The Use of Categorization Information in Language Models for Question Retrieval

Xin Cao[†], Gao Cong[†], Bin Cui[‡], Christian S. Jensen[†], Ce Zhang[‡]
[†]Department of Computer Science, Aalborg University, Denmark
{xcao, gaocong, csj}@cs.aau.dk
[‡]School of EECS, Peking University, China
{bin.cui, ce.zhang}@pku.edu.cn

ABSTRACT

Community Question Answering (CQA) has emerged as a popular type of service meeting a wide range of information needs. Such services enable users to ask and answer questions and to access existing question-answer pairs. CQA archives contain very large volumes of valuable user-generated content and have become important information resources on the Web. To make the body of knowledge accumulated in CQA archives accessible, effective and efficient question search is required. Question search in a CQA archive aims to retrieve historical questions that are relevant to new questions posed by users. This paper proposes a category-based framework for search in CQA archives. The framework embodies several new techniques that use language models to exploit categories of questions for improving question-answer search. Experiments conducted on real data from Yahoo! Answers demonstrate that the proposed techniques are effective and efficient and are capable of outperforming 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

Keywords

question-answering services, question search, categorization, language model

CIKM'09, November 2–6, 2009, Hong Kong, China.
Copyright 2009 ACM 978-1-60558-512-3/09/11 ...$10.00.

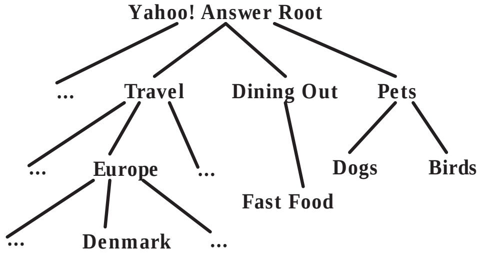

Figure 1: The Category Structure of Yahoo! Answers

1. INTRODUCTION

Community Question Answering (CQA) services are Internet services that enable users to ask and answer questions, as well as to search through historical question-answer pairs. Examples of such community-driven knowledge market services include Yahoo! Answers (answers.yahoo.com)[1], Naver (www.naver.com), Baidu Zhidao (zhidao.baidu.com), and WikiAnswers(wiki.answers.com).

The success of Question Answering (QA) services motivates research in question-answer search where the pre-existing, historical question-answer pairs that best match a user's new question are to be retrieved [3, 7, 11, 12, 24, 25], as such functionality is an essential component of a CQA service. In addition, when a user chooses to ask a new question in a CQA service, the CQA service could automatically search and display pre-existing question-answer pairs that match the new question, if any. If good matches are found, the user needs not wait for other users to answer the question, thus reducing the waiting time and improving the user satisfaction. Hence, it is important the search service offers relevant results efficiently. This paper's focus is to improve question search for CQA services.

When a user asks a question in a CQA service, the user typically needs 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 and questions in CQA services are organized into hierarchies of categories. Figure 1 shows a small part of the hierarchy of Yahoo! Answers.

[1] Yahoo ! Answers dominate the answer site market share in U.S. according to a study by Hitwise (http://www.hitwise.com/press-center/hitwiseHS2004/question-and-answer-websites.php)

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 in the country of Denmark. Although recent work has been done on question search in CQA data, we are not aware of any such work that aims to exploit the available categorizations of questions as exemplified above for question search.

To exemplify how a categorization of questions may be exploited, consider a user who enters the following question (**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 question (**d**) "Can you recommend sightseeing opportunities for senior citizens in Texas?" 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 **d** highly among the list of returned results.

In this paper we propose a new framework for exploiting categorization information in question search, and several approaches to realizing the framework. More specifically, the categorization information will be utilized in two respects.

First, the category information of all candidate historical questions can be incorporated into computing the relevancy score of a historical question to a query question. The idea consists of two levels. 1) Comparing the relevancy of historical questions in different categories to a query question: If words in the query question are frequent in one category (i.e., it occurs in many questions in the category) while being neglectable in other category, this indicates that the questions in the former category are more likely relevant to question **q** than questions in the latter. For example, recall the query question **q**, where word "Denmark" in **q** is frequent in category "Travel.Europe.Denmark" but not in category "Travel.US.Texas". Suppose that other words in **q** are not distinguishable in the two categories. This indicates that questions in category "Travel.Europe.Denmark" category are more likely to be relevant. 2) Comparing the relevancy of questions within the same category: frequent words in a category are less useful to distinguish questions in a category. For example, the word "Denmark" will be unimportant when we compare the relevancy of two questions in "Travel.Europe.Denmark" to query **q**, since nearly all questions in the category are about "Denmark".

Second, we make use of the results of query question classification to enhance the retrieval effectiveness. One straightforward approach is to predetermine the category of a question **q** using a classifier, and then to search relevant questions within that category. This approach is able to improve efficiency by pruning the search space. Unfortunately, we find that not only the category of a question **q** , but also other categories may contain relevant questions of question **q**. Thus, searching only within the category of question **q** will miss relevant questions even if the category of **q** could be correctly determined.

In contrast to searching questions within the category of query question **q**, we can compute the probability of the question **q** belonging to each category. The probability can be used to adjust the relevancy score of a question **d** in the category to question **q**. Historical questions from categories with high probability should be promoted.

Additionally, we can utilize query question categorization to prune search space to improve efficiency. Specifically, for each category we compute the probability that a query question belongs to the category and we search relevant questions only within the categories with probability values larger than a threshold.

In this paper, we explore the ideas outlined above in language model based question search. More specifically, the paper's contributions are twofold.

