TREC-10 Experiments at KAIST: Batch Filtering and Question Answering

Jong-Hoon Oh, Kyung-Soon Lee, Du-Seong Chang, Chung Won Seo, and Key-Sun Choi

Computer Science Division, Department of EECS, Korea Terminology Research Center for Language and Knowledge Engineering (KORTERM)
Korea Advanced Institute of Science & Technology (KAIST)
Kusong-Dong, Yusong-Gu Taejon, 305-701 Republic of Korea
{rovellia,kslee,dschang,cwseo,kschoi}@world.kaist.ac.kr

1.Introduction

In TREC-10, we participated in two tasks: batch filtering task in the filtering task, and question answering task. In question answering task, we participated in three sub-tasks (main task, list task, and context task).

In batching filtering task, we experimented a filtering technique, which unifies the results of support vector machines for subtopics subdivided by incremental clustering. For a topic, we generated subtopics by detecting similar documents in training relevant documents, and unified the results of SVM classifier for subtopics by OR set operation.

In question answering task, we submitted two runs for main task (KAISTQAMAIN1, KAISTQAMAIN2), two runs for list task (KAISTQALIST1, KAISTQALIST2), and one run for context task (KAISTQACTX).

2. Batch Filtering track

2.1 Experimental Procedure

We experimented a filtering technique, which unifies the results of support vector machines (Vapnick, 1995) for subtopics subdivided by incremental clustering. For a topic, we generated subtopics by grouping similar documents in training relevant documents, and unified the results of SVM classifiers for subtopics by OR set operation.

2.1.1 Subdividing a topic into several subtopics by incremental clustering

For each topic, we generated subtopics by incremental clustering. In incremental clustering, the input documents consist of relevant documents among training documents for a topic.

Incremental clustering sequentially processes the input documents and grows clusters incrementally. The first input document itself becomes one cluster. A new document is

assigned to a member of all the clusters if the similarity between the document and the pre-generated cluster is above threshold. (A document could be a member of several clusters.) Otherwise, the document becomes a new cluster.

In our experiment, we set a cluster threshold to 0.1. It is a random selection. The comparative experiment was not conducted according to various cluster thresholds. Each cluster has a centroid vector which represents all the documents included in the cluster. We used cosine coefficient as a measure to calculate similarity between a document vector and a centroid vector. For 84 topics of Reuters test collection, we generated 275 clusters. For a topic, a cluster represents a subtopic. Therefore, we generated 275 subtopics. Training relevant documents of a subtopic are the subset of those of a topic.

2.1.2 Filtering based on SVM for each subtopic

Support vector machines are based on the Structural Risk Minimization principle (Vapnik, 1995) from computational learning theory. We tested RBF (radial basis function) models offered by SVM^{light} system (Joachims, 1998), which is an implementation of Vapnik's Support Vector Machine for the problem of pattern recognition. SV learning is based on sub-relevant documents subdivided by incremental clustering and all non-relevant documents for each topic. For each subtopic, we filtered the test documents by using SVM classifier.

2.1.3 Unifying the results of subtopics

We unified the results generated from SVM classifiers by OR set operation. If the binary decision of SVM classifier has true (+1) for at least one subtopic, we decided the document to be relevant for the user's interest.

2.2 Results

We submitted two runs, KAIST10bfo1 and KAIST10bfo2, for the batch filtering track. The TREC-10 filtering track used the Reuters test collection. In batch filtering, the number of training documents is 23,208 and the number of test documents is 822,805. We didn't use any other data. The number of features is 51,265. The weights of terms are calculated by ltc weighting scheme in SMART (Salton, 1983). The KAIST10bf01 run was generated by SVM, and the KAIST10bf02 run was generated by our method.

The batch filtering task was evaluated according to a utility measure (T10SU), a version of the F measure (beta=0.5), precision, and recall. We also evaluated the results according to macro averaged F1 and micro averaged F1. Table 4.1 shows the results. The result of KAIST10bfo1 differ little from KAIST10bfo2.

We expected that the unified results of SVM for subtopics might perform much better than SVM for a topic. However, the result is negative.

Topic set / Measure	KAIST10bfo1	KAIST10bfo2
MeanT10SU	0.295	0.298
F-beta	0.496	0.498
Set Precision	0.788	0.785
Set Recall	0.288	0.292
Macro averaged F1	0.379	0.384
Micro averaged F1	0.578	0.585

Table 1. TREC-10 Batch Filtering Results

3.Question and Answering Track

We participate in main task, list task, and context task in TREC-10. Our QA system operates on a set of documents retrieved by information retrieval system. For convenience, we worked with the top-ranked document set generated by NIST. First, 'Question Analyzer' analyzes the given question. It generates question types and extracts keywords of the given question. Then top documents retrieved by the information retrieval system are analyzed for extracting relevant answer. POS tagger and Named entity tagger are used for the purpose. Finally, 'Answer Extractor' generates relevant answers from named entity tagged documents using question type and keywords of the question.

