scores, i.e., LM, TR, and TRLM. We also apply QC to models VSM and Okapi in the experiments. CE exploits all the five baseline models to compute both the global and local relevance scores. In addition, DC can be incorporated with LS, CE, and QC to further improve the performance, as shown by the dashed lines.

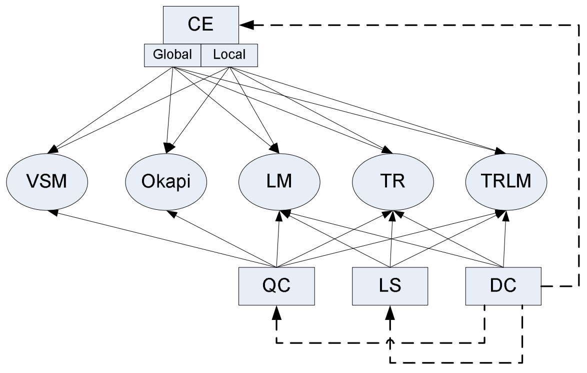

Fig. 2. Combinations of the models proposed

Table I. Number of questions in each first-level category

Category	Question#	Category	Question#
Arts & Humanities	114,737	Health	183,181
Beauty & Style	49,532	Home & Garden	50,773
Business & Finance	154,714	Local Businesses	69,581
Cars & Transportation	208,363	News & Events	27,884
Computers & Internet	129,472	Pets	72,265
Consumer Electronics	126,253	Politics & Government	85,392
Dining Out	58,980	Pregnancy & Parenting	63,228
Education & Reference	107,337	Science & Mathematics	116,047
Entertainment & Music	196,100	Social Science	61,011
Environment	28,476	Society & Culture	122,358
Family & Relationships	53,687	Sports	275,893
Food & Drink	55,955	Travel	403,926
Games & Recreation	72,634	Yahoo! Products	228,368

7.1.3. Dataset. We collected questions from all categories in Yahoo! Answers and then divided the questions randomly into two data sets. The division maintains the distributions of questions in all categories. We obtain a data set containing 3,116,147 questions as the training data for classification and also a *question repository* for question search. The other data set contains 127,202 questions and is used as the test set for classification. We use 1 million question-description pairs from another data set[2] for training the word translation probabilities. Note that we use a much larger question repository than those used in previous work for question search; a large, real data set is expected to better reflect the application scenario of real CQA services. This also explains why the training data is larger than the test data: training data should come from the question repository used for question search, and we would like to use most of the data for question search. Table I shows the question distribution across first-level categories in the training data set. There are 26 categories at the first level and 1263 categories at the leaf level. Each question belongs to a unique leaf-level category.

We randomly select 300 questions from the test set (127,202 questions) to evaluate the performance of retrieval. We remove the stop words. For each model, the top 20 retrieval results are kept. We put all the results from different models for one query question together for human annotation. Thus annotators do not know which results are from which model. Annotators are asked to label each returned question with "relevant" or "irrelevant." Two annotators are involved in the annotation process. If conflicts happen, a third annotator determines the final result. We eliminate the

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

query questions that have no relevant questions. The procedure yields 252 queries that have relevant questions and that are used as the *query set*.[3]

7.1.4. Metrics. We evaluate the performance of our approaches using Mean Average Precision (MAP), Mean reciprocal rank (MRR), R-Precision, and Precision@n as metrics. MAP rewards approaches returning relevant questions earlier and also emphases the ranking in the returned lists. MRR captures how far down we in a ranked list a relevant question can first be found. R-Precision is the precision after R questions have been retrieved, where R is the number of relevant questions for the query. Precision@n is the fraction of the Top-**n** questions retrieved that are relevant. 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 the metrics that we used. Finally, we use the tool *trec_eval* from TREC[4] to compute the different metrics.

We measure the performance of the classifier using the Micro-F1 score, which is appropriate in the presence of large-scale categories. The Micro-F1 score is calculated by averaging the F1 scores [5] over all decisions.

7.1.5. Parameter Settings. The experiments use five parameters that we aim to set so that all approaches perform at their best. The first is the smoothing parameter λ in LM, TR, and TRLM; the parameter η controls the self-translation impact in TRLM; the parameter β is the leaf category smoothing parameter; the parameter α is used for linear interpolation in the CE approach; the parameter τ is used in the question classification enhanced approach, which balances the trust in user-supplied classifications and classification results.

To find appropriate settings for parameters λ and β, we conduct a small-scale experiment, where we use a small data set containing 20 queries (this set is also extracted from the Yahoo! Answers data, and it is not included in the test *query set*). Table II shows the MAP results using LM@LS. As a result, we set λ to 0.2 and β to 0.2. Zhai and Lafferty [2004] also find that 0.2 is a good choice for parameter λ. In Okapi, we follow Robertson et al. [1994b] and set parameters k_1, b, and k_3 to 1.2, 0.75, and ∞, respectively. In TRLM, we set β to 0.8 following the literature of Xue et al. [2008]. For the parameter α, different combinations of models computing global and local relevance scores need different settings. We do an experiment on the small data set to determine the best value among $0.1, 0.2, 0.3, ..., 0.9$ in terms of MAP for each combination in the CE approach. This leads to the setting $\alpha = 0.1$ for the combinations that use LM, TR, and TRLM for computing local relevance. When computing local relevance using VSM and Okapi, and computing the global relevance using Okapi, the value of α is set to 0.7 and 0.5, respectively; otherwise, $\alpha = 0.9$. The best result is achieved when $\tau = 0.5$ and $k = 3$ on the small data set using model LM@DC. We thus use $\tau = 0.5$ and $k = 3$ in the experiments with the DC approach.

Table II. Parameter Selection (MAP)

β \ λ	0.1	0.2	0.3
0.1	0.4364	0.4437	0.4359
0.2	0.4470	0.4512	0.4320
0.3	0.4385	0.4419	0.4172

7.1.6. Learning Word Translation Probabilities. The performance of the Translation Model and the Translation-based Language Model rely on the quality of the word-to-word translation probabilities. We follow the approach of Xue et al. [2008] to learn the word translation probabilities. In our experiments, question-description pairs are used for training, and the GIZA++[6] toolkit is used to

[3]The query set and the relevant annotations are available in `http://homepages.inf.ed.ac.uk/gcong/qa/`.
[4]`http://trec.nist.gov/`
[5]`http://en.wikipedia.org/wiki/F_score`.
[6]`http://www.fjoch.com/GIZA++.html`.

learn the IBM translation model 1. There are two ways of accomplishing the training: the word translation probabilities 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. [2008] 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 then use this combined set of pairs for learning the word-to-word translation probabilities. We learn the word translation probabilities using this method.

