A Word Clustering Approach for Language Model-based Sentence Retrieval in Question Answering Systems

Saeedeh Momtazi
Spoken Language Systems
University of Saarland, Saarbrücken, Germany
saeedeh.momtazi@lsv.uni-saarland.de

Dietrich Klakow
Spoken Language Systems
University of Saarland, Saarbrücken, Germany
dietrich.klakow@lsv.uni-saarland.de

ABSTRACT

In this paper we propose a term clustering approach to improve the performance of sentence retrieval in Question Answering (QA) systems. As the search in question answering is conducted over smaller segments of data than in a document retrieval task, the problems of data sparsity and exact matching become more critical. In this paper we propose Language Modeling (LM) techniques to overcome such problems and improve the sentence retrieval performance. Our proposed methods include building class-based models by term clustering, and then employing higher order *n-grams* with the new class-based model. We report our experiments on the TREC 2007 questions from QA track. The results show that the methods investigated here enhanced the mean average precision of sentence retrieval from 23.62% to 29.91%.

Categories and Subject Descriptors: H.3.3 Information Storage and Retrieval:Information Search and Retrieval [clustering, retrieval models]

General Terms: Theory, Algorithms

Keywords: information retrieval, sentence retrieval, question answering, language modeling, term clustering

1. INTRODUCTION

In the recent past, open domain QA has become one of the most actively investigated topics in natural language processing. Its popularity stems from the fact that a user receives an exact answer to his questions rather than being overwhelmed with a large number of retrieved documents, which he must then sort through to find the desired answer.

In complete a QA 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 the QA 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 which is our research focus. Retrieved sentences are then further processed using a variety of techniques to extract the final answers. Shen [12] showed that improvement in sentence retrieval performance has a significant positive effect on the accuracy of the QA system.

For retrieving relevant sentences, we adopt here the LM-based information retrieval proposed by Ponte and Croft [10]. This technique has also been used for sentence retrieval and found to outperform other methods like *tfidf* and *Okapi* [9].

In the majority of the information retrieval literature, a simple word-based unigram is used. The major fault of this model is that all terms are treated independently and no relationships between words are considered. When estimating word-based unigrams, a search is performed for only the exact literal words present in the query. In consequence, such algorithms will fail to retrieve other relevant information. For example, having the question *"Who invented the car?"* as an input for a QA system, the sentence retrieval component might retrieve *"Between 1832 and 1839, Robert Anderson of Scotland invented the first crude electric car carriage."*. But using the exact match model, the retrieval algorithm misses sentences like *"Nicolas-Joseph Cugnot built the first self propelled mechanical vehicle."* or *"An automobile powered by his own engine was built by Karl Benz in 1885 and granted a patent."*, although these sentences contain the words *"built"*, *"vehicle"*, and *"automobile"* which are very likely to be relevant to the question.

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. Improvements in system performance from such schemes have proven chalenging, however, 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 scheme [6]. In this paper, we introduce a class-based model by applying term clustering to estimate term relationships. Then, use a linear interpolation to integrate both word- and class-based models.

The paper is organized as follows. In Section 2, we describe our class-based model. Section 3 introduces the idea of using higher order n-grams in the new class-based model. In Section 4, experimental results are presented. Finally, Section 5 summarizes the paper and suggests future work.

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

2. TERM CLUSTERING MODEL

In this research, we use a class-based LM by applying term clustering to capture term relationships. Term clustering has been widely used in different natural language processing applications, including speech recognition [11], machine translation [15], query expansion [1], text categorization [4], automatic thesaurus generation [7], and word sense disambiguation [8]. However, to the best of our knowledge, it has not been applied to LM-based information retrieval.

One of the advantages of term clustering is its flexibility to capture different features. Semantic, syntactic, and statistical properties of terms can be utilized in various term clustering algorithms. Another advantage of term clustering is its inherent reduction in data sparsity. As there are vastly fewer clusters than words, the variation in possible patterns is reduced and the probability of seeing a given pattern is thereby enhanced. Hence, in addition to relaxing the exact match requirement the proposed approach solves a portion of the data sparsity problem.

2.1 New Class-based Model

In LM-based sentence retrieval, the probability $P(Q|S)$ of generating query Q conditioned on the observation of sentence S is first calculated, and thereafter sentences are ranked in descending order of this probability. For word-based unigrams, the probability of the query Q given the sentence S is estimated based on query terms.

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

where M is the number of query terms, q_i denotes the i^{th} query term in Q, and S is the sentence model.

For class-based unigrams, $P(Q|S)$ is computed based on the cluster to which each query term belongs,

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

where C_{q_i} is the cluster that contains 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. To calculate $P(C_{q_i}|S)$, the sentence model is created based on clusters instead of terms. As a result, each cluster is considered as a single entity and these new entities are used to create the model.

