# Yahoo! Answers for Sentence Retrieval in Question Answering

**Saeedeh Momtazi, Dietrich Klakow**

Spoken Language Systems
Saarland University, Saarbrücken, Germany
{saeedeh.momtazi,dietrich.klakow}@lsv.uni-saarland.de

**Abstract**

Question answering systems which automatically search for user's information need are considered as a separate issue from the community-generated question answering which answers users' questions by human respondents. Although the two answering systems have different applications, both of them aim to present a correct answer to the users' question and consequently they can feed each other to improve their performance and efficiency. In this paper, we propose a new idea to use the information derived from a community question answering forum in an automatic question answering system. To this end, two different frameworks, namely the class-based model and the trained trigger model, have been used in a language model-based sentence retrieval system. Both models try to capture word relationships from the question-answer sentence pair of a community forum. Using a standard TREC question answering dataset, we evaluate our proposed models on the subtask of sentence retrieval, while training the models on the *Yahoo! Answer* corpus. Results show both methods that trained on Yahoo! Answers logs significantly outperform the unigram model, in which the class-based model achieved 4.72% relative improvement in mean average precision and the trained triggering model achieved 18.10% relative improvement in the same evaluation metric. Combination of both proposed models also improved the system mean average precision 19.29%.

## 1. Introduction

While information retrieval historically focuses on searching for relevant documents, it is often the case that only a portion of a *relevant* document is related to a user's information need. In such a situation, it may be preferable instead to retrieve only the relevant portion of the document which includes the information that the user requires. Such an idea has recently motivated researchers to develop question answering systems which retrieve the exact information required by the users.

Within a question answering system, document retrieval is an important component which should provide a list of candidate documents to be analyzed by the rest of the system. Document retrieval, however, is insufficient, as the retrieved documents are much larger than the required answer, and topic changes typically occur within a single document. In addition, in the question answering context, the relevant information is most often found in one sentence or two. Hence, it is essential to split the text into smaller segments, such as sentences, and rank them in a sentence retrieval step. This is the focus of our research. Retrieved sentences are then further processed using a variety of techniques to extract the final answers. It has been proven that this component has an important role in a question answering system such that improvement in sentence retrieval performance has a significant positive effect on the accuracy of the question answering system (Shen, 2008). Figure 1 shows a simple structure of a question answering system considering both levels of the information retrieval component and also the information extraction component.

Although available retrieval methods used for document retrieval are also applicable for the task of sentence retrieval, the performance of these models in the sentence retrieval task is worse than for the task of retrieving documents. Because there are major differences between document retrieval and sentence retrieval which affect their performance. As a result, many of the assumptions made about

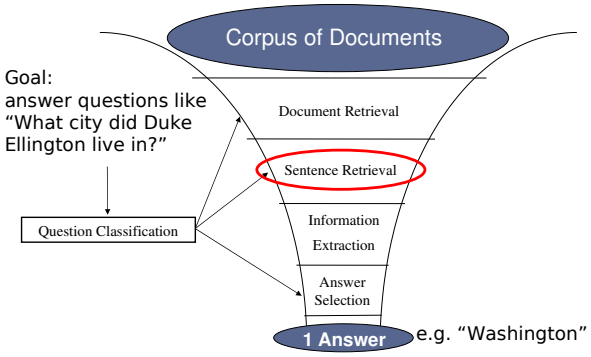


Figure 1: The Structure of a Question Answering System.

document retrieval do not hold for sentence retrieval (Murdock, 2006). The brevity of sentences vs. documents is the most important feature that exacerbates term-mismatch problems.

In the common retrieval methods, a search is performed for only the exact literal words presented in the query. Consequently, these algorithms fail to retrieve other relevant information. For example, consider the sample question in Table 1 and its correct answer sentences. The sentence retrieval component might retrieve the first sentence, because there are two shared terms, *"invented"* and *"car"*, between the question and this sentence. Using the exact matching models, however, the retrieval algorithm misses the second and the third sentences, because these sentences do not share any term with the question. In other words, although these sentences contain the words *"built"*, *"vehicle"*, and *"automobile"* which are very likely to be relevant to the question terms, the sentence retrieval model is not able to recognize their relationship.

While different approaches such as automatic query expansion have been a great success story for solving the term-

Table 1: A sample question and possible answer sentences in a search space

| | |
|---|---|
| Question | *"Who invented the car?"* |
| Answer | *"Between 1832 and 1839, Robert Anderson of Scotland invented the first crude electric car carriage."* |
| Answer | *"Nicolas-Joseph Cugnot built the first self propelled mechanical vehicle."* |
| Answer | *"An automobile powered by his own engine was built by Karl Benz in 1885 and granted a patent."* |

mismatch problem in document retrieval, attempts to apply them to sentence retrieval have had rather mixed success (Murdock, 2006). For this reason, it is desirable to have a more sophisticated model to capture the semantics of sentences rather than just the term distributions. This issue has motivated a great deal of research on term relationships over the last decades. However, improvements in system performance from such schemas have proven challenging, for two primary reasons: the difficulty of estimating term relationships and the difficulty of integrating both exact match and term relationships in a single weighting schema (Gao et al., 2004).

