TREC2007 QUESTION ANSWERING EXPERIMENTS AT TOKYO INSTITUTE OF TECHNOLOGY

Edward W. D. Whittaker, Matthias H. Heie, Josef R. Novak and Sadaoki Furui

Tokyo Institute of Technology, Department of Computer Science
2-12-1 Ookayama, Meguro-ku, Tokyo, 152-8552 Japan,
{edw,heie,novakj,furui}@furui.cs.titech.ac.jp

ABSTRACT

In this paper we describe Tokyo Institute of Technology's attempt at the TREC2007 question answering (QA) track. Keeping the same theoretical QA model as for the TREC2006 task, this year we once again focused on the factoid QA task, while investigating a new method for sentence retrieval. We deviated from our earlier approach of using web data, and instead relied solely on the supplied news wire and blog data. Our factoid and list score fell significantly from last year, while we achieved a higher *other* question score compared to TREC2006, using sentence retrieval rather than last year's summarization method.

1. INTRODUCTION

In this paper we describe the application of our data-driven and non-linguistic framework for the QA task of TREC2007. Three runs were submitted for evaluation (`asked07a,b,c`).

As in TREC2005 [1] and TREC2006 [2], our focus was on the factoid question task. We experimented with retrieving sentences for answer extraction using an approach that was successfully applied in Question Answering on speech transcripts (QAst) [3], a pilot task at the Cross Language Evaluation Forum (CLEF) evaluations 2007. In this approach, a language model (LM) is generated for each sentence and these models are combined with document LMs to take advantage of contextual information. We ran experiments using both this sentence retrieval technique and document retrieval, and further combined sentence retrieval and document retrieval using a method of system combination that has been found to be robust and effective at boosting overall system performance.

For the list task, an extension to the factoid QA system was used, similar to what we did last year. For the *other* question task we filtered sentences retrieved by our sentence retrieval module, using two different methods.

As data sources we used the AQUAINT-2 and Blog06 collections. This contrasts with our approach in earlier years, where we relied solely on web data and then projected answers into the AQUAINT-1 collection.

2. SENTENCE RETRIEVAL

This section explains the LM-based sentence retrieval method presented in [4].

Language modeling for IR has gained in popularity over the last decade since the approach was proposed [5]. Under this approach a LM is estimated for each document. The documents are then ranked according to the conditional probability $P(Q|D)$, the probability of generating the query Q given the document D.

A language model based approach to sentence retrieval for QA is presented in [6]. Due to lack of data to train the sentence specific LM, it is assumed that all words are independent, hence unigrams are used:

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

where q_i is the ith query term in the query $Q = (q_1 ... q_{|Q|})$ composed of $|Q|$ query terms.

Smoothing methods are normally employed with LMs to avoid the problem of zero probabilities when one of the query terms does not occur in the document. This is typically achieved by redistributing probability mass from the document model to a background collection model $P(Q|B)$. We use absolute discounting, where the probability of a query term q given a sentence S is calculated as:

$$\begin{aligned} P_1(q|S) &= \frac{\max\{tf(q,S) - \delta, 0\}}{l(S)} \\ &\quad + \frac{\delta \cdot h(S,\delta)}{l(S)} \cdot P(q|B), \end{aligned} \tag{2}$$

where $tf(q,S)$ is the term frequency of q in S, $l(S)$ is the length (number of words) of S, δ is the discount parameter, $h(S,\delta)$ is the count of how many unique words in S have a term frequency higher than δ, and $P(q|B)$ is the unigram probability of the query term q according to the background collection model. Note that if $\delta < 1$ then $h(S,\delta)$ is equal to the number of unique words in S.

A problem with the model presented in [6] is that words relevant to the sentence might not occur in the sentence itself, but in the surrounding text. For example, for the question *Where was George Bush born?*, the sentence *He was born in Connecticut* in an article about George Bush should ideally be assigned a high probability, despite the sentence missing important query terms. To account for this, we train document LMs, $P_1(q|D)$, in the same manner as for $P_1(q|S)$ in Eq. (2), and perform a linear interpolation between $P_1(q|S)$ and $P_1(q|D)$:

$$P_2(q|S) = (1-\alpha) \cdot P_1(q|S) + \alpha \cdot P_1(q|D), \qquad (3)$$

where $0 \leq \alpha \leq 1$ is an interpolation parameter.

3. ANSWER EXTRACTION

For answer extraction we used the same framework as in TREC2006 [2], described in detail in [7]. The explanation below is largely re-produced from [8].