To be more specific, for each sentence in the word-based model, the sentence terms are extracted. Then the sentence unigram is created from the terms. While in the class-based model, using a term clustering technique, a map between terms and clusters is created. In this case, instead of extracting the terms, the clusters which the terms belong to are extracted. Then the sentence unigram is created from the clusters. Finally, $P(Q|S)$ is computed based on the new model of S, and the sentences are ranked according to this probability.

2.2 Term Clustering Algorithm

In order to cluster lexical items, we used the word clustering algorithm proposed by Brown [2] which is one of the most well-known and effective clustering algorithms in LM.

The Brown algorithm uses mutual information between cluster pairs in a bottom-up approach to maximize Average Mutual Information (AMI) between adjacent clusters.

As shown in Algorithm 1, the clusters are initialized with a single term. Then, in each iteration, all pairs of clusters are temporarily combined and the AMI between adjacent clusters is computed. Thereafter, the best cluster pair, which offers a minimum decrement in AMI, is permanently combined together. This continues for $V - K$ iterations, where V is the number of terms and K is the predefined number of clusters. To increase the AMI, a final step is performed, whereby each term is moved to that cluster for which the resulting partition has the greatest AMI. The algorithm terminates when AMI ceases to increase.

Algorithm 1 The Brown Word Clustering Algorithm

```
Initial Mapping: Put a single word in each cluster
Compute the initial AMI of the collection
repeat
  for each pairs of clusters do
    Merge the pair of clusters temporarily
    Compute the AMI of the collection
  end for
  Select a pair of clusters with min. decrement of AMI
  Compute AMI of the new collection
until reach the predefined number of clusters
repeat
  Move each term to the cluster for which the resulting
  partition has the greatest AMI
until no more increment in AMI
```

Upon examining the results of the clustering, it is evident that there is a clear relationship between the terms belonging to the same cluster. The following lines show some examples of clusters created by the Brown algorithm:
— *significantly, highly, strongly, fairly*
— *shipping, carrying*
— *Jazz, symphony, theater*
which indicate having only a single term of a cluster is sufficient to search for other terms of the same cluster. For example, seeing the term *"highly"* in a query, the class-based model is able to retrieve sentences that do not have this term but are nonetheless relevant to the query in that the cluster also contains other terms like *"strongly"*. The word-based model, on the other hand, would not rank such sentences highly.

2.3 Interpolation Model

The class-based model can reasonably be expected to increase system recall, in that it is able to find more relevant sentences. This method may decrease the system precision, however, by retrieving more irrelevant sentences. To avoid this problem, we use the linear interpolation of our new class-based model and the word-based model, thereby benefitting from the advantages of both models and avoiding their disadvantages.

To use the interpolation model, the probability q_i given both word- and class-based models is computed from (1) and (2) and interpolated by a weighting parameter λ.

$$\begin{aligned} P(Q|S) = \prod_{i=1\ldots M} [\lambda\ P(q_i|C_{q_i}S)P(C_{q_i}|S) \\ + (1-\lambda)\ P(q_i|S)] \end{aligned} \tag{3}$$

As mentioned before, one of the difficulties of relaxing the exact matching assumption is the problem of integrating both exact matching of single words and term relationship in a weighting scheme. By using the interpolation model we can test our model with different weights of λ.

3. N-GRAM MODELS

In practice, the statistical LM is often approximated by the recent sequence of words, namely n-grams. The simplest of n-gram models, namely the unigram, is most often used in information retrieval applications. In this model, however, the word dependencies are not considered. To overcome this problem, it is necessary to consider longer word contexts, such as those used in the bigram and trigram. As these models include local context, they are able to condition the probability of the next word on those previous.

Much of the research into traditional information retrieval methods failed to result in performance improvements by applying either bigram or trigram models [3]. Song and Croft [14] used bigram for the first time in LM-based information retrieval and obtained a better result in document retrieval.

It is very difficult to apply these models in the context of sentence retrieval, however, as data sparsity at the sentence level is much more pronounced than that at the document level. At the sentence level, the probability of seeing large patterns is very low and hence it is very difficult to use higher order n-grams for sentence retrieval. To overcome this problem, we applied the class-based bigram and trigram instead of the word-based bigram in order to test the hypothesis that by using the class-based model, the data sparsity problem could be sufficiently ameliorated so that higher order n-grams would become effective for this task.

