Question Classification by Ensemble Learning

LI Xin, HUANG Xuan-Jing, WU Li-de

Dep. of Computer Science and Engineering, FUDAN Univ., Shanghai, PRC.

Summary
One of the major and crucial difficulties in Question Answering (QA) consists in reducing the gap between question and answer to pair them. In this perspective, Question Classification (QC) appears as an important module as it indicates the answer type from the question semantic. The paper present the particularities of our question classification, based on the use of linguistic knowledge and machine learning approaches. Different classification features and multiple classifier combination method are exploited. By using compositive statistic and rule classifiers, and by introducing dependency structure from Minipar and linguistic knowledge from WordNet into question representation, the research shows high accuracy in question classification.

Key words:
Question Answer, Classify, SVM , TBL.

Introduction

The World Wide Web continues to grow at an amazing speed. So, there are also a quickly growing number of text and hypertext documents. Due to the huge size, high dynamics, and large diversity of the web, it has become a very challenging task to find the truly relevant content for some user or purpose. The Open-domain Question Answering system (QA) has been attached great attention for its capacity of providing compact and precise results for users.

This paper presents our approaches at Question Answering to improve the precision of question categorization. Our method combines statistic and rule classifiers, specifically statistics preceding regulation, to classify questions. With rule classifier as supplementary to statistic, the advantages of respective classifier can be given full play to, and therefore the overall performance of the classifier combination will be better than the single one. Moreover, as far as the Question Classification task is concerned, the paper compares various classifier combinations, statistic-rule classifier, voting, Adaboost and ANN. To represent questions, the paper uses dependency structure from Minipar[1] and linguistic knowledge from WordNet [2, 3].

In the following parts of the paper, the first segment is about question classification within QA systems. After this overview, classifying method and features which also aim at improving the categorization performance are introduced, and then we comparisons are made among different type features and different feature combination methods. The comparisons are testified in experiments based on the use of linguistic knowledge and Ensemble Learning. The last part of the paper is about the conclusion of the present research and about the introduction of the further work to be done on this issue.

2. Question Classification in QA

2.1 Question Answer System

How to acquire accurate and effective information has become one of the great concerns among Internet users. QA has gain great popularities among scholars who care the above challenge [4, 5], for QA can meet users' demand by offering compact and accurate answers, rather than text with corresponding answers, to the questions presented in natural language. It moves forward from document retrieval to information retrieval. Therefore, it saves users' great trouble to find out specific facts or figures from large quantity of texts.

A classical QA system is composed of several components: a question analyzer and a question categorizer, a document retrieval software that retrieves candidate documents (or passages) according to a query (the query is automatically derived from the question), a fine-grained document analyzer (parsers, named-entity extractors, …) that produces candidate answers and a decision process that selects and ranks these candidate answers.

Most of QA question categorizers take natural questions as input to produce answer categories. In order to categorize questions, most QA system developed question patterns based on the collection of questions and employed a tokenizer, a part-of-speech tagger and a noun-phrase chunker etc. .

2.2 Question Classification

QC can be defined to match a question to one or several classes in K category so as to determine the answer type. Every class presents some semantic restrictions on the answer searching, which serves QA with various strategies in locating the correct answer.

Manuscript received January 10, 2006.
Manuscript revised March 30 , 2006.

The study of Question Classification, as a new field, corresponds with the research of QA. At present the studies on QC are mainly based on the text classification. Though QC is similar to text classification in some aspects, they are clearly distinct in that : Question is usually shorter, and contains less lexicon-based information than text, which brings great trouble to QC. Therefore to obtain higher classifying accuracy, QC has to make further analysis of sentences, namely QC has to extend interrogative sentence with syntactic and semantic knowledge, replacing or extending the vocabulary of the question with the semantic meaning of every words.

In QC, many systems apply machine-learning approaches[6-8]. The classification is made according to the lexical, syntactic features and parts of speech. Machine learning approach is of great adaptability, and 90.0% of classifying accuracy is obtained with SVM method and tree Kernel as features. However, there is still the problem that the classifying result is affected by the accuracy of syntactic analyzer, which need manually to determine the weights of different classifying features.