Various research has been done on estimating of term relationships for information retrieval, as will be described in Section 2.2. In the task of question answering, however, it is more difficult to find relevant sentences. This is due to the fact that there are so many sentences in the search space that are relevant to the question, but do not include the answer. In this case, it is necessary to find a novel information resource which is closer to the question answering purpose. We believe that although community-generated question answering and automatic question answering systems use two separate approaches, the community question answering forums that collect answers from human respondents are a useful resource that can be exploited to the term-mismatch problem of the automatic systems. Generally, community-generated question answering forums provide very informative corpora that contain pairs of question-answer sentences. As the question and its relevant answer sentences typically talk about the same topic, there is a latent relation between the question words and the terms appearing in the answer sentences, even though there are not many common terms between the pair of the question and the answer sentences. In this paper, we propose a novel approach to use community question answering logs in two different language model-based frameworks, namely class-based model and trained triggering model, and apply them to the sentence retrieval task.

The paper is organized as follows. In Section 2 we review related work using both the general language modeling approaches for information retrieval and term relationship approaches. Section 3 and 4 describe the class-based and trained triggering models that we use to capture word relation from community question answering logs. The dataset, corpus, and experimental results are presented in Section 5. Finally, Section 6 summarizes the paper.

# 2. Related Work

## 2.1. Language Models for Information Retrieval

Statistical language modeling has successfully been used in speech recognition (Jelinek, 1998) and many natural language processing tasks including part of speech tagging, syntactic parsing (Charniak, 1993), and machine translation (Brown et al., 1990).

Language model for information retrieval has received researchers' attention during the recent years. The efficiency of this approach, its simplicity, the state-of-the-art performance, and clear probabilistic meaning are the most important factors which contribute to its popularity (Lafferty and Zhai, 2001; Ponte and Croft, 1998).

The idea of using language model techniques for information retrieval applications was proposed by Ponte and Croft (1998). They inferred a language model for each document and estimated the probability of generating the query according to each of these models. In their method, each query is considered as a set of unique terms and two different probabilities are computed. The first one is the probability of producing the query terms; and the second one is the probability of not producing other terms. Then they use the product of these two factors as their model. In addition, Hiemstra (1998) considered each query as a sequence of terms and computed the query probability by multiplying the probability of each individual term. Song and Croft (1999), and Miller (1999) also used the same method.

Berger and Lafferty (1999) proposed a translation-based approach which computes the probability of generating a query as a translation of a document. They utilized this probability as a measure of relevance of a document to a query to rank the documents. Following this method, Lafferty and Zhai (2001) proposed another technique to extend their current model by estimating the language models of both documents and queries. In this approach, the language models of documents and queries are computed and then the Kullback-Leibler divergence between the probabilities of document model and query model is used.

Zhai and Lafferty (2001) estimated the conditional probability $P(D|Q)$ by applying the Bayes' formula and dropping a document-independent constant:

$$P(D|Q) \propto P(Q|D)P(D) \tag{1}$$

where $P(Q|D)$ is the probability of the query given a document and $P(D)$ is the prior probability of a document. Since $P(D)$ is assumed to be uniform for ranking the documents, it will be ignored in further computations.

$P(Q|D)$ is the probability of generating a query $Q$ given the observation of the document $D$; and the documents are ranked in descending order of this probability. The word-based unigram model estimates the probability of generating the query by:

$$P(Q|D) = \prod_{i=1\ldots M} P(q_i|D) \tag{2}$$

where $M$ is the number of terms in the query, $q_i$ denotes the $i^{th}$ term of query $Q$, and $D$ is the document model (Song and Croft, 1999). Merkel and Klakow (2007) used the same model for the sentence retrieval task. Since in this case

the documents are divided into sentences, $P(Q|S)$ is computed; where *S* is the sentence to be ranked. The unigram model for sentence retrieval computes the probability of the query *Q* given the sentence *S* by:

$$P(Q|S) = \prod_{i=1...M} P(q_i|S) \quad (3)$$

### 2.2. Term Relationship Models for Information Retrieval

As mentioned in the previous section, by estimating the word-based unigram model, the ranking algorithms only try to match the literal words that are present in queries and texts; but they will fail to retrieve other relevant information. To avoid this problem, researchers have tried to apply methods using different types of term relationships to the retrieval task.