We model the most straightforward and obvious dependence of the probability of an answer A depending on a question Q:

$$P(A \mid Q) = P(A \mid W, X), \qquad (4)$$

where A and Q are considered to be strings of l_A words $A = a_1, \ldots, a_{l_A}$ and l_Q words $Q = q_1, \ldots, q_{l_Q}$, respectively. Here $W = w_1, \ldots, w_{l_W}$ represents a set of features describing the "question-type" part of Q such as *when*, *why*, *how*, etc. while $X = x_1, \ldots, x_{l_X}$ represents a set of features that describe the "information-bearing" part of Q i.e. what the question is actually about and what it refers to. For example, in the questions, *Where was Tom Cruise married?* and *When was Tom Cruise married?*, the information-bearing component is identical in both cases whereas the question-type component is different.

Finding the best answer $\hat{A}$ involves a search over all available A for the one which maximizes the probability of the above model i.e.,

$$\hat{A} = \arg\max_A P(A \mid W, X). \qquad (5)$$

Given the correct probability distribution, this is guaranteed to give us the optimal answer in a maximum likelihood sense. We don't know this distribution and it is still difficult to model but, using Bayes' rule and making various simplifying, modeling and conditional independence assumptions (as described in detail in [7]) Equation (5) can be rearranged to give

$$\arg\max_A \underbrace{P(A \mid X)}_{\substack{answer \\ retrieval \\ model}} \cdot \underbrace{P(W \mid A)}_{\substack{answer \\ filter \\ model}}. \qquad (6)$$

The $P(A \mid X)$ model is essentially a statistical language model that models the probability of an answer sequence A given a set of information-bearing features X. We call this model the *answer retrieval model* and do not examine it further in this paper (see [7] for more details).

The $P(W \mid A)$ model matches a potential answer A with features in the question-type set W. For example, it relates place names with *where*-type questions. In general, there are many valid and equiprobable A for a given W so this component can only re-rank candidate answers obtained by the retrieval model. We call this component the *answer filter model* and it is structured as follows.

The question-type feature set $W = w_1, \ldots, w_{l_W}$ is constructed by extracting n-tuples ($n = 1, 2, \ldots$) such as *Where*, *In what* and *When were* from the input question Q. A set of $|V_{\mathcal{W}}| = 2522$ single-word features is extracted based on frequency of occurrence in our collection of example questions.

Modeling the complex relationship between W and A directly is non-trivial. We therefore introduce an intermediate variable representing classes of example questions-and-answers (q-and-a) c_e for $e = 1 \ldots |C_E|$ drawn from the set C_E. In order to construct these classes, given a set E of example q-and-a, we then define a mapping function $f : E \mapsto C_E$ which maps each example q-and-a t_j for $j = 1 \ldots |E|$ into a particular class $f(t_j) = e$. Thus each class c_e may be defined as the union of all component q-and-a features from each t_j satisfying $f(t_j) = e$. Finally, to facilitate modeling we say that W is conditionally independent of c_e given A so that

$$P(W \mid A) \quad = \quad \sum_{e=1}^{|C_E|} P(W \mid c_W^e) \cdot P(c_A^e \mid A), \qquad (7)$$

where c_W^e and c_A^e refer respectively to the subsets of question-type features and example answers for the class c_e.

Assuming conditional independence of the answer words in class c_e given A, and making the modeling assumption that the jth answer word a_j^e in the example class c_e is dependent only on the jth answer word in A we obtain:

$$P(W \mid A) = \sum_{e=1}^{|C_E|} P(W \mid c_e) \cdot \prod_{j=1}^{l_{A^e}} P(a_j^e \mid a_j). \qquad (8)$$

Since our set of example q-and-a cannot be expected to cover all the possible answers to questions that may be asked we perform a similar operation to that above to give us the following:

$$P(W \mid A) = \sum_{e=1}^{|C_E|} P(W \mid c_e) \prod_{j=1}^{l_{A^e}} \sum_{k=1}^{|C_A|} P(a_j^e \mid c_k) P(c_k \mid a_j), \qquad (9)$$

where c_k is a concrete class in the set of $|C_A|$ answer classes C_A. The independence assumption leads to underestimating

System	Data source	Ret. model	Filter model (factoid&list)	Lev. dist. (other)	Submitted run
asked07a	AQUAINT-2+Blog06	Sent.	ONE	no	yes
asked07b	AQUAINT-2+Blog06	Doc.	ONE	yes	yes
asked07c	AQUAINT-2+Blog06	Sent.+Doc.	ONE+TWO	no	yes
asked07i	AQUAINT-2+Blog06	Doc.	TWO	-	no

Table 1. Descriptions of systems developed for TREC2007 including the 3 submitted runs asked07a,b,c and the intermediate run asked07i that was not submitted for evaluation.

the probabilities of multi-word answers so we take the geometric mean of the length of the answer (not shown in Equation (9)) and normalize $P(W \mid A)$ accordingly.