Some other systems adopting manual-rule method make QC, though may have high classifying accuracy, lack of adaptability, because regulation determination involves manual interference to solve the conflicts between regulations and to form orderly arranged rule base. Sutcliffe states that simple ad-hoc keyword-based heuristics allowed his system to correctly classify 425 of the 500 TREC-11 questions into 20 classes[9]. The QUANTUM Q&A system[10] employed 40 patterns to correctly classify 88% of the 492 TREC-10 questions into 11 function classes (a function allows to determine what criteria a group of words should satisfy to constitute a candidate valid answer). They added 20 patterns for the TREC-11 evaluation. Last but not least, the MITRE corporation‘s system Qanda annotates questions with part-of-speech and named entities before mapping question words to an ontology of several thousand words and phrases[11].

Moreover, Bellot[12] experiment confirms that the combination of a small set of manually and quickly built patterns in conjunction with a probabilistic tagger yields very good categorization results (80% precision with several dozens of categories) even if an extensive rule-based categorizer may perform even better[13]. Bellot developed a rule-based tagger and employed a probabilistic tagger based on supervised decision trees for the question patterns that did not correspond to any rule. The probabilistic tagger was based on the proper names extractor presented during ACL-2000[14]. This module used a supervised learning method to automatically select the most distinctive features (sequence of words, POS tags…) of question phrases embedding named entities of several semantic classes. The learning process generates a semantic classification tree[15] that is employed to tag a new question. By using a subset of 259 manually tagged TREC-10 questions as a learning set,

The main input of the rule-based tagger was a set of 156 manually built regular expressions that does not claim to be exhaustive since it was based on previous TREC questions only. Among the 500 TREC-11 questions, 277 questions were tagged according to these rules. But this method has a significant weakness that need write rule manually, and our method improved it.

2.3 Important Role of Question Classification

Question Answering systems are based on Information Retrieval techniques. This means that the question asked by the user is transformed into a query from the very beginning of the process. Thus, the finest nuances are ignored by the search engine which usually:

1) transforms the question into a « bag of words » and therefore loses meaningful syntactical and hierarchical information;

2) lemmatizes the words of the query if necessary, which deletes information about time and mode, gender and number (singular vs. plural);

3) eliminates —stop words although they may be significant.

However, if the user has got the opportunity to ask a question thanks to a QA system, it is not only to obtain a concise answer but also to express a complete and precise question. But when the question is transformed into a bag of words, a lot of information is lost. For instance, the question *How much folic acid should an expectant mother get daily?* becomes: *folic + acid + expectant + mother +get + daily* when transformed into a query. However six terms are not enough to specify what the user is seeking exactly. It is necessary to tell the system that we are looking for a quantity. This is precisely what question categorization can do.

Question Classification is an essential part of Question Answering system, for to correctly answer users' questions, the system has to know what the users are looking for, and it is QC that presents important searching clues for the system. The result of QC can also serve QA in the answer selecting and extract, which influence the performance of QA directly. The first reason is that QC minish searching space. For example, if the system knows that the answer type to the question *"Who was the first astronaut to walk in space?"* is a person's name, it can confine the answer in the names, rather than every word in the texts. The second reason is that QC can determine the searching strategies and knowledge base QA may need. For instance, the question *"What county is California in?"* needs the name of a country as its answer, so system needs the knowledge of countries' name and name entities tagging to identify and testify the place name, while the question *"What is*

Teflon?" expects an answer in a sentence or a fragment, in the form of Teflon is <…. >. In fact, almost all the QA have the QC module and QC is the one of the most important factors what determines the QA system performance[5].
Finally the type of the question may be very helpful for generating the query and retrieving candidate documents. For example, if the answer type of a question is — length", the query generated from the question may contain the words — miles, kilometers". A set of words may be associated with each answer type, words being candidates for query expansion.