Using hand-crafted thesauri such as WordNet is one of the prevalent techniques (Mandala et al., 1998; Schütze and Pedersen, 1997). The thesaurus is incorporated in different retrieval models to find the dependencies among the words. Robertson et al. (1981) used a thesaurus for the probabilistic retrieval model; Cao et al. (2005; 2006), and Croft and Wei (2005) applied it for the language model-based retrieval. Voorhees (1994) used WordNet for query expansion; and Liu et al. (2004) used this approach to disambiguate word senses by expanding the query. Since the results of using manual processing provide useful information, it seems like a good method for our goal. However, manual processing causes many problems such as inconsistency and ambiguity (Jones, 1971). The absence of proper nouns causes another problem; since WordNet and most of the thesauri do not consider proper nouns and we cannot find relationship between these nouns and other terms as a result. In addition, we can not find the measure of dependency between the terms in the manual processing, since they only give us a binary classification of relevant and non-relevant terms to an special term. Beside these problems, building a thesaurus is labor intensive. Because of such problems, an automatic method is more desirable.

Grammatical analysis has also been applied as an automatic approach to find dependencies between terms in a sentence. Nallapati and Allan (2002) introduced a new probabilistic sentence tree language model approach. Gao et al. (2004) described a linkage model to relax the independence assumption. Although grammatical analyses can provide very specific knowledge about term relations, they are not robust (Manning et al., 2008) and also need a deep sentence processing.

The use of co-occurrence statistics is another well-known method which focuses on term relations. Cao et al. (2005) used the co-occurrence relations and integrated them with the relations extracted from WordNet. Wei and Croft (2007) introduced a probabilistic term association measure and utilized this measure in document language models. Van Rejsbergen. (1979), and Burgess et al. (1998) also used words co-occurrence in window scaling. Qui and Frei (1993) applied another similar method to expand a query. In their proposed method, each new query term takes the same weight as its similarity to the original query term. Chung and Chen (2002) described another technique called correlation-verification smoothing to find correlations among terms. Since the term co-occurrence method is a window-based approach, finding a suitable window size automatically is not easy (Wei and Croft, 2007).

For applying term relations, some researchers also tried to use document reformulation. Cluster-based document models (Liu and Croft, 2004; Tao et al., 2006) and LDA-based document models (Wei and Croft, 2006) are two important models in this area. They are both expensive, especially for large collections.

Momtazi and Klakow (2009) proposed the class-based language model by applying term clustering. This model is found to be effective in capturing relevant terms. The flexibility of this model in using different types of word co-occurrence (Momtazi et al., 2010) offers a distinct advantage as it is also adaptable for question-answer pair co-occurrence which is our goal.

Trained trigger language model is another approach recently proposed for sentence retrieval and proven to outperform the unigram model. As this model can also be trained on a question-answer pair corpus, it is a useful framework for our task. In the next sections we will describe both class-based and trained trigger models in more detail.

## 3. Class-based Model

The idea of class-based model is clustering similar words together to reduce the term-mismatch problem. Partitioning vocabulary into a set of word clusters, the sentence retrieval engine can retrieve sentences which do not contain question words, but their terms are in the same clusters as question words.

As mentioned, in the basic language model-based sentence retrieval, the word-based model, $P(Q|S)$ is estimated based on the probability of generating each query term $q_i$ conditioned on a candidate sentence $S$. In class-based unigrams, $P(Q|S)$ is computed using only the *cluster labels* of the query terms as

$$P(Q|S) = \prod_{i=1...M} P(q_i|C_{q_i}, S)P(C_{q_i}|S), \quad (4)$$

where $C_{q_i}$ is the cluster containing $q_i$ and $P(q_i|C_{q_i}, S)$ is the emission probability of the $i^{th}$ query term given its cluster and the sentence. $P(C_{q_i}|S)$ is analogous to the sentence model $P(q_i|S)$ in (3); however in this model, the probability is calculated based on clusters instead of terms. To calculate $P(C_{q_i}|S)$, each cluster is considered an atomic entity, with $Q$ and $S$ interpreted as sequences of these entities (Momtazi and Klakow, 2009).

To cluster lexical items, we use the algorithm proposed by Brown et al (1992), as implemented in the SRILM toolkit (Stolcke, 2002). The Brown algorithm uses mutual information between cluster pairs in a bottom-up approach to maximize *Average Mutual Information* between adjacent clusters. Algorithm 1 shows the details of the Brown clustering.

The algorithm requires an input corpus statistics in the form $\langle w, w', f_{ww'} \rangle$, where $f_{ww'}$ is the number of times the word $w'$ is seen in the context $w$. Both $w$ and $w'$ are assumed to come from a common vocabulary.