First, we propose two approaches to enhancing question search with categorization information: 1) we use a category language model to smooth a question language model to explore the first idea. And 2) we integrate the classification scores returned by a classifier built with historical question data into language models to explore the second idea outlined above. We also explore a solution built on top of both ideas. To our knowledge, this is the first work that leverages categorization information for question search.

Second, we conduct experiments with a large real data set consisting of more than 3 millions of questions from Yahoo! Answers to empirically elicit pertinent properties of the techniques that make up the proposed framework. Experimental results show that the proposed technique is capable of significantly improving the baseline language model without using category information for question search in terms of both effectiveness and efficiency, i.e. category information is indeed useful for question search.

The remainder of this paper is organized as follows. Section 2 details the proposed techniques. Section 3 reports on the experimental study. Section 4 reviews related works. Finally, Section 5 concludes and identifies research directions.

2. CATEGORY BASED RETRIEVAL FRAMEWORK

The questions are organized into hierarchical categories in Yahoo! Answers. This section first introduces language models and then presents the proposed techniques based on language models to exploit the category information in question retrieval.

2.1 Preliminaries on language models

Language models have performed quite well empirically in many information retrieval tasks [18, 17, 26], and also have performed very well in question search [12]. The basic idea is to estimate a language model for each document (resp. question), and then rank documents (resp. questions) by the likelihood of the query according to the estimated model. Given a query **q** and a document **d**, the ranking function for the query likelihood language model using Jelinek-Mercer smoothing method [26] is as follows:

$$P(\mathbf{q}|\mathbf{d}) = \prod_{w \in \mathbf{q}} P(w|\mathbf{d})$$

$$P(w|\mathbf{d}) = (1-\lambda)P_{ml}(w|\mathbf{d}) + \lambda P_{ml}(w|Coll),$$

$$\begin{aligned} P_{ml}(w|\mathbf{d}) &= \frac{tf(w,\mathbf{d})}{\sum_{w' \in \mathbf{d}} tf(w',\mathbf{d})} \\ P_{ml}(w|Coll) &= \frac{tf(w,Coll)}{\sum_{w' \in Coll} tf(w',Coll)} \end{aligned} \tag{1}$$

where w is a word in the query; $P_{ml}(w|\mathbf{d})$ is the maximum likelihood estimate of word w in **d**; $P_{ml}(w|Coll)$ is the maximum likelihood estimate of word w in the collection $Coll$;

$tf(w, \mathbf{d})$ is the frequency of word w in document $\mathbf{d}$; and λ is the smoothing parameter.

2.2 Language model with leaf category smoothing

Each question in Yahoo! Answers belongs to a leaf category. This approach is to realize the first idea discussed in introduction. In this approach, category information of historical queries is utilized such that category-specific frequent words will play an important role in comparing the relevancy of historical questions across categories to a query, while category-specific frequent words are less important than category-specific infrequent words in comparing the relevancy of questions within the same category.

This idea can be realized by two levels of smoothing. Namely, 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 model and then show why this smoothing model meets the requirements.

Given a user search question $\mathbf{q}$ and a candidate question $\mathbf{d}$ (in a QA repository), we compute the probability $P(\mathbf{q}|\mathbf{d})$ of how likely $\mathbf{q}$ could have been generated from $\mathbf{d}$. $P(w|\mathbf{d})$ will be 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 need to estimate language model $P(w|\mathbf{d})$. In this approach Equation 1 will be modified as follows.

$$\begin{aligned} P(\mathbf{q}|\mathbf{d}) &= \prod_{w \in \mathbf{q}} P(w|\mathbf{d}) \\ P(w|\mathbf{d}) &= (1-\lambda)P_{ml}(w|\mathbf{d}) \\ &\quad + \lambda[(1-\beta)P_{ml}(w|Cat(\mathbf{d})) + \beta P_{ml}(w|Coll)] \end{aligned} \tag{2}$$

where w is a word in question $\mathbf{q}$, λ and β are two different smoothing parameters, $Cat(\mathbf{d})$ denotes the category of historical question $\mathbf{q}$, $P_{ml}(w|\mathbf{d})$ is the maximal likelihood estimate of word w in question $\mathbf{d}$ and can be computed by Equation 1, 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 maximal 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 with Equation 2. We call the approach LM+L.

We proceed to show that category-specific frequent words will play an important role in ranking the relevancy of questions across different categories in this model to a query $\mathbf{q}$. We can 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:

$$\begin{aligned} \log P(\mathbf{q}|\mathbf{d}) &= \sum_{w \in \mathbf{q}} \log P(w|\mathbf{d}) \\ &= \sum_{w \in \mathbf{q}}^{tf(w,Cat(\mathbf{d}))>0} \log P_{cs}(w|\mathbf{d}) + \sum_{w \in \mathbf{q}}^{tf(w,Cat(\mathbf{d}))=0} \log P_{cu}(w|\mathbf{d}) \\ &= \sum_{w \in \mathbf{q}}^{tf(w,Cat(\mathbf{d}))>0} \log \frac{P_{cs}(w|\mathbf{d})}{P_{cu}(w|\mathbf{d})} + \sum_{w \in \mathbf{q}} \log P_{cu}(w|\mathbf{d}) \end{aligned} \tag{3}$$

According to the leaf smoothing model, we have:

$$\begin{aligned} P_{cs}(w|\mathbf{d}) =& (1-\lambda)P_{ml}(w|\mathbf{d}) + \\ & \lambda[(1-\beta)P_{ml}(w|Cat(\mathbf{d})) + \beta P_{ml}(w|Coll)] \\ P_{cu}(w|\mathbf{d}) =& \lambda\beta P_{ml}(w|Coll) \end{aligned} \tag{4}$$