3. Classifying Features

In machine learning method, every question should at first be transformed into a feature vector. Bag-of-word is one typical way of transforming questions, where every feature is one word in a corpus, whose value can be Boolean, showing whether the word is present in questions, and which can also be an integer or a real number, showing the presence frequency of the word. In this paper, every question is represented as a Boolean vector.
1. Bag-of-word: all lexical items in questions are taken as classifying features.
As stop words such as *"what"* and *"is"* do appear on many occasions, they are considered less significant than other words and are not taken into account by search engines. However, stop words play an important role in QA. First, their meaning can be useful during the categorization phase. Secondly, they can help locate the answer during the extraction phase. In this case, stop words must be kept in the query. For example, the question *How far away is the moon?* could become a one- keyword query: *moon*. Without any other information, this simple query may find no answer to question 206 in a document collection. In order to find the right answer, information about the answer type needs to be added. For question, we could mention that we are looking for a distance: the distance between the Earth (implicit data which needs to be made explicit!) and the moon.
2. WordNet Synsets: WordNet was conceived as a machine-readable dictionary. In WordNet, word form is represented by word spellings, and the sense is expressed by Synsets, and every synset stands for a concept. WordNet shows both lexical and semantic relationships. The former exists between word forms, while the latter exists between concepts. Among various semantic relations in WordNet, we choose hypernyms between nouns as our only concern. The classifying features are the senses of the nouns in the sentences and synsets of their hypernyms.
3. N-gram: the model is founded on a hypothesis that the presence of a word is only relevant to the n words before it. The frequently used are Bi-gram and Tri-gram, and Bi-gram is chosen as the classifying features in the present research. Compared with word, Bi-gram model investigates two historical records, and reflects the partial law of language. It embodies the features of word order, and therefore it can reflect the theme of the sentence more strongly.
4. Dependency Structure: Minipar is a syntactic analyzer, which can analyze the dependency relation of words in sentences. It describes the syntactic relationships between words in sentences. Such relation is direction-oriented, semantically rather than spatially, namely one word governs, or is governed by, another concerning their syntactic relation. In one sentence *(W1W2 ⋯ Wn)*, compared with Bi-gram, Dependency structure concerns *WiWj* , but not need limitation *j= i+1*. Obviously, Dependency Relation goes further than Bi-gram in language understanding. Dependency structure is specified by a list of labeled tuples. The format of a labeled tuple is as follows:

label (word pos root governor rel exinfo …)

"Label" is a label assigned to the tuple. If the tuple represents a word in the sentence, label should be the index of the word in the sentence. *"Word"* is a word in the input sentence. *"Pos"* is the part of speech. *"Root"* is the root form. "Governor" if the label of the governor of word (if it has one), *"rel"* is type of dependency relationship, and *"exinfo"* for extra information. Minipar output is represented by the word dependency relationship via *"governor"*. Though only 79% of recall and some word relations fail to be analyzed, the accuracy reaches 89%, which guarantees that a large proportion of dependency relations from the output are correct. And the experiment proves that Dependency structure has more classify precision than Bi-gram as classifying feature.
For example, as to the question *"Which company created the Internet browser Mosaic?"* Minipar may produce the following results:

*E0 (() fin C *)*
1 (Which ~ Det 2 det (gov company))
2 (company ~ N E0 whn (gov fin))
3 (created create V E0 i (gov fin))
E2 (() company N 3 subj (gov create) (antecedent 2))

……

According to the tuple, we can get dependency relationships between words in sentences. For example, tuple 1 *(Which ~ Det 2 det gov company)* shows us the *det* relationship between *"which"* and *"company"* in the sentence. Therefore, we can get a words-pair（which company）, and likewise other five pairs of words can be obtained —（*company create*）、（*the Mosaic*）、 *(Internet Mosaic)* 、 *(browser Mosaic)* 、 *(create Mosaic),* which will be the item of vector represented the question.

4. Classifying Method Description

Ensemble Learning combines multiple learned models under the assumption that "two (or more) heads are better than one." The decisions of multiple hypotheses are combined in ensemble learning to produce more accurate results. Boosting and bagging are two popular approaches. Our work focuses on TBL (Transformation-Based Error-Driven Learning) combining multiple classifier that are more effective than those built by BP and Voting methods, and, in particular, are useful for increasing more classifier.