Having class-based sentence retrieval, we use class bigram and class trigram models formulated as follows:

$$\begin{aligned} P^{bi}(Q|S) = {} & P(q_1|C_{q_1},S)P(C_{q_1}|S) \\ & \prod_{i=2\ldots M} P(q_i|C_{q_i},S)P(C_{q_i}|C_{q_{i-1}},S) \end{aligned} \tag{4}$$

$$\begin{aligned} P^{tri}(Q|S) = {} & P(q_1|C_{q_1},S)P(C_{q_1}|S) \\ & P(q_2|C_{q_2},S)P(C_{q_2}|C_{q_1},S) \\ & \prod_{i=3\ldots M} P(q_i|C_{q_i},S)P(C_{q_i}|C_{q_{i-2}}C_{q_{i-1}},S) \end{aligned} \tag{5}$$

Although the class-based model helps to have less sparse data, this problem is not fixed completely. Hence, for using the trigram model, the bigram model is used as background model. The bigram probabilities have also been combined with the unigram as a background model.

4. EXPERIMENTS

To evaluate our models, we used the set of TREC 2007[1] questions from the QA track. The TREC 2007 QA task contains 70 question series, which focused on a target, consisted of several *factoid* questions, one to two *list* questions, and exactly one *other* question [5]. For our experiments, the factoid questions have been used, while the sentence retrieval model described in this paper are suitable for different types of questions. The total number of factoid question for all 70 series used in our experiments is 218 questions.

The relevance judgments available in NIST data are annotated at the document level. To evaluate our sentence retrieval, we need to perform our own annotation to find relevant sentences as "ground truth" in evaluating our model. A relevant sentence in the QA task, is a sentence which is related to the question and contains the answer. To extract the set of relevant sentences, we used an automatic step followed by a manual step. In the automatic step, for each answer of each question, the relevant document to that particular answer released by NIST was processed and all sentences that contain the answer string were extracted. In the second step, the extracted sentences were checked manually and the sentences which are not related to the context of the question were filtered out.

To do our sentence retrieval experiments, a set of 1,960 sentences were used as input to our system. For each question, the sentence set contains the relevant sentences to the question which is almost 10 sentences on average with the rest of the sentences in the set being irrelevant. This input set was selected in the same way as described by Shen et. al [13]. Since the answers of TREC 2007 questions are extracted from the *BLOG06* and *AQUAINT* corpora, we also used the same corpora for document retrieval and consequently the sentence retrieval task. The lexicon contained nearly 10,000 terms, which included all words present in the questions and the sentences to be processed.

To create the proposed model, in the first step, word-based unigram and bigram models were created. Then, by clustering lexical items, class-based unigram, bigram, and trigram models were created, wherein lower order n-grams were used as back-off models. Finally, the new class-based model was interpolated with the word-based model. For each question, the system ranked the sentences according to the probability of generating the query from the sentences models.

For all models three different smoothing methods proposed by Zhai and Lafferty [16] for the document level were applied. The smoothing methods include Jelinek-Mercer linear interpolation, Bayesian smoothing with Dirichlet priors, and absolute discounting.

Finally, we need to specify the number of classes to cluster the words. Tests were conducted with several numbers of clusters. Figure 1 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 10,000 lexical items into 3,000 clusters. We have the same result in bigram and trigram. Hence, this value was used in all further experiments.

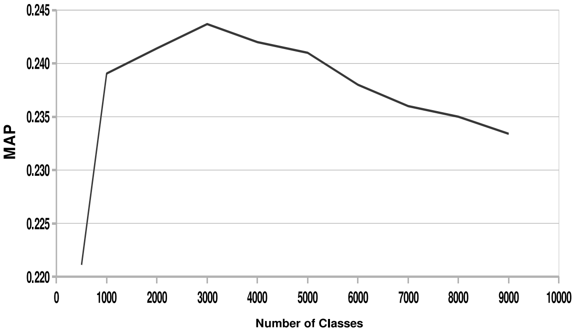

Figure 1: MAP over different numbers of classes

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

Table 1: Comparison between word-based models and class-based models
(* and ** indicate the difference is statistically significant at the level of *p-value*< 0.05 and *p-value*< 0.01 based on two tailed t-tests.)

Smoothing	Word-based			Class-based					
	Unigram	Bigram		Unigram		Bigram		Trigram	
	MAP	MAP	%change	MAP	%change	MAP	%change	MAP	%change
Absolute Discounting	0.2332	0.2305	-1.18	0.2437	+4.48*	0.2664	+14.20**	0.2610	+11.90**
Dirichlet Prior	0.2362	0.2318	-1.85	0.2414	+2.22	0.2662	+12.72**	0.2603	+10.21**
Jelinek-Mercer	0.2340	0.2302	-1.63	0.2426	+3.68*	0.2656	+13.50**	0.2598	+11.02**

Table 1 reports the upper bound results of our experiments while using different smoothing parameters, in which we use the word-based unigram as our baseline. The columns titled "% change" illustrate the relative change of each model compared to word-based unigram model. As shown, we achieved a significant improvement on class-based unigram compared to the word-based unigram model. This improvement indicates that the term clustering approach proposed here could successfully capture relationships between terms and retrieve more relevant sentences.