From Equation 3 and the two equations in Equation 4, we get:

$$\begin{aligned} &\log P(\mathbf{q}|\mathbf{d}) = \\ &\sum_{w \in \mathbf{q}}^{tf(w,Cat(\mathbf{d}))>0} \log(\frac{(1-\lambda)P_{ml}(w|\mathbf{d}) + \lambda(1-\beta)P_{ml}(w|Cat(\mathbf{d}))}{\lambda\beta P_{ml}(w|Coll)} \\ &\quad + 1) + \sum_{w \in \mathbf{q}} \log \lambda\beta P_{ml}(w|Coll) \end{aligned} \tag{5}$$

Now we can see that the second term in the right hand of Equation 5 is independent of $\mathbf{d}$, and thus can be ignored in ranking. We can also see from the first term in the right hand that for questions from different categories, the larger $P_{ml}(w|Cat(\mathbf{d}))$, the larger $P(\mathbf{q}|\mathbf{d})$, i.e. the more a word w in question $\mathbf{q}$ occurs in a category, the higher relevancy score the questions in the category will get. Hence, the category smoothing model will play 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 in this model. As in [26], we define a model $P_s(w|\mathbf{d})$ used for "seen" words that occur in 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:

$$\begin{aligned} \log P(\mathbf{q}|\mathbf{d}) &= \sum_{w \in \mathbf{q}} \log P(w|\mathbf{d}) \\ &= \sum_{w \in \mathbf{q}}^{tf(w,\mathbf{d}))>0} \log P_s(w|\mathbf{d}) + \sum_{w \in \mathbf{q}}^{tf(w,\mathbf{d}))=0} \log P_u(w|\mathbf{d}) \\ &= \sum_{w \in \mathbf{q}}^{tf(w,\mathbf{d}))>0} \log \frac{P_s(w|\mathbf{d})}{P_u(w|\mathbf{d})} + \sum_{w \in \mathbf{q}} \log P_u(w|\mathbf{d}) \end{aligned} \tag{6}$$

In the leaf smoothing model, from Equation 2 we know:

$$\begin{aligned} &P_s(w|\mathbf{d}) = (1-\lambda)P_{ml}(w|\mathbf{d}) + \\ &\quad \lambda[(1-\beta)P_{ml}(w|Cat(\mathbf{d})) + \beta P_{ml}(w|Coll)] \\ &P_u(w|\mathbf{d}) = \lambda[(1-\beta)P_{ml}(w|Cat(\mathbf{d})) + \beta P_{ml}(w|Coll)] \end{aligned} \tag{7}$$

From Equation 6 and the two equations in Equation 7 we get:

$$\begin{aligned} &\log P(\mathbf{q}|\mathbf{d}) = \\ &\sum_{w \in \mathbf{q}}^{tf(w,\mathbf{d}))>0} \log(\frac{(1-\lambda)P_{ml}(w|\mathbf{d})}{\lambda[(1-\beta)P_{ml}(w|Cat(\mathbf{d})) + \beta P_{ml}(w|Coll)} + 1) \\ &+ \sum_{w \in \mathbf{q}} \log(\lambda[(1-\beta)P_{ml}(w|Cat(\mathbf{d})) + \beta P_{ml}(w|Coll)) \end{aligned} \tag{8}$$

As we can see, for the questions in the same category $Cat(\mathbf{d})$, the second term in the right hand side of Equation 8 is the same, and thus will not affect the relative ranking of them; but the first term in the right hand side will be inversely proportional to the maximal 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 the 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 the model LM+L.

As a summary of the above analysis: On one hand, for questions in different categories, leaf category smoothing will enable the questions in the category which is more relevant to the query to gain higher relevancy scores; On the other hand, for questions in the same category, leaf category smoothing plays a similar role as IDF computed with regard to the category.

This approach is inspired by the clustering based retrieval model $CBDM$ for document retrieval in [14, 16]. However, previous work on clustering based retrieval model does not establish the above analysis and the analysis will also provide insight for the cluster based retrieval model.

2.3 Language model with query classification

One straightforward method of leveraging the classification of a query question can be done as follows: first find out the category of the query using a classification model, and then rank questions in this category using the language model in Section 2.1 to retrieve relevant questions. Specifically, we build a classification model using historical questions to classify a query question, and we compute the probability $P_{Cat}(\mathbf{q}|\mathbf{d})$ which represents how likely the query question $\mathbf{q}$ could have been generated from the historical question $\mathbf{d}$ with regard to the category $Cat(\mathbf{d})$ containing $\mathbf{d}$ as Equation 9. The probability is used to rank the relevancy of historical questions to query $\mathbf{q}$.

$$P_{Cat}(\mathbf{q}|\mathbf{d}) = \begin{cases} P(\mathbf{q}|\mathbf{d}), CLS(\mathbf{q}) = Cat(\mathbf{d}) \\ \quad 0, CLS(\mathbf{q}) \neq Cat(\mathbf{d}) \end{cases} \tag{9}$$

where $P(\mathbf{q}|\mathbf{d})$ is computed by Equation 1, and $CLS(\mathbf{q})$ represents the category of the query $\mathbf{q}$ determined by the classifier.

The simple approach can greatly improve the efficiency of question search since it can greatly prune the search space by limiting search in a category. The number of questions in a leaf category is usually not exceeding 5% and thus searching in a category will be much more efficient than in the whole collection. However, as to be shown in Section 3.2.2, this simple approach is not good in terms of effectiveness even if we assume that perfect classification results could be achieved. This is because not all the relevant questions come from the same category with the category of the query question. In addition, the effectiveness of question search will highly depend on the accuracy of classifier: if the query question is not correctly classified, then the retrieval results will be poor since we will search in a wrong category.