4.1 Support Vector Machine (SVM)

SVM is a kind of machine learning approach based on statistic learning theory. SVM are linear functions of the form f (x) = <w•x> +b, where <w•x> is the inner product between the weight vector w and the input vector x. The SVM can be used as a classifier by setting the class to 1 if f(x) > 0 and to -1 otherwise. The main idea of SVM is to select a hyperplane that separates the positive and negative examples while maximizing the minimum margin, where the margin for example xi is yi f(x) and yi ∈[-1,1] is the target output. This corresponds to minimizing <w•w> subject to yi (<w•x> +b) ≥ for all i. Large margin classifiers are known to have good generalization properties. An adaptation of the LIBSVM[16] implementation is used in the following. Four type of kernel function linear, polynomial, radial basis function, and sigmoid are provided by LIBSVM .

4.2 SVM-TBL QC Algorithm

Transformation-Based Error-Driven Learning (TBL) has been a part of NLP since Eric Brill's breakthrough paper[17] in 1995, which has been as effective as any other approach on the Part-of-Speech Tagging problem. TBL is a true machine learning technique. Given a tagged training corpus, it produces a sequence of rules that serves as a model of the training data. Then, to derive the appropriate tags, each rule may be applied, in order, to each instance in an untagged corpus.

TBL generates all of the potential rules that would make at least one tag in the training corpus correct. For each potential rule, its improvement score is defined to be the number of correct tags in the training corpus after applying the rule minus the number of correct tags in the training corpus before applying the rule. The potential rule with the highest improvement score is output as the next rule in the final model and applied to the entire training corpus. This process repeats (using the updated tags on the training corpus), producing one rule for each pass through the training corpus until no rule can be found with an improvement score that surpasses some predefined threshold. In practice, threshold values of 1 or 2 appear to be effective.

Therefore, we present compositive QC approach with rule and statistic learning. At first, questions are represented by Bag-of-word, WordNet Synsets, Bi-gram, and Dependency structure, and are classified by the same samples and same SVM. Then output of SVM is transformed to the input of TBL, and thus every sample in TBL training data is featured by four-dimensioned vectors, from which a new is obtained as training data of TBL. When the errors produced in initial marking process are corrected in TBL to the greatest extent, a final-classifier is produced as follows (Fig. 1).

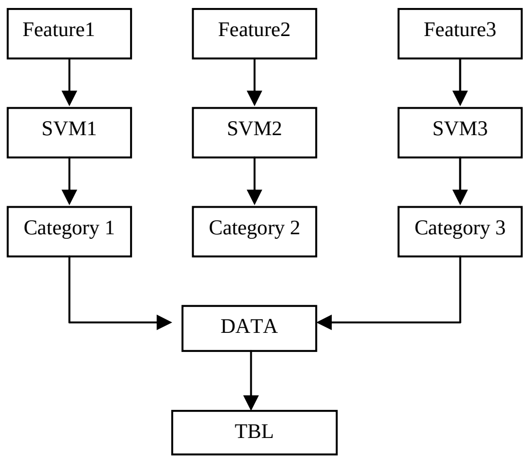

Fig. 1 SVM-TBL QC Algorithm

Transformation-Based Error-Driven Learning is composed of three parts: unannotated text, transformation templates, and objective function. In the experiment, unannotated text is obtained from SVM. The transformation templates define the space of transformations; here is combination of SVM output. Suppose we have k basic classifiers, and each classifier may put questions into N types, then we have rule templates. Objective function is the precision of classifier.

5. Results and Analysis

The research adopts the same UIUC data and classifying system as Zhang[8] shows. There are about 5,500 labeled questions randomly divided into 5 training sets of sizes

1,000, 2,000, 3,000, 4,000 and 5,500 respectively. The testing set contains 500 questions from the TREC10 QA track. Only coarse category is test.