7.1.7. 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 4.2 and 4.3. We need to normalize them into approximately identical ranges before combining linearly 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 III—for each model.

Table III. Approximate value range for each model

	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	$(10^{-80}, 10^{-2})$	$(10^{-50}, 10^{-2})$

We use the simple min-max normalization method to scale the values into the range [0,1]. 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 logarithmic transformation of the original relevance values): $N(S) = \frac{S - S_{min}}{S_{max} - S_{min}}$, where S denotes the original score, S_{max} denotes the maximum value, S_{min} denotes the minimum value, and $N(S)$ is the normalized value.

7.2. Experimental Findings

In this section, we report results for the different question search approaches using different metrics. We also study the effect on both efficiency and effectiveness of pruning the question search space.

7.2.1. Classification Results. The classification results provide indispensable information for the QC approach described in Section 5. The Micro-F1 scores of our flat classification model is 44.84.

The QC approach uses not only the Top-1 returned category, but also other returned categories. Even if the correct category is not the Top-1 returned category, this question search approach can still benefit from classification information if the correct category is contained in the Top-**n** returned categories. In order to see if the correct category is contained in the Top-**n** returned categories, we compute the percentage of test query questions whose correct categories are contained in the Top-**n** categories returned by the classifier. We call the percentage "Success@n." The correct categories of about 75% of all questions are in the Top-10 categories returned by the classifier.

7.2.2. Question Search Results. Table IV shows the results of VSM@LS, VSM@QC, the CE approaches, i.e., five combinations that use VSM for local relevance computation (Section 4.3.1) and use the five models introduced in Section 2 (VSM, Okapi, LM, TR, and TRLM) for global relevance computation (Section 4.2). The baseline method in this table is VSM. The results of VSM Top1C (searching in the category determined by the classifier) and VSM OptC (searching in the category containing the query question) are also given. We can see that all our three approaches

Table IV. Results of approaches applied to VSM (**%chg** denotes the improvement in percent of each model; * indicates a statistically significant improvement over the baseline using the t-test, p-value $<$ 0.05)

	VSM	**Top1C**	**VSM@QC**	**%chg**	**VSM+VSM**	**%chg**	**Okapi+VSM**	**%chg**
MAP	0.2407	0.2231	0.3075	27.8*	**0.3711**	54.2*	0.3299	37.1*
MRR	0.4453	0.4234	0.4943	11.0*	**0.5637**	26.6*	0.5314	19.3*
R-Prec	0.2311	0.2056	0.2937	27.1*	**0.3419**	48.0*	0.3094	33.9*
P@5	0.2222	0.2002	0.2501	12.6*	**0.2789**	25.5*	0.2559	15.2*
		OptC	**LM+VSM**	**%chg**	**TR+VSM**	**%chg**	**TRLM+VSM**	**%chg**
MAP		0.2414	0.3632	50.9*	0.3629	50.8*	0.3628	50.7*
MRR		0.4534	0.5596	25.7*	0.5569	25.1*	0.5585	25.4*
R-Prec		0.2298	0.3366	45.7*	0.3346	44.8*	0.3357	45.3*
P@5		0.2289	0.2746	23.6*	0.2746	23.6*	0.2753	23.9*

Table V. Results of approaches applied to Okapi (**%chg** denotes the improvement in percent of each model; * indicates a statistically significant improvement over the baseline using the t-test, p-value $<$ 0.05)

	Okapi	**Top1C**	**Okapi@QC**	**%chg**	**VSM+Okapi**	**%chg**	**Okapi+Okapi**	**%chg**
MAP	0.3401	0.2637	0.3622	6.5	0.4007	17.8*	0.3977	16.9*
MRR	0.5406	0.4502	0.5713	5.6	0.6131	13.4*	0.5884	8.8
R-Prec	0.3178	0.2325	0.3345	5.3	0.3648	14.8*	0.3613	13.7*
P@5	0.2857	0.2589	0.2998	4.9	0.3140	9.9*	**0.3176**	11.2*
		OptC	**LM+Okapi**	**%chg**	**TR+Okapi**	**%chg**	**TRLM+Okapi**	**%chg**
MAP		0.2862	**0.4138**	21.7*	0.4082	20.0*	0.4132	21.5*
MRR		0.4887	0.6214	15.0*	0.6172	14.2*	**0.6215**	15.0*
R-Prec		0.2625	0.3758	18.3*	0.3677	15.7*	**0.3762**	18.4*
P@5		0.2824	0.3161	10.6*	0.3111	8.8	0.3147	10.2*

of utilizing category information are able to significantly improve the performance of the baseline VSM. We discuss the reasons for the improvement in the next section.

Similarly, Table V shows the results of the LS, QC, and CE approaches applied to Okapi. The results of Okapi Top1C and Okapi OptC are also given in this table. Next, Table VI shows the results of the baseline LM, LM Top1C, LM OptC, and the four approaches applied to LM. Similarly, Table VII shows the results of the baseline TR, TR Top1C, TR OptC, and the four approaches applied to TR. Table VIII shows the results of the baseline TRLM, TRLM Top1C, TRLM OptC, and the four approaches applied to TRLM.

Finally Table XVI shows the results of the LS, CE, and QC approaches when incorporated with DC. We can see that they gain better performance in terms of all the four metrics.

Observations similar to those made for Table IV can be made for Tables V–VIII: All the approaches utilizing category information consistently outperform the corresponding baseline methods. This clearly shows that the category information indeed can be utilized to improve question retrieval, and it shows that our four proposed approaches are 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 [Jeon et al. 2005b; Xue et al. 2008].

In the CE approach, 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 local relevance. The underlying reasons are studied in the next section.

The Top1C and 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 does the query question. In addition, the Top1C approach is affected by classification errors and thus has the worst results. Table IX and Table XIII show that searching in only the top categories results in many relevant questions being missed. This is because not all the relevant questions come from one category.