The model given by Equation (9) is referred to as model ONE, while the model given by Equation (7) are referred to as model TWO. Model TWO is described in detail in [9].

4. SYSTEM COMBINATION

For the run asked07c the answers for the factoid and list tasks were generated through combination of multiple systems. Answer combination was performed by simply summing the inverse rank of an answer a from each component system s to generate a new score for the answer as follows:

$$score(a) = \sum_{s} \frac{1}{r_s(a)}. \quad (10)$$

For the *other* question task, no system combination was performed.

5. EXPERIMENTAL WORK

Three different runs (asked07a,b,c) were submitted for evaluation with characteristics given in Table 1.

Run asked07a used sentence retrieval and model ONE for answer extraction. Run asked07b used document retrieval and model ONE for answer extraction. We executed an intermediate run asked07i that used document retrieval and model TWO for answer extraction. Run asked07c combined asked07i with asked07a and asked07b using system combination.

5.1. Data pre-processing, indexing and retrieval

This year we chose to use only the AQUAINT-2 and Blog06 collections as data sources, where all reference answers (for questions with an answer) are drawn from. This contrasts with our approach in TREC2005 and TREC2006, where all answers were extracted from web data.

Raw text was extracted from the XML format of the AQUAINT-2 and Blog06 collections. This text was converted to upper-case and cleaned using a series of regular expressions. Moreover, the text was sentence segmented, using a rule-based algorithm, to facilitate efficient sentence retrieval. The preprocessed documents were then indexed using the Xapian search engine library[1]. A set of 41 stopwords was used during indexing and retrieval. 1000 documents were retrieved for each question, and used directly by the answer extraction module in run asked07b and asked07i. In run asked07a the sentences in these documents were re-ranked using the sentence retrieval model described in Section 2.

Questions were cleaned in the same way as for documents. If the target for a question did not appear character-for-character in the question string, it was simply appended to the end of the question string. Common question-type words, such as *when*, *why*, *how*, etc. for factoid questions, and *list*, *name*, etc. for list questions, were removed. For *other* questions, the query terms used were the words describing the target in the question file. Each question was treated independently of all other questions.

5.2. Factoid question task

For system development this year we optimized performance on earlier TREC evaluation questions. As in TREC2006, the filter model was trained by using 288812 example q-and-a from the Knowledge Master (KM) data[2] plus q-and-a from the TREC-8 to TREC-13 evaluations.

The most frequent 224000 words from the AQUAINT-1 corpus, augmented with a large set of numbers, in total 819316 tokens were used to obtain C_A for $|C_A| = 504$ clusters as described in [7].

5.3. List question task

This year once again no development was performed on list questions. Our factoid QA system always outputs a list of candidate answers ranked by their probabilities. The issue for the list task is therefore to determine how many of the top answers to output so as to maximize the F-score. We chose simply to output the top 10 answers of the answer extraction module.

[1] http://www.xapian.org/
[2] http://www.greatauk.com/

System	Factoid task				List task	Other task	Avg. per-series score
	Globally right	Locally correct	Unsupp.	Inexact			
asked07a	44 (12.2%)	3 (0.8%)	10 (2.8%)	14 (3.9%)	0.027	0.118	0.089
asked07b	44 (12.2%)	5 (1.4%)	9 (2.5%)	17 (4.7%)	0.028	0.115	0.089
asked07c	43 (11.9%)	4 (1.1%)	9 (2.5%)	20 (5.6%)	0.035	0.110	0.089

Table 2. Performance on the 3 tasks of the 3 submitted runs.

5.4. Other question task

This year we used our sentence retrieval module to answer *other* questions. This differs from last year's evaluation, where we treated the answering of *other* questions as a summarization task and employed a variation on a method used for speech summarization for this purpose [10].

For this task we did two experiments. In the first experiment we retrieved sentences using our sentence retrieval model. The retrieved sentences, from which nuggets were to be extracted, were first cleaned to remove words that are unlikely to be required in a nugget but which occur frequently in the data. Duplicate sentences were also removed. After that we simply submitted the 10 highest ranking sentences.

Our second experiment was similar, but here we performed a simple comparison using the minimum Levenshtein difference between each new sentence and those that had already been selected. Sentences that were found to be too similar to already chosen sentences, were discarded.

We used the results of the first experiment for runs `asked07a` and `asked07c`, and the results of the second experiment for run `asked07b`.

6. RESULTS AND DISCUSSION

The results for all 3 submitted runs on all 3 tasks are shown in Table 2.