5.1 SVM Classifying Result

We experiment the QC by SVM with four kernel function, and the following table (Table1) is the illustration of classifying accuracy by using single-kind classifying feature.

It is shown that as to the four type features, no matter what Kernel is used, using Dependency relation feature have more precision than others and feature of Synsets is better than Bag-of-word. Therefore it is safe to draw the conclusion that Synsets and dependency relationship are helpful to represent questions. Among the four Kernel function, Liner has the best classifying precision. That is why we use Liner in the following experiment.

Table1. Four kernel function Question Classifying Accuracy (%)

Num. of Training / Kernel & feature		1000	2000	3000	4000	5500
Liner	Bag-of-word	79.6	81.2	83.4	85.8	84.8
	WordNet	77.8	83.8	85.2	86.4	86.8
	Bi-gram	73.6	80.6	83.2	87.4	88.6
	Dependency	82.0	86.8	87.2	88.4	89.2
Polyn-omial	Bag-of-word	52.4	69.2	66.0	61.4	62.6
	WordNet	48.4	69.8	70.0	68.8	73.2
	Bi-gram	27.6	49.2	46.4	49.6	50.8
	Dependency	73.0	78.8	81.8	82.4	85.2
RBF	Bag-of-word	68.8	73.2	80.2	81.4	83.6
	WordNet	69.0	73.2	79.8	80.2	81.0
	Bi-gram	62.2	70.2	76.0	80.0	81.2
	Dependency	72.8	78.8	81.0	83.2	85.0
Sig moid	Bag-of-word	65.6	74.2	77.0	78.2	80.2
	WordNet	74.2	82.6	83.4	83.8	84.4
	Bi-gram	68.6	74.4	79.8	83.2	84.8
	Dependency	75.2	78.0	82.4	83.4	85.2

5.2 Result of SVM multi-kind-feature classification

A question can be represented directly as a vector with multi-kind-features: Bag of Word, Dependency Structure, Synonym and Bi-gram. Fig.2 provides an accuracy comparison of the results derived from classification with four features and classification with only one kind feature. Experimental result indicates that, results from classification with four type features do not excel the best classification precision with only one feature.

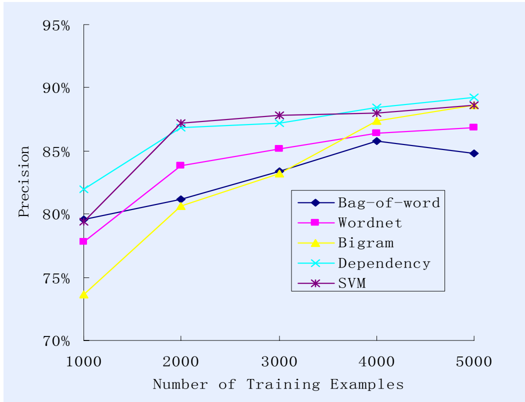

Fig. 2 Multi-type Feature

5.3 Using Adaboost to combine several classification results

Multi-classifier combination is often used to obtain better classification results. Adaboost[18, 19] is an effective classifier combination method. Yet in question classification training, chances of samples to be faultily classified are slim. Therefore, greater accuracy on classification can hardly be realized with Boost.

5.4 Using BP to combine several classifiers

We have also tried to use nerve network to combine the output results of 4 classifiers. We build a BP network with 4 input nodes and 1 output node. The number of hidden nodes chosen comes from the empirical formula: *m=sqrt(nl)*, whose "*m*" indicates hidden nodes, "*n*" input nodes, and "*l*" output nodes. Thus, the number of hidden layer nodes is "*2*".

Fig.3 shows, when training samples are relatively less, classification accuracy of BP is greater compared to that of single-feature classifier, but not in cases where the number of samples increases.

5.5 Using the method of voting to combine several classifiers

Through the method of voting, we can also get the combination results, according to the class label outputted by SVM with different type features. Experimental results are given in Fig. 4. We may see that, due to the rule of "more votes winning" in voting, when there are a number of not-so-accurate classifiers, the accuracy of voting can not compete with the greatest accuracy of a single classifier.