Table VI. Results of approaches applied to LM (**%chg** denotes the improvement in percent of each model; * indicates a statistically significant improvement over the baseline using the t-test, p-value < 0.05)

	LM	**Top1C**	**OptC**	**LM@LS**	**%chg**	**LM@DC**	**%chg**
MAP	0.3821	0.3188	0.3402	0.4586	20.9*	0.4548	19.0*
MRR	0.5945	0.4992	0.5219	0.6620	11.4*	0.6640	11.7*
R-Prec	0.3404	0.2989	0.3129	0.4072	19.6*	0.4107	20.7*
P@5	0.3040	0.2552	0.2810	0.3460	13.8*	0.3473	14.2*
		LM@QC	**%chg**	**VSM+LM**	**%chg**	**Okapi+LM**	**%chg**
MAP		0.4571	19.6*	**0.4620**	20.9*	0.4599	20.4*
MRR		**0.6716**	13.0*	0.6630	11.5*	0.6651	11.9*
R-Prec		**0.4151**	21.9*	0.4101	20.5*	0.4079	19.8*
P@5		0.3514	15.5*	0.3512	15.5*	0.3498	15.1*
		LM+LM	**%chg**	**TR+LM**	**%chg**	**TRLM+LM**	**%chg**
MAP		0.4609	20.6*	0.4603	20.5*	0.4616	20.8*
MRR		0.6622	11.4*	0.6633	11.6*	0.6667	12.1*
R-Prec		0.4087	20.1*	0.4087	20.1*	0.4100	20.4*
P@5		**0.3519**	15.8*	0.3512	15.5*	0.3513	15.6*

Table VII. Results of approaches applied to TR (**%chg** denotes the improvement in percent of each model; * indicates a statistically significant improvement over the baseline using the t-test, p-value < 0.05)

	TR	**Top1C**	**OptC**	**TR@LS**	**%chg**	**TR@DC**	**%chg**
MAP	0.4010	0.3202	0.3417	0.4563	13.8*	0.4564	13.8*
MRR	0.6084	0.5088	0.5392	0.6652	8.9	0.6627	8.9
R-Prec	0.3717	0.2991	0.3175	0.4177	12.4*	0.4274	15.0*
P@5	0.3168	0.2402	0.2670	0.3495	10.3*	0.3501	10.5*
		TR@QC	**%chg**	**VSM+TR**	**%chg**	**Okapi+TR**	**%chg**
MAP		**0.4613**	15.0*	0.4592	14.5*	0.4528	12.9*
MRR		**0.6702**	10.2*	0.6607	8.6	0.6532	7.4
R-Prec		**0.4313**	16.0*	0.4153	11.7*	0.4079	9.7*
P@5		**0.3534**	11.6*	0.3505	10.6*	0.3519	11.1*
		LM+TR	**%chg**	**TR+TR**	**%chg**	**TRLM+TR**	**%chg**
MAP		0.4507	12.4*	0.4526	12.9*	0.4522	12.8*
MRR		0.6527	7.3	0.6552	7.7	0.6540	7.5
R-Prec		0.4054	9.1	0.4071	9.5*	0.4058	9.2
P@5		0.3505	10.6*	0.3497	10.4*	0.3490	10.2*

Table VIII. Results of approaches applied to TRLM (**%chg** denotes the improvement in percent of each model; * indicates a statistically significant improvement over the baseline using the t-test, p-value < 0.05)

	TRLM	**Top1C**	**OptC**	**TRLM@LS**	**%chg**	**TRLM@DC**	**%chg**
MAP	0.4369	0.3323	0.3645	0.4797	9.8*	0.4794	9.7
MRR	0.6316	0.5215	0.5506	0.6706	7.3	0.6659	5.4
R-Prec	0.4008	0.3221	0.3474	0.4408	10.0*	**0.4414**	10.1
P@5	0.3398	0.2667	0.2910	0.3522	3.6	0.3524	3.7
		TRLM@QC	**%chg**	**VSM+TRLM**	**%chg**	**Okapi+TRLM**	**%chg**
MAP		0.4832	10.6*	**0.4937**	13.0*	0.4823	10.4*
MRR		**0.6717**	6.3	0.6704	6.1	0.6652	5.3
R-Prec		0.4410	10.0	0.4407	10.0	0.4349	8.5
P@5		0.3567	5.0	**0.3570**	5.1	0.3556	4.6
		LM+TRLM	**%chg**	**TR+TRLM**	**%chg**	**TRLM+TRLM**	**%chg**
MAP		0.4836	10.7*	0.4876	11.6*	0.4886	11.8*
MRR		0.6675	5.6	0.6685	5.8	0.6678	5.7
R-Prec		0.4319	7.8	0.4331	8.1	0.4343	8.4
P@5		0.3541	4.2	0.3548	4.4	0.3527	3.8

7.2.3. Search Space Pruning. In addition to retrieval effectiveness, runtime efficiency is also very important, considering that CQA archives are large and growing. In Section 5.2 we proposed a method that utilizes query classification to improve runtime efficiency by introducing a threshold ξ. We prune categories that are less likely to contain relevant questions based on the probability scores returned by the classifier. This can greatly save the runtime of question search, though it might also degrade the effectiveness of question search. Note that the classification time is very low compared with the question search time.

We next evaluate the effect of threshold pruning using the models LM, LM@LS, LM@QC, LM@DC, and LM+LM (our four different approaches of utilizing category information to improve question retrieval) with respect to both effectiveness and runtime efficiency. Considering the probability value of the leaf categories returned by the classifier, we vary pruning threshold ξ from 10^{-7} to 10^{-1}. For all the four models, the runtime speed-up compared to that of the baseline model LM is shown in Figure 3, and the MAP results are shown in Figure 4.

From Figures 3 and 4, we can see that the pruning adversely affects the retrieval effectiveness while it can greatly improve the efficiency for all four models. For example, the saving is about 85% when threshold ξ is set at 0.1 for all models. As the threshold becomes smaller, the MAP results become better while the runtime savings decrease. However, even when the threshold ξ is set at 10^{-7}, for the four models, the runtime is still about 40% of the original runtime of the baseline, while the MAP value is around 0.45. LM@LS, LM@QC, LM@DC and LM+LM with pruning still achieve significant improvements over LM without pruning in terms of MAP, MRR, R-Precision, and P5.

The baseline model LM is the most efficient, and LM@LS, LM@QC, LM@DC, and LM+LM are a bit slower than LM because they need additional processing to utilize category information. However, these four models have better performance.

We find that threshold-based pruning enables us to choose different trade-offs between effectiveness and efficiency. When efficiency is not a concern, we can simply ignore the parameter.

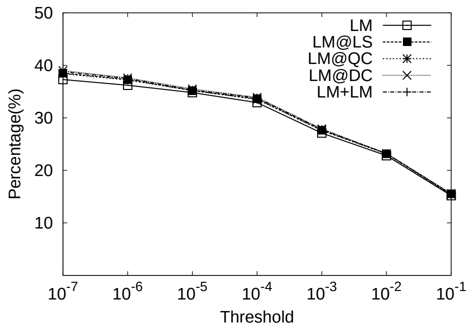

Fig. 3. Relative running time of different models using different pruning thresholds compared with the baseline model LM

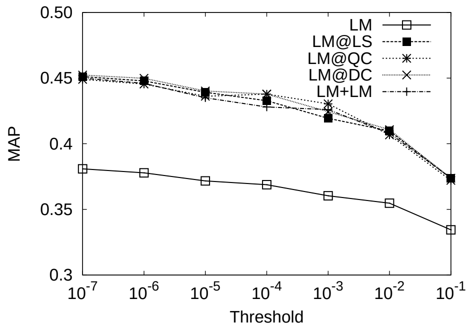

Fig. 4. Performance on MAP using different pruning thresholds compared with the baseline model LM

As a summary, the experimental results show that the category information can be utilized to significantly improve the effectiveness of question search in the proposed approaches. Moreover, we can trade improved runtime performance for (some of the gained) query effectiveness by using query classification based search space pruning.