To alleviate the aforementioned problems, we consider the probability of query $\mathbf{q}$ belonging to the category $Cat(\mathbf{d})$ of a historical question $\mathbf{d}$. The probability is denoted by $P(Cat(\mathbf{d})|\mathbf{q})$. According to Equation 9, under the condition $CLS(\mathbf{q}) = Cat(\mathbf{d})$ we have $P_{Cat}(\mathbf{q}|\mathbf{d}) = P(\mathbf{q}|\mathbf{d})$, and under the condition $CLS(\mathbf{q}) \neq Cat(\mathbf{d})$ we have $P_{Cat}(\mathbf{q}|\mathbf{d}) = 0$. Actually $P(Cat(\mathbf{d})|\mathbf{q})$ represents the probability of $CLS(\mathbf{q}) = Cat(\mathbf{d})$, and thus according to the total probability formula we have:

$$\begin{aligned} &P_{Cat}(\mathbf{q}|\mathbf{d}) \\ &= P(\mathbf{q}|\mathbf{d})P(Cat(\mathbf{d})|\mathbf{q}) + 0 \times (1 - P(Cat(\mathbf{d})|\mathbf{q})) \\ &= P(\mathbf{q}|\mathbf{d})P(Cat(\mathbf{d})|\mathbf{q}) \\ &= P(Cat(\mathbf{d})|\mathbf{q})\prod_{w \in \mathbf{q}} [(1-\lambda)P_{ml}(w|\mathbf{d}) + \lambda P_{ml}(w|Coll)] \end{aligned} \tag{10}$$

where $P(Cat(\mathbf{d})|\mathbf{q})$ is computed by classification model (to be discussed in Section 2.6). For a query question, the classification model can return the probability of the query belonging to each category.

Equation 10 suggests a way to rank questions by combining the query classification probability and the language model. We call this model LM+QC.

In this model, the ranking of a historical question $\mathbf{d}$ will be promoted if the probability of the query question $\mathbf{q}$ belonging to the category $Cat(\mathbf{d})$ of question $\mathbf{d}$ is high.

2.4 Language model enhanced with question classification and category smoothing

The approach LM+L in Section 2.2 establishes the connection of a query and a category by smoothing with category language model, while the approach LM+QC in Section 2.3 establishes the connection of a query and a category by classifying the query.

This section will present a new model combining the models LM+L and LM+QC. It will benefit from both the two models. That is we enhance question language model using both category language model and query question classification for question search. In this model we will compute $P(\mathbf{q}|\mathbf{d})$ in Equation 10 using Equation 2, finally we have the following model:

$$\begin{aligned} P_{Cat}(\mathbf{q}|\mathbf{d}) = P(Cat(\mathbf{d})|\mathbf{q}) \prod_{w \in \mathbf{q}} [(1-\lambda)P_{ml}(w|\mathbf{d}) \\ + \lambda((1-\beta)P_{ml}(w|Cat(\mathbf{d})) + \beta P_{ml}(w|Coll))] \end{aligned} \tag{11}$$

where $P(Cat(\mathbf{d})|\mathbf{q})$ is the same with that in the model LM+QC. We call this model LM+LQC.

2.5 Pruning search space using query classification

Efficiency is important in question search since the question archive of popular community QA is huge and it keeps growing. [2]

The results of query question classification are used to distinguish historical questions across different categories to improve the performance in the models LM+QC and LM+LQC. Actually query classification can also help to prune the question search space and save the runtime when efficiency is a main concern. We can introduce a threshold ξ and prune the categories if the probability of $\mathbf{q}$ belonging to them is smaller than the threshold ξ. In other words, the different models including the baseline LM, LM+L, LM+QC, and LM+LQC will search questions only in the categories such that the probability of $\mathbf{q}$ belonging to them is larger than ξ.

However, the pruning might deteriorate the effectiveness of question search while saving the runtime. The balance

[2] The Community QA in Baidu ZhiDao has more than 62.1 million resolved questions as of Aug. 18 2009

between effectiveness and the efficiency will be evaluated in Section 3.2.4.

2.6 Hierarchical classification

Hierarchical classification approach has been shown to be more efficient and usually more effective than a flat classification model[8]. We use a top-down hierarchical classification approach [8] to build a hierarchical classification model for classifying new questions. Top-down hierarchical classification approach first learns to distinguish among categories at the top level, then lower level distinctions are learned only within the appropriate top level of the tree. Given a new question $\mathbf{q}$ to be classified, the hierarchical classification approach will traverse the classification tree, starting from the root node of the classification model tree; at each node, it check if the probability of $\mathbf{q}$ belong to the category at the node is larger than a threshold ζ: if it is, it will assign a probability to each category (child node) in the current node; otherwise the subtree rooted at the node will not be traversed.

Given a question $\mathbf{q}$ and a category Cat, the probability of $\mathbf{q}$ belonging to category Cat is defined as $P(Cat|\mathbf{q})$, and can be 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 $P(\mathbf{q}|c_i) > \zeta, \forall c_i \in Path(Cat)$.

Some leaf categories might not be traversed due to the threshold ζ. In the LM+QC and LM+LQC models we will assign an untraversed category the probability of its nearest ancestor which has been traversed.

The classification task is essentially a text classification problem and using bags of words as features works well in text classification [21]. Hence we treat each question as a bag of words. There has been a lot of work on question classification in Question Answering, e.g. [23], which classifies questions (mainly factoid questions) into a number of categories mainly based on expected answer types, e.g. number category (the expected answer is a number). Although the proposed techniques there could be adapted for our classification task, we conjecture that they may not fit well our text classification task since the classification task considered in Question Answering is different.