3.1 Main Task

3.1.1 Question Analyzer

A question analyzer parses the given question to identify question types and extract keywords. We define seven kinds of question type for the expected answer as following:

<Person>, <Location>, <Organization>, <Time>, <Currency>, <Measure>, <OTHERS>

They are detected by various patterns (Lee, *et.al*, 2000) and WordNet (Miller, *et.al*, 1991) synsets. <OTHERS> question type is assigned to a question, when there is no pattern for the question.

For extracting keywords, POS tagger (Brill, 1992) and WordNet are used. Noun, adjective,

countable numeric and verb except “be (is, are, was, were)” and “do (do, does, did)” are extracted as keywords. Noun phrases in the question are also considered and are extracted with a CFG-styled grammar rule.

NP = (DT (JJ | JJR | JJS)) { NN | NNS | NNP | NNPS }*

If there is an acronym, its expanded form is added to keyword lists. For example, ‘Cable News Network’, which is expanded form of CNN, is added into keyword list for the question containing word ‘CNN’. A query expansion technique is used for a noun phrase and a keyword, which is the only keyword for the given question. They are expanded with noun phrases in the WordNet definition. In the question “What is autism[1]?”, for example, ‘abnormal absorption’, ‘communication disorders’ and ‘short attention span’ is added to keyword lists

3.1.2 Answer Extractor

Our answer extractor generates top-5 ranked 50byte phrases. Its inputs are keyword, question type of the given question and top-50 retrieved texts. The top-50 texts are tagged with a POS tagger and a named entity tagger. There are two steps in answer extractor. First, a candidate sentence group is selected. Since, we believe that the context is very important for selecting relevant sentences, we group the previous two sentences, the current sentence and the next two sentences. Among these groups, top-50 sentence groups are selected with a keyword and a question type. Second, selected sentence groups are partitioned into phrases with fixed length. In this step, we use a WordNet definition for detecting relevant answers for the question type <PERSON>, and <LOCATION>. If there is terms with <PERSON> or <LOCATION> tag in the partitioned phrase and a question type of the question is <PERSON> or <LOCATION>, the number of keywords, which appear in WordNet definition of the term[2] is used as a feature for extracting answer. Scoring formula for is as follow:

$$Score(s_i) = 0.5 \times S(s_i, k_j) + 0.25 \times S(s_i, ek_j) + 0.25 \times (S(d_i, ek_j) + S(d_i, k_j))$$

where, S(A,B) represents scoring function for sentence group A and keyword B, s_i represents the i^{th} sentence group, k_j and ek_j represent keywords and expanded

[1] Its definition in WordNet is “autism -- ((psychiatry) an abnormal absorption with the self; marked by communication disorders and short attention span and inability to treat others as people)”

[2] We call the term as a boosting term

keywords for the j^{th} question, and d_i represents WordNet definition for a boosting term in the i^{th} sentence group.

If the $Score(s_i)$ of the top ranked phrase is below T (T is threshold), it is determined that there is no answer for the question in the text.

We expected that the results of this year might perform much better than that of last year (Lee, *et.al*, 2000). However, the result is negative. Since, list task and context task use the module of main task, the result of those is not really good.

3.2 List Task

List task requires the given number of answers. In TREC-10 list task, each question has information of required answer number. Question analyzer should give a question type and the number of the answer. Answer extractor should give the proper number of answers.

For example, “Question Number 1: Name 20 countries that produce coffee.”, the question requires twenty answers. Question analyzer just finds number sequence and passes it to the answer extractor.

Answer extraction processes of list task are similar to those of main task except for the limitation of number of the answer. We extended the passage search algorithm for giving the right number of answer.

Answer extractor consists of two phases.

1) Candidate answer listing
2) Finding the right number of answers

First, ‘Candidate answer listing’ finds the passages and candidate answer set of the passages. We sort the candidate answer by the frequency. Candidate answers are marked with named entity tagger and if a question type and a named-entity type are equal, the entity is added to an answer list. The answer list is sorted by the frequency of the each candidate.

Second, ‘Finding the right number of answers’ explores the passages for making the answer set. Answer set select by the rules as follow.

1) Answers in the same passages are selected all or nothing.
2) Answers that are on the previous answer passages are deleted from the list.
3) Answers that are included at the highly ranked passages but not in the list are excluded from the output.
4) If the numbers of the candidate answer list are smaller than the needed answer number, we uses the same methods for main task and stop when reached the right

number of the output.

Answers are weighted by normalized frequency of answers and passage weight. The passage-weighting scheme is the same as that of main task. Frequencies are normalized by total frequency and scaled up for balancing passage weights. Frequencies of answer sets are just added to each answer's frequency. It makes the passages to contain more candidates in top-rank.