7.3. Result Analysis

We proceed to scrutinize the results to understand the effects of the use of category information in our approaches. In this section, we determine whether our approaches outperform the baselines for a specific query based on the metric MAP. The analyses based on other metrics are qualitatively comparable.

7.3.1. Result Analysis of Leaf Category Smoothing Enhancement. As discussed in Section 3.1, the LS model improves on the baseline models in two aspects (we use LM@LS as our example in the following analysis. Note that it is also valid for TR@LS and TRLM@LS): First, leaf smoothing in the LS model enables the category-specific frequent words to play a more important role in distinguishing questions across categories. It is effective in promoting the rankings of historical questions from categories that are relevant to a query. We find that about 90 queries (out of 252) benefit from this. Second, leaf smoothing makes the rankings of questions from the same category more reasonable since it plays an IDF-like role in distinguishing questions within a specific category. About 85 queries benefit from this. The performance on some queries is improved due to both improvements.

We also notice that LM@LS performs worse than LM on 45 queries. We investigate these cases and find the following reasons (the performance of some queries is affected by more than one reason).

1) Relevant questions come from the categories whose topics are not very relevant to query questions. The LS model demotes questions from "non-relevant categories"; thus, if those categories contain relevant questions, the performance becomes worse than the baselines. One reason for the problem is that a question may be submitted to a wrong category by the user. Another reason is that many categories have an "Other" subcategory that may contain relevant questions, but such categories cover diverse topics and are not quite relevant to the queries, and thus LM@LS fails. We find that about 10 queries are affected by this.

2) The overlapping of categories leads to worse performance of LS. Some queries may be contained in multiple categories. For example, "How many hours from portland to hawaii air time?" is relevant to both "Travel.United States.Honolulu" and "Travel.United States.Portland." In our data set, the relevant question is contained in "Travel.United States.Portland," but LM@LS ranks questions from "Travel.United States.Honolulu" higher. About 15 queries are affected by this.

3) Although the leaf smoothing usually helps to when ranking questions from the same category, it leads to poor performance for some queries where the smoothing on the whole collection better describes the importance of words than does smoothing on the category. The collection smoothing in the baseline LM sometimes performs better than the leaf smoothing. Some 25 queries are affected by this.

7.3.2. Result Analysis of Category Enhancement. To understand why the combinations in the CE model improve on the baseline models, let us recall the two components of the CE model: (1) the global relevance score between a query and the category containing a historical question, and (2) the local relevance score between 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 idea behind is 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 examine the validity of this idea, we analyze the relevant questions returned by TRLM, 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 at every rank. We then aggregate the number of relevant questions at each rank across all queries and compute the percentages of relevant questions at each rank.

Table IX 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. Hence, it is indeed 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.

Comparing the Models for Computing Global Relevance: All the five models improve the retrieval performance, as shown in Tables IV-VIII. However, TR and TRLM perform slightly worse

Table IX. Distribution of relevant questions in different ranks of category in percentage (%, TRLM used for local relevance)

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

than LM when used for computing 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. Recall that category documents are usually much longer than query questions; thus, 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 might sometimes even result in noise in this case. We can also see 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) $= 0.017$, "Pets.Dogs" is also very relevant to the query as determined by TR and TRLM. Table IX also shows that the top-1 category as determined by VSM tends to contain 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 observe 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 X shows the results of these two and VSM as the baseline.

Table X. Results of VSM+VSM.Local, VSM+VSM.Global, and VSM baseline

	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

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. Similarly, in the Okapi Model, the use of local IDF also improves the retrieval performance.

Table XI. MAP results of using LM for computing local relevance score, local smoothing vs. global smoothing

	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

In LM, TR, and TRLM, the smoothing plays a role similar to that of IDF [Zhai and Lafferty 2004]. The three models also improve in performance when the smoothing is done on the categories instead of on the whole collection. We compare the results of LM with local smoothing, called LM.Local, and with global smoothing, called LM.Global. Table XI 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 model is used to compute the global relevance score. In TR and TRLM, the local smoothing also contributes to better performance, and we omit details.

Comparing the Two Methods of Using VSM to Compute Global Relevance. In Section 4.2.1, the second method of computing the global relevance score using VSM uses the centroid vector of a category to compute similarity. We call this method VSMCEN. Table XII 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 4.2.1.

Table XII. Comparison of VSM and VSMCEN

	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

7.3.3. Result Analysis of Query Classification Enhancement. We proceed to analyze the results of LM@QC. Similar to the CE approach, this approach benefits from two aspects: questions from relevant categories are promoted according to the query classification probability, and word importance is determined with regard to a category.

From Equation 18, we can see that the rankings of historical questions from categories with high classification probability scores are promoted. To see whether the promotion is useful, we compute statistics as follows. For each query, we rank all the categories according to the classification results and then count the number of relevant questions at every rank. We next aggregate the number of relevant questions at each rank across all queries and compute the percentage of relevant questions at each rank. Table XIII gives the results. It shows that about 86% of the relevant questions (returned by LM, the baseline model) come from the top-5 ranks. Hence it is reasonable for the LM@QC model to promote the questions from top-ranked categories.

Table XIII. Distribution of relevant questions in different ranks of category (LM results)

Rank	**1**	**2**	**3**	**4**	**5**	**Rest**
Percentage (%)	69.5	6.1	4.4	3.8	2.1	14.1

To observe the effectiveness of local relevance scoring, we compare the results of two models enhanced by the query classification results: LM with local smoothing, denoted as LM.Local@QC, and LM with global smoothing, denoted as LM.Global@QC. Table XIV shows the results of the two models and baseline LM. We can see that smoothing on a category performs better than smoothing on the whole collection, when both are incorporated in the query classification probability. The local smoothing in TR and TRLM, and the local IDF computing in VSM and Okapi, also contributes to better performance, and we do not show the details. We can also see that LM.Global@QC performs better than LM, which also indicates that the query classification indeed improves the retrieval.