3. EVALUATION

3.1 Experimental setup

3.1.1 Classification model:

To obtain classification information of each query question, we employ hierarchical top-down method using the approach in [8], and the threshold ζ parameter is set at 0.01.

3.1.2 Question search models:

We evaluate the baseline approach Language Model (LM) and the category based retrieval approaches, namely Language Model with leaf category smoothing (LM+L), Language Model with query classification (LM+QC), Language Model enhanced with question classification (LM+LQC). We also compare with the other two models that are briefly mentioned in Section 2.3, namely search in the Top-1 category determined by classifier (LM@Top1C) using LM and search in the correct category specified by users of Yahoo! Answers (LM@OptC) using LM. We also report the results of Vector Space Model (VSM), Okapi Model and Translation Model (TR) which have been used for question retrieval in the previous work [12] (we use GIZA++ [3] for training the word translation probabilities). Note that they are reported as references and the main purpose of this experimental study is to *see whether the three proposed category based approaches can improve the performance of baseline LM, i.e. whether the category information is indeed useful for question search.*

3.1.3 Data set:

We collected questions in all categories from Yahoo! Answers, and then divided the questions randomly into two data sets. The division maintains the distributions of questions in all categories. We get a data set containing 3,116,147 questions as the training data for classification and also the *question repository* for question search. We also get another data set containing 127,202 questions, which is used as the test set for classification. Note that we use a much larger question repository than those used in previous work for question search, and a large real data is expected to better reflect the real application scenario of 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. Figure 2 shows the distribution of questions in the training data set on first-level category. There are 26 categories in the first level and 1263 categories in the leaf level. Each question belongs to a unique leaf category.

We randomly select 300 questions from the test set (127,202 questions). 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 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 person will make judgement for the final result. We eliminate the query questions that do not have relevant questions. Finally we get 252 queries which have relevant questions and they are used as *query set.* [4]

3.1.4 Metrics:

We evaluate the performance of our approaches using Mean Average Precision (MAP), Mean reciprocal rank (MRR), R-Precision and Precision@n. MAP rewards the approach returning relevant questions earlier, and also emphases the rank in returned list. MRR gives us an idea of how far down we must look in the ranked list in order to find a relevant question. 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

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

[4] The query set is available in
http://homepages.inf.ed.ac.uk/gcong/qa/.

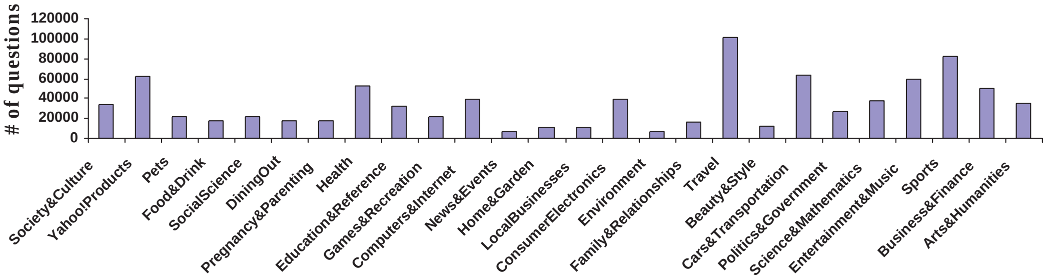

Figure 2: Distribution of Questions in the First-Level Categories in Yahoo! Answer

to compute some of metrics that we used. Finally, we use the tool *trec_eval* from TREC [1] to compute all kinds of metrics.

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

3.1.5 Parameter selection:

In our experiments, we need two smoothing parameters. Table 1 shows the results on a small data set, which contains 20 queries, when we vary both parameters to determine an optimal value in terms of MAP using LM+L. Finally we set λ to 0.2 and β to 0.2. It is also shown in [26] that 0.2 is a good choice for the parameter λ. In the Okapi Model, we follow the work presented in [20] to select the parameter.

λ β	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

Table 1: Parameter Selection (MAP)

3.2 Experimental result

In this subsection, we will report the results of different question search approaches on different metrics. We will also study the effect of pruning question search space on both efficiency and effectiveness. Before reporting question search results, we first present classification results.

3.2.1 Classification results:

The classification results provide indispensable information for the approaches utilizing question classification, including LM+QC, LM+LQC, and LM@Top1C. Note that approach LM+L does not need question classification. Table 3 gives the Micro. F-scores for both hierarchical classification model and flat classification model. We do not see significant improvement of hierarchical classification over flat classification on the performance. However, we note that the classification time of hierarchical classification model is usually only about 40% of the time of flat classification.

The approaches LM+QC and LM+LQC use not only the Top-1 returned category, but also other returned categories. Even if the correct category is not the Top-1 returned category, our question search approach may still benefit from classification information if the correct category is contained in Top-n returned categories. In order to see if the correct category is contained in 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 as "Success@n". It is shown in Figure 3. We can see that the correct categories of about 75% questions are in the Top-10 categories returned by the classifier

	Flat	Hierarchical
Micro. F1 score	44.84	45.59

Table 3: Performance of two classification models

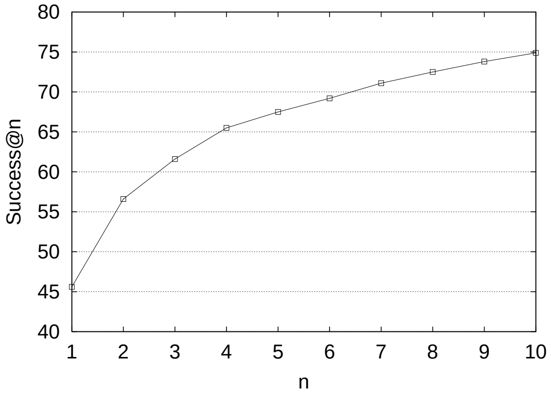

Figure 3: Success@n: the percentage of questions whose correct categories are contained in Top-n returned categories by the classifier

3.2.2 Question search results:

Table 2 shows the performance of different models in terms of different metrics, including MAP, R-Precision, MRR and Precision@n. Each column of Table 2 corresponds to an approach.