**Algorithm 1** The Brown Word Clustering Algorithm (AMI stands for Average Mutual Information)

**Initial Mapping:** Put a single word in each cluster
Compute the initial AMI of the collection
**repeat**
  Merge a pair of clusters which has the minimum decrement of AMI
  Compute AMI of the new collection
**until** reach the predefined number of clusters K
**repeat**
  Move each term to the cluster for which the resulting partition has the greatest AMI
**until** no more increment in AMI

As shown in Algorithm 1, the clusters are initialized with a single term. Then, a bottom-up approach is used to merge the pair of clusters that minimizes the loss in average mutual information between the word cluster $C_{w'}$ and its context cluster $C_w$. Different words seen in the same contexts are good candidates for the merger, as there are different contexts in which the same words are seen. This step continues for $V-K$ iterations, where $V$ is the number of terms and $K$ is the predefined number of clusters. To increase the average mutual information, a final step is performed, whereby each term is moved to that cluster for which the resulting partition has the greatest average mutual information. The algorithm terminates when average mutual information ceases to increase.

While originally proposed with bigram statistics, the algorithm is *agnostic* to the definition of co-occurrence and several notions of co-occurrence can be used to cluster words (Momtazi et al., 2010). For example if $\langle w, w' \rangle$ are adjacent words, the algorithm clusters words based on their surroundings terms; if $f_{ww'}$ is the number of times $w$ and $w'$ appear in the same document, it will produce semantically (or topically) related word-clusters. Since we want to apply this class-based model to the sentence retrieval for question answering system, the pair of question and answer sentences is an informative resource for this task. In this model, the questions' terms that have the same words in their answer sentence are clustered together and also the answers' terms that have the same words in their related question are clustered together.

Considering the community question answering forums as a set of question-answer sentence pair, we can say $w$ and $w'$ co-occurred if the word $w$ appears in the question and $w'$ appears in the related answer. Because if the two content words $w$ and $w'$ are seen in the pair of question and answer sentence, they are usually topically related.

Statistics of this co-occurrence may be collected in two different ways. In the first case, $f_{ww'}$ is simply the number of question-answer pairs that contain both $w$ and $w'$. Alternatively, we may want to treat each *instance* of $w'$ in an answer sentence that contains an instance of $w$ in its question to be a co-occurrence event. Therefore, if $w'$ appears three times in an answer sentence that contains two instances of $w$ in its question, the former method counts it as one co-occurrence, while the latter as six co-occurrences.

We use the latter statistic, since we are concerned with retrieving sentence sized information, wherein a repeated word is more significant.

## 4. Trained Trigger Model

The goal of the trained trigger model is using wider information to relax the exact matching assumption. The available question-answer sentence pairs is one of the most informative resource that can be used for finding pairs of trigger and target words. In this model, each word in the question triggers all of the answer words.

Such a model can retrieve sentences which have no or a few words in common with the question but their terms have frequently co-occurred with question terms in the pairs of question-answer sentences used for training the model.

As an example, consider the following question and its correct answer sentence.

Q: *How high is Everest?*
A: *Everest is 29,029 feet.*

We see the above question and the answer sentence share a very limited number of terms, the term *"Everest"* in this example. In such a situation, it is very unlikely that the basic query likelihood model ranks the correct answer on the top of the list. Because there are a lot of irrelevant sentences in the search space which contain the same word such as:

S: *Everest is located in Nepal.*
S: *Everest has two main climbing routes.*

However, the triggering model which is trained on a question-answer sentence pair corpus gives a higher score to the correct sentence because the model knows that in a large portion of questions that contain the word *"high"*, the term *"feet"* appear in the answer. As a result, in the trained model, the word *"high"* triggers the target word *"feet"*. The following sentences are some of the samples that can be found in a training corpus:

Q1: *How high is Mount Hood?*
A1: *Mount Hood is in the Cascade Mountain range and is 11,245 feet.*

Q2: *How high is Pikes peak?*
A2: *Pikes peak, Colorado At 14,110 feet, altitude sickness is a consideration when driving up this mountain.*

In the basic language model-based sentence retrieval, for each sentence in the search space a language model is trained, and then using the maximum likelihood estimation, $P(q_i|S)$ is calculated based on the frequency of query term $q_i$ in sentence $S$:

$$P(q_i|S) = \frac{c(q_i, S)}{\sum_w c(w, S)} \tag{5}$$

where $c(q_i, S)$ is the frequency of $i^{th}$ query term in sentence $S$.

In trained trigger language model, contrary to the maximum likelihood, first a single model is trained on a large

corpus, then it is being used for all of the sentences to be retrieved. The trained model is represented by a set of triples $< w, w', f_{ww'} >$, where $f_{ww'}$ is the number of times the word $w$ triggers the target word $w'$. Having such a trained model, the language model-based sentence retrieval is reduced to:

$$P(q_i|S) = \frac{1}{N} \sum_{j=1...N} P_{trigger}(q_i|s_j) \quad (6)$$

where $s_j$ is the $j^{th}$ term in the sentence and *N* is the sentence length. In this model $P(q_i|s_j)$ is calculated as follows:

$$P_{trigger}(q_i|s_j) = \frac{1}{M} f_{q_i,s_j} \quad (7)$$

where *M* is the query length and $f_{q_i,s_j}$ is the number of times the query term $q_i$ triggers the sentence word $s_j$ based on the training corpus. As in the maximum likelihood model, any smoothing method can be used. In all of our experiments we use Bayesian smoothing with Dirichlet prior (Lafferty and Zhai, 2001).