We combine the frequency and the passage weight using the geometric mean.

$$Weight_{passage} = \sqrt{\left(S \bullet \sum_{i \in answer} freq_{i}^{list} \right)^2 \bullet passage^2}$$

If passages have no candidate answer, the weight is zero; it is similar to context that contains no support information.

3.3 Context Task

In this TREC-10, the context task is introduced for the QA system to exploit context when answering questions. Each individual question in this task has short, fact-based answers as in the main task, and each question has an answer in the collection. However, the interpretation of the question depends on the meaning of and answers to one or more earlier questions in a series. Interpreting a question correctly often involve resolution of referential links within and across sentences (TREC, 2001).

The QA system for the main task did not prepare any solution for this referential link problem. An anaphora resolution module and a keyword expansion module were made up for this weak point of the main task QA system. The system architecture is shown at figure 1. The anaphora resolution module recognizes the anaphora and finds its referent. This resolved referent could be added to the question keyword list.

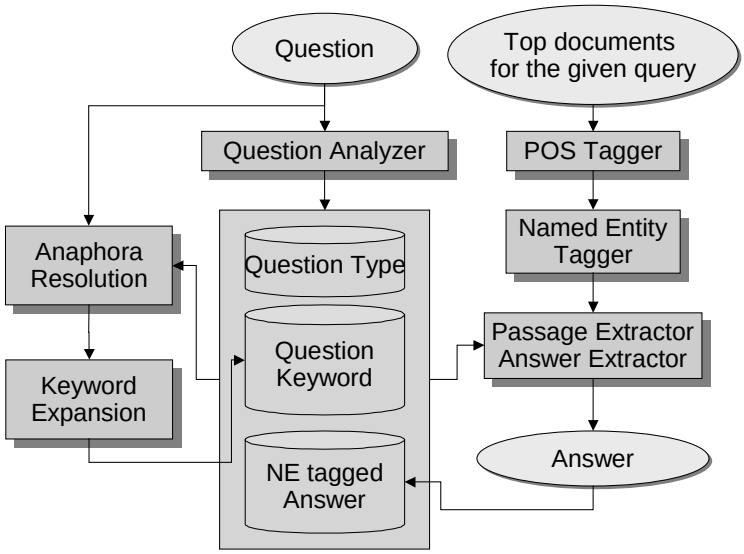

Fig 1 System Description for context task

The anaphora resolution module is composed of NP extractor, anaphora finder, and referent finder. For extracting noun phrase from previous questions and answers, NP extractor has some CFG-styled grammar rules. All found NPs are added to a referent candidate list. Some of the rules are shown as following:

NP = (DT (JJ | JJR | JJS)) { NN | NNS | NNP | NNPS }*

NP = { WP | WP$ } ({ JJ }* NP)

These rules were implemented as FSN (Finite State Network). And, after some rules are exchanged, this FSN is also used to find the anaphora from the question. For all extracted NP, keyword type was given as ‘PERSON’, ‘LOCATION’, …, ‘Other’.

When it finds the anaphora for the given question, the anaphora resolution module does its duty, looking up the referent candidate list as following sequences.

1) It tries to match the keyword type and agreement between anaphora and referent candidates of the previous question.
2) If the referent was not found, it considers others of the given question series.
3) If the decided referent is WH-pronoun, It selects the real referent from the answer list.

The resolved referent could be added to the question keyword list, and real QA process is working on the top 50 documents of the first one of question list.

References

Brill, E. (1995). Transformation-Based error-driven learning and natural language processing: a case study in part of speech tagging. Computational Linguistics

Miller, G.A., R.Beckwith, C.Fellbaum, D.Gross, and K.Miller. (1991). Five Papers on WordNet. International Journal of Lexicography.

Joachims, T. (1998). Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In Proceedings of the European Conference on Machine Learning (ECML).

Lee, K.S., J.H. Oh, J. Huang, J.H. Kim, and K.S. Choi. (2000). TREC-9 Experiments at KAIST : QA, CLIR, Batch Filtering, TREC-9

Rocchio, J.J. (1971). Relevance feedback in information retrieval. In THe SMART Retrieval System - Experiments in Automatic Document Processing, (pp. 313-323). Prentice Hall, Inc.

Salton, G. & McGill, M.J. (1983). Introduction to Modern Information Retrieval, McGraw-Hill, Inc.

Salton, Gerard & Buckley, Chris. (1990). Improving retrieval performance by relevance feedback. Journal of the American Society for Information Science, 41(4):288-297.

Singhal, Amit & Mitra, & Mandar & Buckley, Chris. 1996. Learning routing queries in a query zone. In Proceedings of the Twentieth ACM SIGIR Conference on Research and Development in Information Retrieval, (pp. 21-29).

TREC (2001). http://trec.nist.gov/act_part/guidelines/qa_track_spec.html, TREC 2001 Question Answering Track Guideline,

Vapnick, Vladimir N. (1995). The Nature of Statistical Learning Theory, Springer-Verlag, New York.