As shown in Table 2, LM+QC has better performance than the baseline method. LM+L and LM+LQC (utilizing category information) achieve statistically significant improvement (p-value < 0.05) compared with the baseline model LM in terms of all the metrics. This clearly shows that the category information indeed is able to improve the performance of question search. We can also see in Table 2 that

[5] http://en.wikipedia.org/wiki/F_score

	VSM	Okapi	TR	Baseline: LM	LM@Top1C	LM@OptC	LM+QC	LM+L	LM+LQC	%chg
MAP	0.2457	0.3423	0.4053	0.3879	0.2909	0.3478	0.4112	0.4646*	**0.4704***	21.3
MRR	0.4453	0.5406	0.6084	0.5945	0.4469	0.5219	0.6083	0.6620*	**0.6649***	11.8
R-Prec	0.2346	0.3204	0.3771	0.3459	0.2773	0.3197	0.3675	0.4153*	**0.4205***	21.6
P@5	0.2222	0.2857	0.3168	0.3040	0.2460	0.2810	0.3230	0.3460*	**0.3476***	14.3
P@10	0.1683	0.2242	0.2438	0.2310	0.1937	0.2155	0.2496	0.2599*	**0.2623***	13.5

Table 2: Performance of Different Approaches Measured by Different Metrics (* means statistically significant improvement compared with the baseline LM, (p-value < 0.05, t-test), and %chg is the promotion of LM+LQC compared with LM)

baseline approach LM outperforms VSM and Okapi, and performs slightly worse than TR.

Among the three approaches (LM@Top1C, LM@OptC and LM+QC) leveraging classification results, LM+QC performs better than the other two. We can see that LM@Top1C performs very poorly. This is because this model searches relevant question only from the questions in the Top-1 category determined by the classifier and highly depends on the reliability of classification results. However, LM@OptC is also not good although it is better than LM@Top1C. This indicates that even if we could have a prefect classifier, the approach LM@Top1C still cannot achieve much better results. The reason is that besides the category containing the query question, other categories also contain relevant questions and this model misses the relevant questions in other categories. LM+QC performs much better because it uses classification probability to adjust the ranking of questions and finds relevant questions in all categories, but not only in the Top-1 category.

LM+L model significantly outperforms the baseline approach LM. It is also shown that LM+L is effective than LM+QC in our experiments. This indicates that smoothing by category information helps more than query categorization in our models. LM+LQC model outperforms all the other approaches on all metrics. Recall that LM+LQC enhances language model by combining the LM+L and LM+QC models. We also note that LM+LQC only slightly improve LM+L and LM+QC. This is perhaps because both the two models, LM+L and LM+QC, utilize the category information, though in different ways, and it could not gain much from the combination.

In Table 2 we also give the result TransLM. It performs good, but still worse than LM+L and LM+LQC because this model does not consider the category information.

3.2.3 Result analysis for LM+L and LM+QC:

We scrutinize the results to understand the effect of category information on performance in our models. Because LM+LQC is a combination of the LM+L model and the LM+QC model, we will only analyze the results of the two models, respectively. In this section we determine whether our models outperforms the baseline LM for a specific query based on the metric MAP. The analysis based on other matric will be qualitatively comparable.

As discussed in Section 2.2, the LM+L model improves the baseline model LM from two aspects. First leaf smoothing in the LM+L model enables the category-specific frequent words to play more important role in distinguishing questions across categories. It will promote the rankings of historic questions from categories that are relevant to a query question. We find that about 90 query questions (out of 252) benefit from this. Second leaf smoothing makes the rankings of questions from the same category more reasonable since it plays a role like IDF in distinguishing questions within a specific category. About 85 query questions benefit from this. Note that the performance of some queries is improved due to both of the two reasons.

We also notice that the LM+L performs worse than the baseline model LM on 45 queries. We investigate these cases and find the following reasons. Note that the performance of some queries is affected due to more than one reason.

1) Relevant questions come from the categories whose topics are not very relevant to query questions. The LM+L model will demote the rankings of the questions from "non-relevant categories", thus if those categories contain relevant questions the performance becomes worse than the baseline. One reason for the problem is that a question may be submitted to a wrong category by the Yahoo! users. Another reason is that many categories have an "Other" subcategory which may contain relevant questions, but such categories cover various topics and are not quite relevant to the queries, and thus LM+L fails. We find that about 10 queries are affected by this.

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

3) Although the leaf smoothing usually helps to rank questions from the same category since it plays a role like IDF computed with regard to a category, it may also lead to poor performance on some queries. For some queries the smoothing with regard to the whole collection better describes the importance of words than the category. Hence, the smoothing in the baseline LM, which plays a role like IDF with regard to the whole collection, may perform better than the leaf smoothing. We find that about 25 queries are affected by this.