# 5. Experimental Results

## 5.1. TREC Question Answering Dataset

To evaluate our model, we used the set of questions from the TREC[1] 2006 question answering track as the test data, while the TREC 2005 set was used for development. The TREC 2006 question answering task contains 75 question-series, each on one topic, for a total of 403 factoid questions. These questions were used as queries for sentence retrieval.

Since the relevance judgments released by NIST are only at the document level (Dang et al., 2006), we required another annotated corpus for sentence-level relevance judgments. To this aim, the *Question Answer Sentence Pair* corpus of Kaisser and Lowe (2008) was used. All the documents that contain relevant sentences are from the NIST AQUAINT1 corpus.

Question answering systems typically employ sentence retrieval after initial, high quality document retrieval. To simulate this, we created a separate *search collection* for each question using all sentences from all documents that are relevant to the topic (question-series) from which the question was derived. On average, there are 17 documents / 270 sentences that are relevant to each question topic (documents which are relevant to any of 5-6 different questions in a question-series) while the number of relevant sentences to each individual question is only 4 sentences (on average). So that for each question there are several irrelevant documents: they may be relevant to another question. Furthermore, irrelevant documents to a question are relevant to a related question, and hence are typical of the false alarms that would arise if one were retrieving documents based on one of the questions. As a result, the sentence search collection is realistic, even if somewhat optimistic.

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

## 5.2. Corpus for Clustering and Triggering

To cluster lexical words for the class-based model and also train the triggering language model, we used *Yahoo! Answers Comprehensive Questions and Answers corpus [version 1.0]*. This dataset derived from *http://answers.yahoo.com/* which is a web site where people post questions and other people can answer their questions. This web site is public to any web user willing to browse or download them (Webscope, 2009).

The Yahoo! Answers corpus has been collected in 10/25/2007. It includes all the questions and their corresponding answers. The size of the corpus is 3.45 GB while containing 4,483,032 questions and their answers. In addition to the question and the answer texts, the corpus contains a small amount of meta data, i.e., which answer was selected as the best answer, and the category and the subcategory assigned to each question. No personal information is included in the corpus. Table 2 shows a sample content in the Yahoo! Answers corpus, in addition to the meta information. In our experiments, both subject and content are considered as question and all answers including the best one are used as answer sentences.

Such a dataset has been used for different task like learning and validating answer extraction models (Surdeanu et al., 2008). However, to the best knowledge of the author this dataset has not been used for the sentence retrieval in question answering systems.

For both class-based and triggering model, we used this corpus to extract the pair of words in which the first word is a question term and the second word is the answer sentence term in addition to their frequency. In the class-based model, the extracted word pairs are used as the input of Brown clustering and then the clusters are used for retrieving more relevant sentences. In the triggering model, however, the word pairs are directly used in the sentence retrieval model, in which if the first word (trigger word) appears in the question and the second word (target word) appears in a sentence, we consider that there is a relation between the question and the sentence.

## 5.3. Building the Model

We evaluated our proposed models in two scenarios: on their own and in combination with the word-based model in which each of the models were interpolated with the original word-based model (Equation 3). We believe that even though using the information derived from community question answering logs can improve the sentence retrieval performance, it is necessary to keep the regular exact matching model too. Because this original model can give priority to the words that are shared between the question and the answer sentence, as they still play an important role in the system.

In this experiment, the word-based unigram model with maximum likelihood estimation is considered as the baseline. For the class-based model, the class-based unigram with maximum likelihood estimation is calculated and for the trained triggering model, the word-based unigram with triggering estimation is used. For all of the experiments Bayesian smoothing with Dirichlet prior (Mackay and Peto, 1995) has been applied and then linear interpolation (Je-

Table 2: A sample content from Yahoo! Answers

| | |
|---|---|
| Subject | *"What are world's 3 smallest nations?"* |
| Content | *"i.e. which is the smallest nation? which is the next one in size? and then the next one? How did they manage to become countries?"* |
| Best Answer | *"1. The Vatican City in central Rome ranks as the smallest nation of the world. It measures 0.17 square miles. 2. Monaco. 3. San Marino. How did they manage to become countries? 1. The political freedom of the Vatican is guaranteed and protected by Italy. It concordat between the Holy See and the kingdom of Italy signed in 1929 in the Lateran Palace, Rome, by Cardinal Gasparri for Pius XI and by Benito Mussolini for Victor Emmanuel III. 2. Monaco came under French protection in 1861. However from 1911, when the first constitution was promulgated, the prince of Monaco was an absolute ruler. 3. According to tradition, Marino, a Christian stonecutter from Dalmatia, took refuge (4th cent) on Mt. Titano, the chief geographical feature of present-day San Marino. By the mid-5th cent., a community was formed; because of its relatively inaccessible location and its poverty, it has succeeded, with a few brief interruptions, in maintaining its independence. In 1631 its independence was recognized by the papacy."* |
| Other Answer(s) | *"In addition to the "real" nations, there are so-called "micronations" which arise when someone claims that their little chunk of land is an independent country. A good example is Sealand, a man-made structure a few miles off the coast of England once used as a lookout post during WWII. Someone took over the platform, declared it to be the Principality of Sealand, and declared himself king. Sealand is about the size of an oil drilling rig."* |
| Category | Geography |
| MainCategory | Science Mathematics |