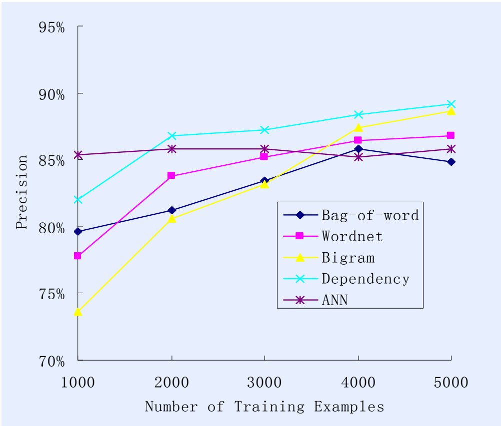

Fig. 3 BP combine several classifiers

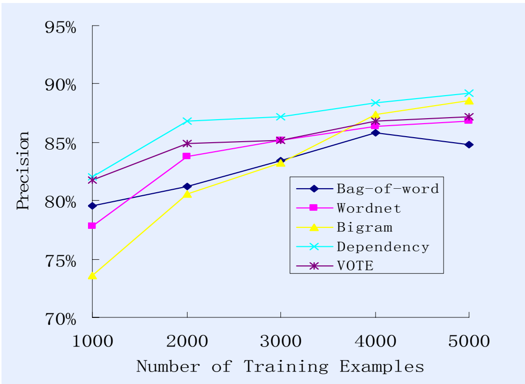

Fig. 4 Voting combine several classifiers

5.6 Using TBL method to combine several classification results

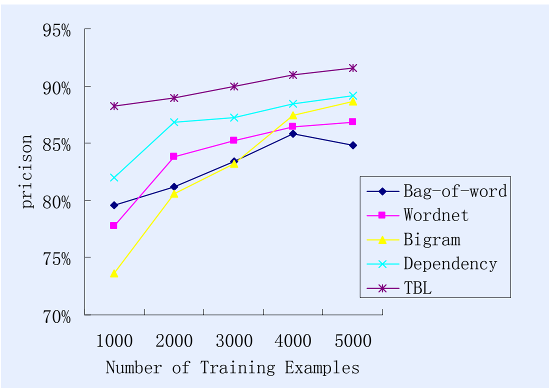

Fig. 5 TBL combines several classifiers

Fig. 5 displays the accuracy of a number of classification results in TBL combination. In our experiment, we construct 5 test-training sets, using 5500 sentences in UIUC. Each test-training set has 1000 stochastically chosen questions as its test set, and the other 4500 as its training. The TBL training set is built upon the SVM classification results from the test set. In comparison with the method to voting, TBL uses the conversion rule to fully rectify the errors of initial tagger. Therefore, TBL classification will not produce results inferior to the best results of initial tagging.

We obtain from the experiment all together 251 conversion rules, the foremost ones of which are listed as follows. From these rules which come from TBL training, we may also deduce that, TBL makes use of, firstly, the results of the most accurate classifier (parser), and secondly, the results of other classifiers, especially those of dependency structure rectified by Bi-gram results. It puts the accuracy of SVM single-feature classification into full use to secure greater accuracy.

1. Parser_2 $$ _#=_2
2. Parser_3 $$ _#=_3
3. Parser_1 $$ _#=_1
4. Parser_5 $$ _#=_5
5. Parser_4 $$ _#=_4
6. Parser_3 && Bigram_2 && Synset_2 && BagOfWord_2 $$ _3=_2
7. Parser_3 && Bigram $$ _3=_1
8. Parser_0 $$ _#=_0

………

Rule *1* shows that: in cases where Dependency Structure is adopted as the feature, when the classification result is 2 and the question is not classified, the question belongs to the second class. Rule *2*, *3*, *4*, and *5* is similar to *1*.

Rules *6*, *7* involve classification results from multiple classifiers. Rule *6* indicates that, if sentence is placed in class 3 when Dependency Structure is adopted as feature, in class 3, when Bi-gram or Synset or Bag-of-Word is adopted, and questions have already been tagged as class 2, it will be put in class 2.