We proceed to analyze the results of LM+QC. This model benefits from the query classification. From Equation 10 we can see that the rankings of historical questions from categories with high classification probability scores will be promoted. To further see if the promotion is useful, we compute some statistics as follows. For each query, we rank all the categories according to classification results, and count the number of relevant questions in every rank. We then aggregate the number of relevant questions in each rank across all queries, and compute the percentage of relevant questions in

Models	Rank	Questions	Categories
LM	1st	How can you talk a mom in to letting you get a motorcycle ?	Motorcycles
	12th	**I am trying to get my mom to get me a corn snake What should i do?**	Reptiles
	13th	**How do I get my mom to let me have a rabbit? She still wouldn't let me have a snake.?**	Other - Pets
LM+L	1st	How can you talk a mom in to letting you get a motorcycle ?	Motorcycles
	7th	**I am trying to get my mom to get me a corn snake What should i do?**	Reptiles
	8th	**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	1st	**I am trying to get my mom to get me a corn snake What should i do?**	Reptiles
	2nd	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
	3rd	**How do I get my mom to let me have a rabbit? She still wouldn't let me have a snake.?**	Other - Pets

Table 4: Search results for "Wat is the best way to talk my mom into letting me get a snake???"

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+L	1	**How to Trim a Bird's Beak ?**	Birds
	4	My parakeets beak is bruised ,what should I do?	Birds
LM+QC	1	My parakeets beak is bruised ,what should I do?	Birds
	2	**How to Trim a Bird's Beak ?**	Birds

Table 5: Search results for "How can I trim my parakeets beak?"

each rank. Table 6 gives the results. It shows that about 86% 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 rankings of questions from top-ranked categories. In the query set about 120 queries have been improved by query classification. However for some queries which have obvious category features, most of the top-ranked historical questions in the results of LM are already from the correct category and thus LM+QC fails to improve the retrieval.

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

Table 6: Distribution of relevant questions in different ranks of category (LM results)

LM+QC performs worse than baseline for 52 queries. The reasons are as follows:

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 will perform poorly. We notice that for some queries LM+QC performs even worse than the baseline when the correct category of the query question is not contained in its Top-10 classification results.

2) The second reason is the same as the first reason for the failure of the LM+L model as we just analyzed. The LM+QC model also performs worse than baseline LM when relevant questions come from "non-relevant" categories.

To have a better understanding of our models, Table 4 gives part of the results of an example query question "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+L model the category-specific frequent word "snake" makes the category "Reptiles" and "Other-Pets" more relevant to the query through the leaf smoothing. Hence the rankings of the questions in the two categories are promoted (from 12th to 7th and 13th to 8th, respectively). In the LM+QC model the category "Reptiles" and "Other-Pets" are the Top-2 categories judged by the classifier, and thus the rankings of questions in the two categories are promoted by LM+QC model (from 12th to 1st and 13th to 3rd, respectively).

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

3.2.4 *Efficiency of pruning search space:*

In addition to retrieval effectiveness, efficiency is also very important considering that the size of CQA repository is huge and it keeps increasing. In Section 2.5 we proposed a method of utilizing query classification to improve the 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 running time of question search, though it might deteriorate the effectiveness of question search. Note that the classification time is very little compared with the question search time.

We next evaluate the effect of threshold pruning using models LM, LM+L, LM+QC and LM+LQC on both effectiveness and 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 the runtime of the base-

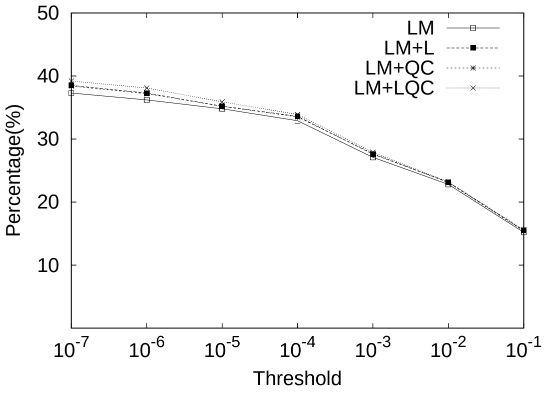

Figure 4: Relative running time of different models using different pruning thresholds compared with the baseline model LM

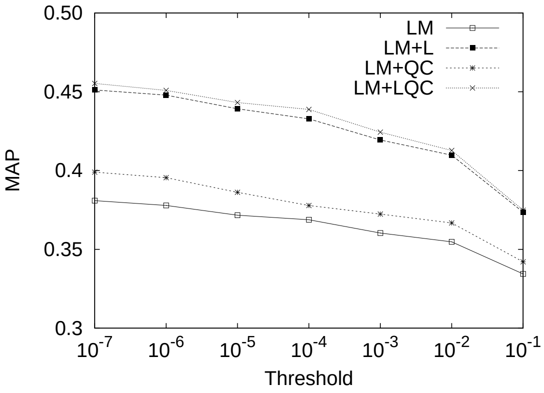

Figure 5: Performance on MAP using different pruning thresholds

line model LM is shown in Figure 4, and the MAP results is shown in Figure 5.

From Figure 4 and Figure 5, we can see that the pruning deteriorates the retrieval effectiveness while it can greatly improve efficiency for all the four models. For example, the saving is about 85% when threshold ξ is set at 0.1 for all of them. As the threshold becomes smaller, the MAP result becomes better while the saving in runtime becomes less. However, even when the threshold ξ is set at 10^{-7}, for LM+LQC the runtime is still 39% of the original runtime of the baseline while the MAP value is 0.455. LM+LQC with pruning still achieves better MAP result than LM+QC without pruning, and gains significant improvements than LM in terms of MAP, MRR, R-Precision and P5.