linek and Mercer, 1989) was used to interpolate our extended models with the baseline model.
To use the interpolation of the baseline and the class-based model, the probability $q_i$ given both word- and class-based models is computed from (3) and (4) and interpolated by a weighting parameter $\lambda$.

$$P(Q|S) = \prod_{i=1\ldots M} [\lambda\ P(q_i|C_{q_i}S)P(C_{q_i}|S) + (1-\lambda)\ P(q_i|S)] \quad (8)$$

The similar interpolation model can be use for combining the baseline and the trained trigger model in which Equations 3 and 6 are used.

$$P(Q|S) = \prod_{i=1\ldots M} [\lambda\ \frac{1}{N} \sum_{j=1\ldots N} P(q_i|s_j) + (1-\lambda)\ P(q_i|S)] \quad (9)$$

The linear interpolation of all models is calculated as follows:

$$P(Q|S) = \prod_{i=1\ldots M} [\lambda_1\ P(q_i|C_{q_i}S)P(C_{q_i}|S) + \lambda_2\ \frac{1}{N} \sum_{j=1\ldots N} P(q_i|s_j) + (1-\lambda_1-\lambda_2)\ P(q_i|S)] \quad (10)$$

For the class-based model, we need to define number of clusters that the vocabulary should be partitioned into. Inasmuch as the optimum number of clusters to be used in the Brown algorithm is not self-evident, tests were conducted with several numbers of clusters. Figure 2 shows the mean average precision (MAP) of the class-based unigram for varying numbers of clusters. Setting the number of clusters to 0, the model works like the uniform model. Equating the number of clusters with the number of words is equivalent to using a word-based model. According to the results, the best mean average precision is achieved by clustering all 34,000 lexical items into 200 clusters. Hence, this value was used in all further experiments.

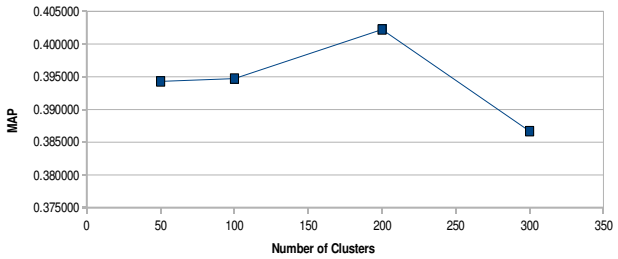


Figure 2: MAP of the class-based model over different numbers of classes

### 5.4. Results

Table 3 shows the results of our experiments for the the pure class-based and pure trained trigger models in which Mean Average Precision (MAP), Mean Reciprocal Rank (MRR), and Precision at 5 (P@5) serve as the primary metrics. Comparing the results to the baseline, we can see that the trained triggering model performs very poorly, while the MAP and P@5 in the class-based model are similar to the baseline and the class-based model outperforms the baseline in MRR.
As we expected, each of the proposed models can not perform accurately when applying individually. Hence, in the second step we used the interpolation of our models with the baseline. The results are presented in Table 4 while the interpolated models are compared with the baseline model. From this table, we observe that although the trained triggering model performed very poorly, interpolating this model with the baseline improves the sentence re-

Table 3: Retrieval results with different language modeling schemas

| Language Model | MAP | MRR | P@5 |
|---|---|---|---|
| Baseline Model | 0.3701 | 0.5047 | 0.2267 |
| Class-based Model | 0.3705 | 0.5239 | 0.2233 |
| Triggering Model | 0.0344 | 0.0415 | 0.0099 |

Table 4: Retrieval results for the linear interpolation of proposed models with the baseline

| Language Model | MAP | MRR | P@5 |
|---|---|---|---|
| Baseline Model | 0.3701 | 0.5047 | 0.2267 |
| + Class-based Model | 0.3876 | 0.5368 | 0.2390 |
| + Triggering Model | 0.4371 | 0.5655 | 0.2628 |
| + Class-based & Triggering | 0.4415 | 0.5729 | 0.2645 |

trieval performance significantly. Interpolating class-based model with the baseline also improved the system performance, but the improvement was not as pronounced with the triggering model. We also interpolated all the three models, the baseline, the class-based, and the trained triggering models together and achieved another step improvement on the system performance.