This year, as in previous evaluations, our focus was on the factoid task. We were especially interested in comparing our new sentence retrieval approach to document retrieval. Our hope was that supplying the answer extraction module with fewer sentences, ranked high by our sentence retrieval module, would perform better than supplying it with a larger amount of whole documents, consisting of more noisy data, at the risk of lower recall and redundancy. Moreover, searching for answers in a few sentences significantly reduces the execution speed of the system.

The results show that retrieving 100 sentences for answer extraction (run `asked07a`), performs nearly the same as retrieving 1000 documents (run `asked07b`). The amount of questions with a globally right answer is 12.2% for both these runs. When the other metrics are also included (locally correct, unsupported and inexact), the performance is slightly worse for sentence retrieval than for document retrieval: 19.7% vs. 20.6%. Manual inspection shows that for approximately 87% of the questions, a correct answer is contained in the 1000 documents retrieved for each question. For the 100 sentences retrieved for each question, the corresponding number is 67%.

For the list task, the performance is approximately the same for both these runs.

Since our method of system combination had been found to be robust and effective at boosting overall system performance in previous evaluations, we hoped this would also be the case when combining sentence retrieval and document retrieval, as we did in run `asked07c`. Unfortunately the percentage of globally right answers goes down slightly to 11.9%. However, by including all the metrics we achieve a score of 21.1%, a higher score than in our other submitted runs this year. The score for the list task is also higher with this combination of models than in the two other runs.

For the *other* questions, it should be noted that the same answers were submitted for run `asked07a` and run `asked07c`, thus the difference in score is due to human judgment. Since run `asked07b` achieves a score between `asked07a` and `asked07c`, we can conclude that using Levenshtein difference had no impact on performance.

For the factoid question task and the list question task, our results were considerably lower than last year. In TREC2006 we got 25.1% of the factoid questions right. For the list task, the best score was 0.074. We believe this decline in performance is due to our decision not to use web data this year. This meant there was less data redundancy and therefore fewer examples of correct answers in the retrieved data, which affected the performance of our statistical approach. For *other* questions, however, we achieve a significant improvement from last year, when our best run yielded a score of 0.064. Thus there is reason to believe that our sentence retrieval approach is more suitable to this task than the summarization technique previously employed.

7. CONCLUSION

In this paper we have given an overview of our methods and results for the TREC2007 question answering evaluation. Our primary focus was on the factoid task. This year's scores were significantly lower than in last year's evaluation, where we relied on web data. The sentence retrieval method employed did not perform much different from document retrieval, but speeded up the QA process significantly. However, using our

sentence retrieval approach on the *other* question task performed significantly better than the summarization technique employed last year.

8. ACKNOWLEDGMENTS

This research was supported by the Japanese government 21st century COE programme on large-scale knowledge resources.

9. REFERENCES

[1] Whittaker, E., Chatain, P., Furui, S. and Klakow, D., "TREC2005 Question Answering Experiments at Tokyo Institute of Technology", *Proc. TREC-14*, 2005.

[2] Whittaker, E., Novak, J., Chatain, P. and Furui, S., "TREC2006 Question Answering Experiments at Tokyo Institute of Technology", *Proc. TREC-15*, 2006.

[3] Whittaker, E., Novak, J., Heie, M. and Furui, S., "CLEF2007 Question Answering Experiments at Tokyo Institute of Technology", *Working Notes CLEF*, 2007.

[4] Heie, M., Whittaker, E., Novak, J. and Furui, S., "A Language Modeling Approach to Question Answering on Speech Transcripts", *Proc. ASRU*, 2007, pp. 219-224.

[5] Ponte, J. and Croft, W., "A Language Modeling Approach to Information Retrieval", *Proc. SIGIR*, 1998, pp. 275-281.

[6] Merkel, A. and Klakow, D., "Comparing Improved Language Models for Sentence Retrieval in Question Answering", *Proc. CLIN-17*, 2007.

[7] Whittaker, E., Furui, S. and Klakow, D., "A Statistical Pattern Recognition Approach to Question Answering using Web Data", *Proc. Cyberworlds*, 2005, pp. 421-428.

[8] Novak, J., Whittaker, E., Heie, M., Imai, S. and Furui, S., "NTCIR-6 CLQA Question Answering Experiments at the Tokyo Institute of Technology", *Proc. NTCIR-6*, 2007, pp. 198-201.

[9] Whittaker, E., Novak, J., Heie, M., Imai, S. and Furui, S., "Using Singular Value Decomposition to Compute Answer Similarity in a Language Independent Approach to Question Answering", To appear in *Proc. LKR*, 2008, pp. 267-279.

[10] Kikuchi, T., Furui, S. and Hori, C., "Automatic Speech Summarization Based on Sentence Extraction and Compaction", *Proc. ICASSP*, 2003, vol. 1, pp. 236-239.