Table XIV. Comparison of LM.Local and LM.Global

	MAP	MRR	R-Precision	P5
LM	0.3821	0.5945	0.3404	0.3040
LM.Global@QC	0.4083	0.6083	0.3624	0.3230
LM.Local@QC	0.4571	0.6716	0.4151	0.3514

LM@QC performs worse than the baseline for 52 queries. The reasons include the following:

1) Query question classification errors. The performance of LM@QC relies on the accuracy of query classification. When a more relevant category has a relative low probability classification score, LM@QC performs poorly. We notice that for some queries, LM@QC performs even worse than the baseline when the correct category of the query question is outside the Top-10 classification results.

2) The LM@QC model performs worse than baseline LM when relevant questions come from "non-relevant" categories.

3) The third reason is the same as the second reason for the failure of the LM@LS model analyzed above. Sometimes, leaf category smoothing and local IDF computing lead to worse performance.

Although this approach is intended for retrieval models that compute probability scores, it can also improve other types of models, i.e., VSM and Okapi. However, the improvement is not as large as when it is applied to LM, TR, and TRLM, as shown in Tables IV-VIII.

7.3.4. Result Analysis of Question Classification Enhancement. We analyze the results of LM@DC in this section to show the effectiveness of the question classification enhancement. This approach benefits from two aspects: leaf category smoothing and question classification probability, as explained in Section 6.

First, when computing $P(\mathbf{q}|\mathbf{d}, Cat(\mathbf{d}))$ and $P(\mathbf{q}|\mathbf{d}, cat_i)$ in Equation 20, we do the smoothing with regard to $Cat(\mathbf{d})$ and cat_i, respectively, instead of the whole collection. Second, utilizing the question classification probability, we are able to compute the final ranking score with regard to a question's "truly relevant" categories instead of the category where it is submitted. Thus, the retrieval model is not affected by the problems caused by mistakenly submitted questions and semantically overlapping categories.

Table XV. Comparison of LM.Local and LM.Global

	MAP	MRR	R-Precision	P5
LM	0.3821	0.5945	0.3404	0.3040
LM.Local	0.4463	0.6522	0.4012	0.3341
LM@DC	0.4548	0.6640	0.4107	0.3473

To show the effectiveness of the two factors in this approach, we compare the results of the following three retrieval models: the original language model LM, the language model with local smoothing, which is denoted by LM.Local, and the model LM@DC in which the question classification enhancement approach is applied to LM. The results are shown in Table XV. It can be observed that LM.Local outperforms LM, which demonstrates the effectiveness of the leaf category smoothing. This result is consistent with that as already analyzed in CE and QC approaches. It is also shown in the table that LM@DC performs better than LM.Local, which indicates that the question classification probability is helpful for the retrieval.

In the DC approach, we compute the ranking scores with regard to the "truly relevant" categories. As Equation 20, if a question is submitted to a category which is also the top-1 in the classification results, the DC approach does not change its ranking score. Therefore, the improvement of the DC approach is reflected on those questions that are mistakenly submitted or that could belong to several overlapping categories. In order to analyze the effectiveness of the question classification probability, we compute the statistics on all the relevant questions we annotated for our test set. There are 1373 relevant questions in total. We notice that the rankings of 52 relevant questions with erroneous category labels are promoted substantially. Another 87 questions that may belong to several categories (i.e., the top categories in the classification results of such questions have similar probability scores) also benefit from this approach. The portion of relevant questions promoted is not large (only about 10%), but 55 queries (about 20% of the test set) benefit from the question classification probability.

Table XVI. Results of approach LS, QC and CE incorporated with DC

	LM@LS	**LM@LS@DC**	**LM@QC**	**LM@QC@DC**	**LM+LM**	**LM+LM@DC**
MAP	0.4586	0.4611	0.4571	0.4602	0.4609	0.4654
MRR	0.6620	0.6718	0.6716	0.6788	0.6622	0.6721
R-Prec	0.4072	0.4141	0.4151	0.4223	0.4087	0.4156
P@5	0.3460	0.3498	0.3514	0.3572	0.3519	0.3562

As shown in Table XVI, after incorporated with the question classification enhanced approach, our first three approaches applied to LM, i.e., LM@LS@DC, LM+LM@DC, and LM@QC@DC (described in Section 6.2, 6.3, and 6.4 respectively) always outperform the original models. This again underlines the effectiveness of question classification probabilities. Table XXIII exemplifies the effectiveness of this approach.

7.3.5. Effects of the Question Length on the Performance. In this section we study the effects of the question length on the performance of our approaches.

Table XVII. The improvement of the four approaches on queries with different lengths; * indicates a statistically significant improvement over the baseline LM using the t-test, p-value < 0.05)

Length	**Number**	**LM**	**LM@LS**	**%chg**	**LM+LM**	**%chg**	**LM@QC**	**%chg**	**LM@DC**	**%chg**
1-4	125	0.3951	0.4690	18.7*	0.4711	19.2*	0.4646	17.6*	0.4620	16.9*
5-8	108	0.3765	0.4518	20*	0.4586	21.8*	0.4569	21.4*	0.4546	20.7*
9-13	19	0.3283	0.4285	30.5*	0.4068	23.9*	0.4088	24.5*	0.4085	24.4*

Table XVIII. The number of relevant questions promoted with different lengths

Length	**Number**	**LM@LS**	**LM+LM**	**LM@QC**	**LM@DC**
1-4	674	165	166	178	173
5-8	593	143	151	165	152
9-13	106	30	32	32	30

We first study the effects of query length. Table XVII shows the improvement of the four approaches, i.e., LS, CE, QC, and DC, applied to the language model over the baseline language model on queries of different lengths. It can be observed that all the approaches are able to significantly improve the performance of question retrieval on both short and long queries.

Next, we study the effect of the length of relevant questions in question repository. Table XVIII gives the statistics of the 1373 relevant questions of the test queries, and shows the distribution of relevant questions promoted of different lengths. We can see that our approaches are able to improve the performance for both short questions and long questions.

Table XIX. Search results for “Wat is the best way to talk my mom into letting me get a snake???”

Models	Rank	Questions	Categories
LM	1	How can you talk a mom in to letting you get a motorcycle?	Motorcycles
	12	**I am trying to get my mom to get me a corn snake What should i do?**	Reptiles
	13	**How do I get my mom to let me have a rabbit? She still wouldn’t let me have a snake.?**	Other - Pets
LM@LS	1	How can you talk a mom in to letting you get a motorcycle?	Motorcycles
	7	**I am trying to get my mom to get me a corn snake What should i do?**	Reptiles
	8	**How do I get my mom to let me have a rabbit? She still wouldn’t let me have a snake.?**	Other - Pets
LM@QC	1	**I am trying to get my mom to get me a corn snake What should i do?**	Reptiles
	2	**How do I get my mom to let me have a rabbit? She still wouldn’t let me have a snake.?**	Other - Pets
	3	How can i talk my mom into letting me move back	Family
LM+LM	1	**I am trying to get my mom to get me a corn snake What should i do?**	Reptiles
	2	**How do I get my mom to let me have a rabbit? She still wouldn’t let me have a snake.?**	Other - Pets
	3	Hi i need information on a snake i have if neone would like to help me get on yahoo messenger and talk with me	Reptiles

Table XX. Search results for “Do guppies die after giving birth?”