Figure 3 shows the precision-recall curve of the baseline model and all of the interpolation models. This curve indicates the superiority of our proposed model to the baseline model such that the proposed model significantly outperform the baseline at the level of $p-value < 0.01$ according to the two-tailed paired *t*-test.

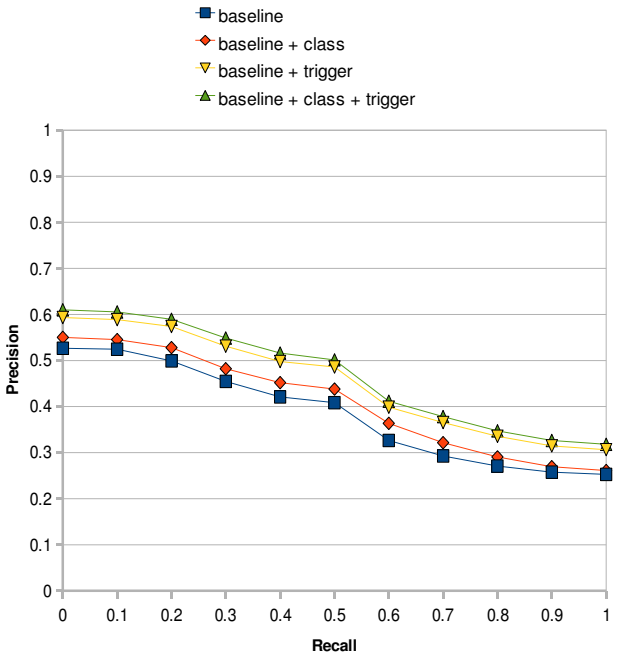


Figure 3: Comparing the precision-recall curve of the baseline model with the interpolated proposed models

## 6. Conclusion

In this paper, we proposed a way of exploiting the logs derived from community-generated question answering forums in automatic question answering systems to offer more accurate answers to users' questions. To this end, we used the Yahoo! Answer Corpus derived from *http://answers.yahoo.com/* as a community question answering log to retrieve more relevant sentences in an automatic question answering system. The retrieved sentences can be further processed using a variety of information extraction techniques to find the final answers.

Two different language model-based frameworks have been introduced here and trained on the Yahoo! Answer Corpus. Our experiments on TREC question answering track verified that both of the models can improve the sentence retrieval performance, in which interpolating both proposed models with the baseline performs the best compared to each of the individual models.

One possible approach to expand the current model is benefiting from the meta data that Yahoo! provides for this corpus. At the moment, no meta information is used in our model. However, it is probable giving a higher priority to the best answer labeled in the corpus or using the category of the question improves the model.

## Acknowledgments

The authors would like to thank Grzegorz Chrupala for interesting discussions. We, in particular, thank Yahoo! Labs Webscope for their Yahoo! Answers Comprehensive Questions and Answers corpus. Saeedeh Momtazi is funded by the German research foundation DFG through the International Research Training Group (IRTG 715) Language Technology and Cognitive Systems.

## 7. References

A. Berger and J. Lafferty. 1999. Information retrieval as statistical translation. In *Proceedings of ACM SIGIR International Conference*, pages 222–229.

P.F. Brown, J. Cocke, S.D. Pietra, V.J Della Pietra, F. Jelinek, J.D. Lafferty, R.L. Mercer, and P.S. Roossin. 1990. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85.

P.F. Brown, V.J.D. Pietra, P.V. Souza, J.C. Lai, and R.L. Mercer. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479.

C. Burgess, K. Livesay, and K. Lund. 1998. Explorations in context space: Words, sentences, discourse. *Discourse Processes*, 25(2,3):211–257.

G. Cao, J.Y. Nie, and J. Bai. 2005. Integrating word relationships into language models. In *Proceedings of ACM SIGIR International Conference*.

G. Cao, J.Y. Nie, and J. Bai. 2006. Constructing better document and query models with markov chains. In *Proceedings of ACM CIKM International Conference*.

E. Charniak. 1993. *Statistical Language Learning*. The MIT Press, Cambridge MA.

C.Y. Chung and B. Chen. 2002. CVS: a correlation-verification based smoothing technique on information retrieval and term clustering. In *Proceedings of ACM SIGKDD International Conference*.

W.B. Croft and X. Wei. 2005. Context-based topic models for query modification. Technical report, University of Massachusetts, Amherst.

H.T. Dang, J.J. Lin, and D. Kelly. 2006. Overview of the trec 2006 question answering track. In *Proceedings of Text REtreival Conference (TREC)*.

J. Gao, J.Y. Nie, G. Wu, and G. Cao. 2004. Dependence language model for information retrieval. In *Proceedings of ACM SIGIR International Conference*.

D.A. Hiemstra. 1998. Linguistically motivated probabilistic model of information retrieval. In *Proceedings of European Conference on Digital Libraries*, pages 569–584.