The baseline model LM is the most efficient, and LM+QC and LM+L are a bit slower than LM, because LM+QC needs to multiply the query classification results of a category and LM+L needs one more level smoothing. LM+LQC is the slowest model because it needs not only to multiply the query classification results but also the leaf smoothing. But it performs the best on all the threshold values. LM+L and LM+QC also performs better than the baseline model LM.

Hence, the threshold based pruning offers us an option to make a compromise between the effectiveness and the efficiency. While efficiency is not the main concern, we can simply ignore the parameter.

As a summary, the experimental results show that the category information can be utilized to significantly improve the performance of question search in the proposed models, LM+L and LM+LQC. Moreover, we can greatly save running time by pruning search space.

4. RELATED WORK

Question Search. There has been a host of work on question search. Most of the existing work focuses on addressing the word mismatching problem between user questions and the questions in a QA archive. Burke et al. [4] combine lexical similarity 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. [2] study several statistical approaches to bridge the lexical gap for FAQ retrieval. Jijkoun and Rijke [13] use supervised learning methods to extract QA pairs from FAQ pages, and use a vector space model for question-answer pair retrieval. Riezler et al. [19] 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 [22] 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. [11, 12] compare four different retrieval methods, i.e., vector space model, Okapi, language model, and translation-based model, for question search on CQA data. In subsequent work [25], they propose a translation-based language model, and also exploit answers for question search. Duan et al. [7] build a structure tree for questions in a category of Yahoo! Answers and then find important phrases in questions that are given higher weight in question matching. Bian et al. [3] propose a learning framework for factual information retrieval in CQA data. The approach needs training data, which is unfortunately difficult to get. Wang et al. [24] parse the questions into syntactic trees and use the similarity of the syntactic trees of the query question and the historical question to rank historical questions.

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. To the best of our knowledge, no previous work attempts to utilize category information to improve question search, although this appears to be a natural idea.

Cluster-based document retrieval. In cluster-based retrieval, documents are grouped into clusters, and an a list of documents is returned based on the clusters that they come from. We can roughly classify cluster-based retrieval into two groups. The first group of approaches retrieve one or more clusters in their entirety in response to a query. For example, in early work [10], entire documents are clustered, and clusters are retrieved based on how well their centroids match a given query. In the second group, ranking scores of individual documents in a cluster are computed against a query, e.g., [6, 9, 14, 16]. No existing cluster-based retrieval approaches have been applied to question search.

Classification for document retrieval. Chekuri and Raghavan [5] introduce the idea of utilizing classification for document retrieval. However, their focus is on the automatic classification of Web documents into high-level categories of the Yahoo! taxonomy; they do not consider how to leverage the classification results for document retrieval. Lam et al. [15] develop an automatic text categorization approach to classify both documents and queries, and investigate the application of text categorization to text retrieval. Our approaches are very different from the approaches in existing work for document retrieval.

5. CONCLUSIONS AND FUTURE WORK

Question search is an essential component in Community Question Answer (CQA) services. In this paper, we propose a new framework that is capable of exploiting classifications of questions in CQA archives for improving question search. In doing so, the framework uses specific language models. It uses a local smoothing technique to smooth a question language model with category language models to exploit the characteristics of the categories. It also incorporates a technique that computes the probabilities of a user question belonging to each existing category and integrates the probabilities into the langauge model. Experiments conducted on a large QA data set from Yahoo! Answers demonstrate the effectiveness of the proposed framework.

Several promising directions for future work exist. First, it is of relevance to apply and evaluate other question search approaches, e.g., translation models (see the coverage of related work for more approaches), within the proposed framework. Such approaches can be integrated into the framework. Second, it is relevant to evaluate the efficiency of the proposed approach more comprehensively. Third, additional empirical studies of the performance of question search across categories are in order. Fourth, we believe that it may be possible to optimize the classification performance by combining or splitting categories so that similar subcategories are combined and incoherent categories are split. Finally, we believe it is interesting to consider answers into our framework.

6. ACKNOWLEDGEMENT

We would like to thank the anonymous reviewers for their constructive suggestions.

7. REFERENCES

[1] *TREC*. http://trec.nist.gov/.

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

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

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

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

[6] F. Diaz. Regularizing ad hoc retrieval scores. In *CIKM*, pages 672–679, 2005.

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

[8] S. T. Dumais and H. Chen. Hierarchical classification of web content. In *SIGIR*, pages 256–263, 2000.

[9] M. A. Hearst and J. O. Pedersen. Reexamining the cluster hypothesis: Scatter/gather on retrieval results. In *SIGIR*, pages 76–84, 1996.

[10] N. Jardine and C. van Rijsbergen. The use of hierarchical clustering in information retrieval. *Information Storage and Retrieval*, 7:217–240, 1971.

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

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

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

[14] O. Kurland and L. Lee. Corpus structure, language models, and ad hoc information retrieval. In *SIGIR*, pages 194–201, 2004.

[15] W. Lam, M. E. Ruiz, and P. Srinivasan. Automatic text categorization and its application to text retrieval. *IEEE Trans. Knowl. Data Eng.*, 11(6):865–879, 1999.

[16] X. Liu and W. B. Croft. Cluster-based retrieval using language models. In *SIGIR*, pages 186–193, 2004.

[17] D. Petkova and W. B. Croft. Hierarchical language models for expert finding in enterprise corpora. *ICTAI*, pages 599–608, 2006.

[18] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *SIGIR*, pages 275–281, 1998.

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

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

[21] F. Sebastiani. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47, 2002.

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

[23] E. M. Voorhees. Overview of the trec 2001 question answering track. In *TREC*, pages 42–51, 2001.

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

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

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