Method	Rank	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+	1	Do you like guppies?	Fish
VSM.	2	How many guppies do I have?	Fish
Global	3	**Is it at all normal for a guppy mother to die a day or so after giving birth?**	Fish
VSM+	1	**Is it at all normal for a guppy mother to die a day or so after giving birth?**	Fish
VSM.	2	Lifespan of platy females after giving birth?	Fish
Local	3	Will my Guppies die if I get them to early? HELP!?	Fish

7.3.6. Examples. To get a better understanding of our models, we illustrate them by means of several examples.

Example 1: The Effectiveness of Promoting the Rankings of Questions from Relevant Categories

Table XIX gives part of the results of the example query “Wat is the best way to talk my mom into letting me get a snake???,” which is originally in the category “Pets.Reptiles.” The questions in bold are labeled as “relevant.” In the LM@LS model, the category-specific frequent word “snake” renders the category “Reptiles” and “Other-Pets” more relevant to the query. Hence the rankings of the questions in the two categories are promoted. In model LM@QC, the category “Reptiles” and “Other-Pets” are the Top-2 categories as judged by the classifier. Similarly, in model LM+LM, “Reptiles” and “Other-Pets” are the categories having the largest global ranking scores. Therefore the rankings of questions in the two categories are promoted by the LM@QC and LM+LM.

Example 2: The Effectiveness of Using Local IDF Computation

Table XX gives the top-3 results of the query question “Do guppies die after giving birth?” (from the category “Pets.Fish”) when using the models VSM, VSM+VSM.Global (using global IDF for computing local relevance) and VSM+VSM.Local (using local IDF for computing local relevance). 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 ques-

Table XXI. Search results for "How can I trim my parakeets beak?"

Models	Rank	Questions	Categories
LM	1	My parakeets beak is bruised ,what should I do?	Birds
	2	**How to Trim a Bird's Beak ?**	Birds
LM@LS	1	**How to Trim a Bird's Beak ?**	Birds
	4	My parakeets beak is bruised ,what should I do?	Birds
LM@QC	1	**How to Trim a Bird's Beak ?**	Birds
	4	My parakeets beak is bruised ,what should I do?	Birds
LM+LM	1	**How to Trim a Bird's Beak ?**	Birds
	4	My parakeets beak is bruised ,what should I do?	Birds

Table XXII. Search results for "Will marijuana leave my system faster if i drink alot of water?"

Method	Rank	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

tion in category "Pets.Fish." This is because the category has a high score for the query, while the scores of categories "Pregnancy & Parenting" and "Dream Interpretation" are lower. Second, VSM+VSM.Local using the local IDF ranks questions within the same category better. This is because in VSM+VSM.Local, the words "giving" and "birth" are more important than the word "guppies," while the opposite is true in VSM+VSM.Global.

Example 3: The Effectiveness of Leaf Category Smoothing

In addition, recall that leaf smoothing in LM@LS improves the results by ranking questions from the same category better. To illustrate this, Table XXI gives an example for the query "How can I trim my parakeets beak?" that belongs to "Pets.Birds." In this category, word "parakeets" occurs 276 times and "trim" occurs only 5 times. As a result, the word "trim" is more important than "parakeets" for the ranking of questions within this category. However, in the whole collection, the word "parakeets" is more important than "trim" since it appears in fewer questions. This is why LM@LS, LM@QC, and LM+LM promote the ranking of the question "How to Trim a Bird's Beak ?" compared with the baseline model LM (from 2nd to 1st).

Example 4: The Effectiveness of Using Word Translation Probabilities

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 lexical gaps and find lexically different but semantically similar questions. Table XXII gives the results of "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 does 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.

Example 5: The Effectiveness of Using Question Classification Probabilities

Table XXIII compares the results of LM@LS and LM@LS@DC. We can see that LM@LS@DC is able to find one more relevant question, which actually should be contained in "Entertainment & Music.Comics & Animation," but was mistakenly submitted to "Entertainment & Music.Music.Country" by a user. According to the classifier, this question belongs to "Entertainment &

Table XXIII. Search results for “Where can if download theme song of Anime shows ?”

Method	Rank	Relevant Questions and their ranks	Category
LM@LS	2	**Where can i get free anime theme song downloads?**	Comics & Animation
	10	**Its everyone know what web for download anime song?**	Comics & Animation
LM@LS@DC	1	**Where can i get free anime theme song downloads?**	Comics & Animation
	8	**Its everyone know what web for download anime song?**	Comics & Animation
	11	**Where can i download anime song fast and free please tell me...?**	Country

Music.Comics & Animation” with probability higher than 99%. Therefore, we also do the smoothing in category “Entertainment & Music.Comics & Animations” in addition to “Entertainment & Music.Music.Country.”

7.4. Summary of Experiments

The experimental results show that all of the four approaches (LS, CE, QC, and DC) that utilize category information are able to improve the question search performance significantly when applied to existing retrieval models. LS benefits from the leaf category smoothing. Category frequent specific words are exploited to distinguish questions from different categories and questions from the same category. CE benefits from both the global and the local relevance score. The global relevance score helps find out the relevant categories, and the local relevance score helps distinguish questions from the same category. QC takes advantage of the query classification probabilities. Questions from categories with high classification probabilities are promoted during retrieval. DC utilizes the question classification probabilities to reduce the effect of mistakenly submitted questions and overlapping categories. When this approach is incorporated into the other three approaches, an even better performance is achieved. In addition, the query classification probabilities can be used to prune search space to reduce the query response time. The threshold-based pruning enables us to choose different trade-offs between effectiveness and efficiency.

8. RELATED WORK

Question Retrieval. The work on question retrieval can be traced back to finding similar questions in Frequently Asked Questions (FAQs) archives. Most existing work focuses on addressing the word mismatching problem between user questions and the questions in a Question Answering (QA) archive. This problem is especially important for QA retrieval, since question-answer pairs are usually short [Xue et al. 2008]. Burke et al. [1997] combine lexical and semantic similarity between questions to rank FAQs, where the lexical similarity is computed using a vector space model and the semantic similarity is computed based on WordNet. Berger et al. [2000] study several statistical approaches to bridging 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 de Rijke [2005] use supervised learning methods to extract QA pairs from FAQ pages, and they use a vector space model for question-answer pair retrieval. Riezler et al. [2007] provide an effective translation model for question search; their translation model is trained on a large amount of data extracted from FAQ pages on the Web. Soricut and Brill [2004] use one million FAQs collected from the Web to train their answer passage retrieval system.