As discussed in Section 4, using word-based bigram model did not improve the mean average precision due to the data sparsity problem at sentence level. The class-based bigram, however, does improve the sentence retrieval performance which shows that our proposed class-based model solved a part of data sparsity. The performance of the class-based trigram is still inferior to that of its bigram counterpart, however, due to data sparsity that our procedure has not entirely eliminated.

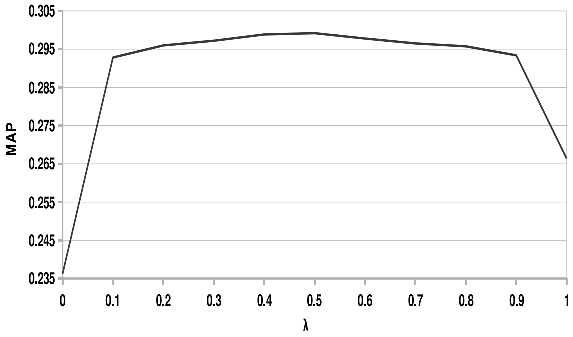

Figure 2: MAP over different interpolation weights

As mentioned previously, to achieve the best performance, we interpolated word- and class-based models. Towards this end, the best class-based model which is bigram and the best word-based model which is unigram were selected. Figure 2 shows the mean average precision for various values of λ. We see the interpolation model with any mid-range weight outperforms both word- and class-based models, while at $\lambda = 0.5$, mean average precision increases to 0.2991.

5. CONCLUSIONS AND FUTURE WORK

In this paper we introduced a class-based LM approach using term clustering for sentence retrieval in QA systems to solve the data sparsity, the exact matching, and the term independence problems. We found out that the class-based model using term clustering is an effective approach to this end. In addition, we also evaluated class-based bigram and trigram. Moreover, linear interpolation of a class-based bigram and a word-based unigram was investigated. Our results indicated a significant improvement of sentence retrieval mean average precision from 23.62% to 29.91%.

In this research we only applied the Brown clustering method, as this is the best-known algorithm for term clustering in LM. In future work, we plan to apply other term clustering algorithms and compare their effectiveness on sentence retrieval performance.

6. REFERENCES

[1] M. Aono and H. Doi. A method for query expansion using a hierarchy of clusters. In *AIRS*, pages 479–484, 2005.

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

[3] J. Callan, W. Croft, and J. Broglio. Trec and tipster experiments with inquery. *Information Processing and Management*, 31(3):327–343, 1995.

[4] W. Chen, X. Chang, H. Wang, J. Zhu, and T. Yao. Automatic word clustering for text categorization using global information. In *AIRS Internation Conference Proceedings*, volume 3411 of *Lecture Notes in Computer Science*, pages 1–11. Springer, 2004.

[5] H. Dang, D. Kelly, and J. Lin. Overview of the trec 2007 question answering track. In *TREC*, 2007.

[6] J. Gao, J. Nie, G. Wu, and G. Cao. Dependence language model for information retrieval. In *Proc. of ACM SIGIR*, 2004.

[7] V. Hodge and J. Austin. Hierarchical word clustering – automatic thesaurus generation. *Neurocomputing*, 48:819–846, 2002.

[8] H. Li. Word clustering and disambiguation based on co-occurrence data. *Nat. Lang. Eng.*, 8(1):25–42, 2002.

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

[10] J. Ponte and W. Croft. A language modeling approach to information retrieval. In *Proc. of ACM SIGIR*, pages 275–281, 1998.

[11] C. Samuelsson and W. Reichl. A class-based language model for large-vocabulary speech recognition extracted from part-of-speech statistics. In *IEEE ICASSP International Conference Proceedings*. IEEE Computer Society, 1999.

[12] D. Shen. *Exploring Rich Evidence for Maximum Entropy-based Question Answering*. PhD thesis, Saarland University, 2008.

[13] D. Shen, G. Kruijff, and D. Klakow. Exploring syntactic relation patterns for question answering. In *Proc. of the IJCNLP*, 2005.

[14] F. Song and W. Croft. A general language model for information retrieval. In *Proc. of ACM SIGIR*, pages 279–280, 1999.

[15] J. Uszkoreit and T. Brants. Distributed word clustering for large scale class-based language modeling in machine translation. In *ACL International Conference Proceedings*, Columbus, OH, USA, 2008.

[16] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proc. of ACM SIGIR*, 2001.