F. Jelinek and R. Mercer. 1989. Interpolated estimation of markov source parameters from sparse data. In *Proceedings of an International Workshop on Pattern Recognition in Practice*.

Fred Jelinek. 1998. *Statistical Methods for Speech Recognition*. The MIT Press, Cambridge MA.

K.S. Jones. 1971. *Automatic Keyword Classification for Information Retrieval*. Butterworths, London.

M. Kaisser and J.B. Lowe. 2008. Creating a research collection of question answer sentence pairs with Amazon's mechanical turk. In *Proceedings of the LREC International Conference*.

J. Lafferty and C. Zhai. 2001. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of ACM SIGIR International Conference*, pages 111–119.

X. Liu and W.B. Croft. 2004. Cluster-based retrieval using language models. In *Proceedings of ACM SIGIR International Conference*, pages 186–193.

S. Liu, F. Liu, C. Yu, and W. Meng. 2004. An effective approach to document retrieval via utilizing WordNet and recognizing phrases. In *Proceedings of ACM SIGIR International Conference*, pages 266–272.

D.J.C. Mackay and B. Peto. 1995. A hierarchical Dirichlet language model. *Natural Language Engineering*, 1(3):1–19.

R. Mandala, T. Tokunaga, and H. Tanaka. 1998. Ad hoc retrieval experiments using wordnet and automatically constructed theasuri. In *Proceedings of Text REtreival Conference (TREC)*, pages 475–481.

C.D. Manning, P. Raghavan, and H. Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.

A. Merkel and D. Klakow. 2007. Comparing improved language models for sentence retrieval in question answering. In *Proceedings of Computational Linguistics in the Netherlands Conference*, pages 475–481.

D. Miller, T. Leek, and R.M. Schwartz. 1999. A hidden markov model information retrieval system. In *Proceedings of ACM SIGIR International Conference*, pages 214–222.

S. Momtazi and D. Klakow. 2009. A word clustering approach for language model-based sentence retrieval in question answering systems. In *Proceedings of ACM CIKM International Conference*, pages 1911–1914.

S. Momtazi, S. P. Khudanpur, and D. Klakow. 2010. A comparative study of word co-occurrence for term clustering in language model-based sentence retrieval. In *Proceedings of NAACL International Conference*.

V.G. Murdock. 2006. *Aspects of sentence retrieval*. Ph.D. thesis, University of Massachusetts Amherst.

R. Nallapati and J. Allan. 2002. Capturing term dependencies using a sentence tree based language model. In *Proceedings of ACM CIKM International Conference*, pages 383–390.

J. Ponte and W.B. Croft. 1998. A language modeling approach to information retrieval. In *Proceedings of ACM SIGIR International Conference*, pages 275–281.

Y. Qui and H. Frei. 1993. Concept based query expansion. In *Proceedings of ACM SIGIR International Conference*, pages 160–169.

C.J. Van Rijsbergen. 1979. *Information Retrieval*. Butterworths, London, 2 edition.

S. E. Robertson, C.J. van Rijsbergen, and M.F. Porter. 1981. Probabilistic models of indexing and searching. In *Proceedings of ACM SIGIR International Conference*, pages 35–56.

H. Schütze and J.O. Pedersen. 1997. A cooccurrence-based thesaurus and two applications to information retrieval. *Information Processing and Management*, 33(3):307–318.

D. Shen. 2008. *Exploring Rich Evidence for Maximum Entropy-based Question Answering*. Ph.D. thesis, Saarland University.

F. Song and W.B. Croft. 1999. A general language model for information retrieval. In *Proceedings of ACM SIGIR International Conference*, pages 279–280.

A. Stolcke. 2002. SRILM - an extensible language modeling toolkit. In *Proceedings of the Spoken Language Processing Conference*.

M. Surdeanu, M. Ciaramita, and H. Zaragoza. 2008. Learning to rank answers on large online QA collections. In *Proceedings of ACL International Conference*.

T. Tao, X. Wang, Q. Mei, and C. Zhai. 2006. Language model information retrieval with document expansion. In *Proceedings of NAACL-HLT International Conference*, pages 407–414.

E. Voorhees. 1994. Query expansion using lexical-semantic relations. In *Proceedings of ACM SIGIR International Conference*, pages 61–69.

Yahoo! Labs Webscope. 2009. Yahoo! answers comprehensive questions and answers [version 1.0]. Website. `http://webscope.sandbox.yahoo.com`.

X. Wei and W.B. Croft. 2006. LDA-based document models for ad-hoc retrieval. In *Proceedings of ACM SIGIR International Conference*, pages 178–185.

X. Wei and W.B. Croft. 2007. Modeling term associations for ad-hoc retrieval performance within language modeling framework. In *Proceedings of European Conference of Information Retrieval*, pages 52–63.