Next, question search has recently been revisited on CQA data. Jeon et al. [2005a,b] compare four different retrieval methods, namely, 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 [Xue et al. 2008], 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. [2008] 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. Bian et al. [2008] propose an interesting learning framework for question retrieval. However, this approach needs training data (which is difficult to get for general questions) and experiments are conducted on factoid questions. Wang et al. [2009] 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. [2009], current parsers are not well-trained to parse real-life questions, especially informally stated questions. They report that they outperform a Bag-of-Words baseline (that matches words between query and question) by 11.99% in terms of MAP. Wang et al. [2010] propose a graph-based approach to segmenting multi-sentence questions. Their approach outperforms previous work that focuses mostly on the retrieval of single-sentence queries. Ming et al. [2010] explore domain-specific term weight for question search.

This paper combines and substantially extend two earlier papers [Cao et al. 2009, 2010] by the authors. In particular, in the paper [Cao et al. 2009], the effective category smoothing based approach is tightly integrated with the Language Model. We extend this approach to the the Translation Model and the Translation-based Language Model. We also modify the classification-based approach in that work. We find that the new approach is able to improve the performance of retrieval significantly. We incorporate the question classification enhancement into the approach proposed in the work [Cao et al. 2010]. The new combined model performs better since it avoids problems caused by wrongly categorized questions and overlapping categories.

Apart from these two works, to the best of our knowledge, no previous works attempt to utilize category information to improve question search, although this appears to be a natural idea. In contrast to all the previous work, our question search approaches exploit the question categories in CQA data, where all questions are organized into a hierarchy of categories.

Although our approaches are very different from previous work on question search, our CE, QC, and DC approaches combine well with existing work in that they can be easily integrated with previous work. In principle, we can employ any question retrieval model within the CE approach, and we an employ any retrieval model that computes probability scores within the QC and DC approaches.

Finally, we note that other works exists on CQA services that consider category information (e.g., [Liu et al. 2008; Wei et al. 2011]). However, these works study other aspects of CQA, not question retrieval.

Classification for Document Retrieval. Chekuri et al. [1997] 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. Lam et al. [1999] develop an automatic text categorization approach to classify both documents and queries, and they investigate the application of text categorization to text retrieval. Our approaches are very different from the approaches in existing work for document retrieval.

Cluster-based Document Retrieval. The research on cluster-based retrieval can be traced back to Van Rijsbergen's cluster hypothesis in 1979. The basic idea is that closely associated documents tend to be relevant to the same requests [van Rijsbergen 1979]. Following its formulation, this hypothesis has been used widely in information retrieval. In cluster-based retrieval, documents are grouped into clusters, and a list of documents is returned based on the clusters that they come from. Most early work on clustering mainly focuses on improving the efficiency [Jardine and van Rijsbergen 1971; Croft 1980]. It proved to be helpful in improving the retrieval performance in later work [Griffiths et al. 1986].

The cluster hypothesis can be interpreted globally or locally. The global interpretation does the clustering on the entire corpus, and the documents in each cluster are related to the same topic derived from the inter-document similarities. We can roughly classify the global interpretation of cluster hypothesis into two groups. Approaches in the first group one or more clusters in their entirety in response to a query. For example, in early work [Jardine and van Rijsbergen 1971], entire documents are clustered, and clusters are retrieved based on how well their centroids match a given query. Approaches in the second group compute ranking scores of individual documents in a cluster against a query (e.g., [Hearst and Pedersen 1996; Kurland and Lee 2004; Liu and Croft 2004]).

In the second group, some works combine the language model with cluster-based retrieval, which is of relevance to our work. Liu and Croft [2004] utilize an offline clustering approach and smooth documents language models with the cluster that a document belongs to. Our leaf category smoothing enhancement is inspired by their *CBDM* model. The difference is that we utilize categories instead of clusters. We also extend this idea to other retrieval models, such as the Translation Model and the state-of-the-art work, Translation-based Language Model. In addition, previous work [Liu and Croft 2004] on cluster-based retrieval model does not offer theoretical analyses of why cluster-based retrieval helps in the Language Model. The theoretical analysis given in this paper is also applied to the cluster-based retrieval model [Liu and Croft 2004].

Kurland and Lee [2004, 2009] utilize an overlapping clustering approach and propose a novel algorithmic framework. They enhance document language models by the incorporating of information drawn from clusters of similar documents. Our question classification enhancement shares some common ideas with one of their algorithms. The difference is: first, they utilize the Language Model to estimate the probability of a document belonging to a cluster; second, they assume that a query is conditionally independent of a document given a cluster. In our DC approach, the question classification is computed by the classifier, and a query is dependent on both the historical question and a category.

The local interpretation focuses mainly on the local relationships between documents instead of global clustering, and thus it avoids the problems of selecting the number of clusters and of modeling the sizes of clusters.

Diaz [2005, 2007] proposes a graph-based score regularization approach. The basic idea is to smooth the document scores with those of related documents. Documents are similar to both the query and the other query related documents have high rankings in their work. They claim that cluster-based retrieval is an instance of regularization. Our category enhancement can also be viewed as a kind of score regularization. Related documents are the historical questions in the same category in our CE approach. The difference is that we do the regularization with the ranking score of the big pseudo-document by concatenating these questions instead of with the ranking score of each single historical question.

Erkan and Radev [2004] also present a graph-based framework called LexRank. A weighted graph is constructed according to inter-document similarities, and then a PageRank like method is applied to the graph. They show that their degree-based method outperforms the previous centroid-based methods. However, building such a text similarity based graph is too time consuming for large data sets such as the ones we consider.

No existing cluster-based retrieval approaches have been applied to question search.

9. CONCLUSIONS

Question retrieval is an essential component in Community Question Answer (CQA) services. In this paper, we propose several new approaches that are capable of exploiting category information associated with questions in CQA archives for improving question retrieval, namely, the leaf category smoothing enhancement (LS), the category enhancement (CE), the query classification enhancement (QC), and the question classification enhancement (DC). These approaches can be applied easily to existing question retrieval models. In summary, LS utilizes the category specific frequent words to improve question retrieval; CE combines a global relevance score between a query and a category and a local relevance score between a query and a historical question; QC takes advantage of the query classification probabilities to adjust the ranking scores of historical questions; DC benefits from the question classification probabilities, and it avoids the problems of mistakenly submitted questions and overlapping categories. Experiments conducted on a large CQA data set from Yahoo! Answers demonstrate the effectiveness and efficiency of these proposed approaches.

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., [Duan et al. 2008]). 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.

Acknowledgement. Gao Cong is supported in part by a grant awarded by Microsoft Research Asia and by a Singapore MOE AcRF Tier 1 Grant (RG16/10). Bin Cui is supported by the grant of Natural Science Foundation of China (No. 61073019). The scholarship of Quan Yuan is funded by Institute for Media Innovation, NTU.

REFERENCES

AGICHTEIN, E., CASTILLO, C., DONATO, D., GIONIS, A., AND MISHNE, G. 2008. Finding high-quality content in social media. In *WSDM*. 183–194.

BERGER, A., CARUANA, R., COHN, D., FREITAG, D., AND MITTAL, V. 2000. Bridging the lexical chasm: statistical approaches to answer-finding. In *SIGIR*. 192–199.

BIAN, J., LIU, Y., AGICHTEIN, E., AND ZHA, H. 2008. Finding the right facts in the crowd: factoid question answering over social media. In *WWW*. 467–476.

BURKE, R. D., HAMMOND, K. J., KULYUKIN, V. A., LYTINEN, S. L., TOMURO, N., AND SCHOENBERG, S. 1997. Question answering from frequently asked question files: Experiences with the faq finder system. *AI Magazine 18,* 2, 57–66.

CAO, X., CONG, G., CUI, B., AND JENSEN, C. S. 2010. A generalized framework of exploring category information for question retrieval in community question answer archives. In *WWW*. 201–210.

CAO, X., CONG, G., CUI, B., JENSEN, C. S., AND ZHANG, C. 2009. The use of categorization information in language models for question retrieval. In *CIKM*. 265–274.

CHEKURI, C., GOLDWASSER, M. H., RAGHAVAN, P., AND UPFAL, E. 1997. Web search using automatic classification. In *WWW*.

CROFT, W. B. 1980. A model of cluster searching based on classification. *Information Systems 5*, 189–195.

DIAZ, F. 2005. Regularizing ad hoc retrieval scores. In *CIKM*. New York, NY, USA, 672–679.

DIAZ, F. 2007. Regularizing query-based retrieval scores. *Information Retrieval 10,* 6, 531–562.

DUAN, H., CAO, Y., LIN, C.-Y., AND YU, Y. 2008. Searching questions by identifying question topic and question focus. In *ACL-HLT*. 156–164.

DUMAIS, S. AND CHEN, H. 2000. Hierarchical classification of web content. In *SIGIR*. 256–263.

ERKAN, G. AND RADEV, D. R. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research 22*, 457–479.

FANG, H., TAO, T., AND ZHAI, C. 2004. A formal study of information retrieval heuristics. In *SIGIR*. 49–56.

GRIFFITHS, A., LUCKHURST, H. C., AND WILLETT, P. 1986. Using interdocument similarity information in document retrieval systems. *Journal of the American Society for Information Science 37,* 1, 3–11.

HEARST, M. A. AND PEDERSEN, J. O. 1996. Reexamining the cluster hypothesis: scatter/gather on retrieval results. In *SIGIR*. 76–84.

JARDINE, N. AND VAN RIJSBERGEN, C. 1971. The use of hierarchical clustering in information retrieval. *Information Storage and Retrieval 7*, 217–240.

JEON, J., CROFT, W. B., AND LEE, J. H. 2005a. Finding semantically similar questions based on their answers. In *SIGIR*. 617–618.

JEON, J., CROFT, W. B., AND LEE, J. H. 2005b. Finding similar questions in large question and answer archives. In *CIKM*. 84–90.

JIJKOUN, V. AND DE RIJKE, M. 2005. Retrieving answers from frequently asked questions pages on the web. In *CIKM*. 76–83.

KURLAND, O. AND LEE, L. 2004. Corpus structure, language models, and ad hoc information retrieval. In *SIGIR*.

KURLAND, O. AND LEE, L. 2009. Clusters, language models, and ad hoc information retrieval. *ACM Trans-*

actions on Information Systems 27, 3.

LAM, W., RUIZ, M., AND SRINIVASAN, P. 1999. Automatic text categorization and its application to text retrieval. *IEEE Transactions on Knowledge and Data Engineering 11,* 6, 865–879.

LIU, X. AND CROFT, W. B. 2004. Cluster-based retrieval using language models. In *SIGIR*. 186–193.

LIU, Y., BIAN, J., AND AGICHTEIN, E. 2008. Predicting information seeker satisfaction in community question answering. In *SIGIR*. 483–490.

MING, Z., CHUA, T.-S., AND CONG, G. 2010. Exploring domain-specific term weight in archived question search. In *CIKM*. 1605–1608.

RIEZLER, S., VASSERMAN, A., TSOCHANTARIDIS, I., MITTAL, V. O., AND LIU, Y. 2007. Statistical machine translation for query expansion in answer retrieval. In *ACL*. 464–471.

ROBERTSON, S., WALKER, S., JONES, S., HANCOCK-BEAULIEU, M., AND GATFORD, M. 1994a. Okapi at trec-3. In *TREC*. 109–126.

ROBERTSON, S., WALKER, S., JONES, S., HANCOCK-BEAULIEU, M., AND GATFORD, M. 1994b. Okapi at trec-3. In *TREC*. 109–126.

SEBASTIANI, F. 2002. Machine learning in automated text categorization. *ACM Comput. Surv. 34,* 1, 1–47.

SINGHAL, A., BUCKLEY, C., AND MITRA, M. 1996. Pivoted document length normalization. In *SIGIR*. 21–29.

SORICUT, R. AND BRILL, E. 2004. Automatic question answering: Beyond the factoid. In *HLT-NAACL*. 57–64.

VAN RIJSBERGEN, C. J. 1979. *Information Retrieval*. Butterworth.

VOORHEES, E. M. 2001. Overview of the trec 2001 question answering track. In *TREC*. 42–51.

WANG, K., MING, Z., AND CHUA, T.-S. 2009. A syntactic tree matching approach to finding similar questions in community-based qa services. In *SIGIR*. 187–194.

WANG, K., MING, Z., HU, X., AND CHUA, T.-S. 2010. Segmentation of multi-sentence questions: towards effective question retrieval in cqa services. In *SIGIR*. 387–394.

WEI, W., CONG, G., LI, X., NG, S.-K., AND LI, G. 2011. Integrating community question and answer archives. In *AAAI*. 1255–1260.

XUE, X., JEON, J., AND CROFT, W. B. 2008. Retrieval models for question and answer archives. In *SIGIR*. 475–482.

ZHAI, C. AND LAFFERTY, J. 2004. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems 22,* 2, 179–214.

ZOBEL, J. AND MOFFAT, A. 2006. Inverted files for text search engines. *ACM Computing Surveys 38,* 2, 6.