Fig. 6 gives us the classification results of 500 questions of Trec10 in different method of combination. It can be seen that, TBL combination of classifiers is better than voting and ANN; TBL and SVM working together is better than SVM classification using multi-type-features to represent questions directly.

Fig. 7 provides a comparison of classification accuracy between TBL combining multi-classifier and SVM directly using several type features, in conditions of adopting or not adopting Dependency Structure as feature. TBL- and SVM- both mean classifier not adopting. The results show: Using such method of QC as blending "statistics" and "rules", that is, the accuracy of classification is 1.6%

greater than that of not using TBL; adopting Dependency Structure as feature can also promote precision, with a percentage of 1.8 .

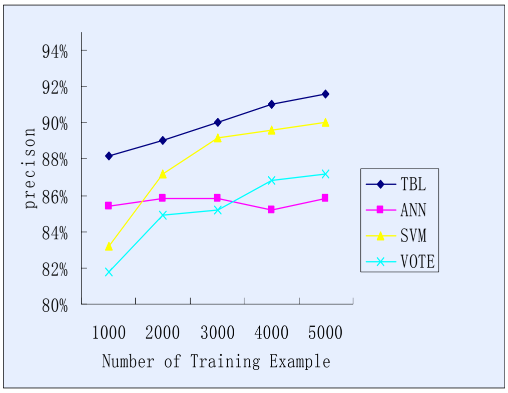

Fig. 6 Different combine method

Fig. 7 Using dependency structure or not

5.7 Result Analysis

Compared to Zhang[8] using "tree kernel" as the classification feature, this thesis adopts the "statistics and rules blended" method in QC ("statistics first and rules next"), lifting the precision of classification to 1.4% higher than it used to be. Moreover, it also avoids the problem of artificial selection in different feature weighting that appearing in Zhang's paper.

Tests using the "statistics and rules blended" pattern of question classification unfold that, 34.1% of faulty classification of sentences arouses from the using of improper statistical methods. The manifest of this is that all the SVM classifiers with 4 features place questions into class "i", while they actually belong to class "j". Classification features that have relatively big differences are needed to work as basic classifier to improve the final result. And also, 31.8% of the faulty classification stems from the fact that, there are no corresponding rules in the rule sets derived from TBL training, so that the rule sets cannot correct the errors caused by wrong statistical methods. This may because our question corpus is Limited, and therefore, some of the classification combinations never even appear.

Table 2. The confusion matrix of category

	ABBR	DESC	ENTY	HUM	LOC	NUM
ABBR	7	2	0	0	0	0
DESC	0	136	1	1	0	0
ENTY	0	9	79	5	2	0
HUM	0	0	2	61	2	0
LOC	0	2	4	1	72	1
NUM	0	5	2	1	1	104

Table 2 presents the confusion matrix of the best classifier, that on the 9 test questions in the ABBR category, 7 have been correctly classified and 2 have been misclassified into DESC category. The most common confusions happen between ENTY and DESC, which is not surprising because the classifier are forced to do disjoint classification.

6. Conclusions

Question answering systems use IR and IE methods to retrieve documents containing a valid answer. Question classification is one importance role in the QA frame to reduce the gap between question and answer. It can conduct answer choosing and selection. Our question classification method based on the use of linguistic knowledge and machine learning approaches and it exploit different classification features and multiple classifier combination method, also. Moreover, our SVM_TBL classifier easily combines other features question and classifier.

This thesis experiment several different methods in QC, and studies features like the Dependency Structure, WordNet Synsets, Bag-of-Word, and Bi-gram. It also analyzes a number of kernel functions and the influence of different ways of classifier combination, such as Voting, Adaboost, ANN and TBL, on the precision of QC. Adopting the "statistics and rules blended" method of question classification ("statistics first and rules next") and using language information such as the Synset from WordNet and the dependency structure of Minipar as classification features promote the accuracy of question classification. TBL combination multi-classifier method can be extended, easily. As long as new classifying algorithm or new feature set is found, the classifying result from them can be transformed to rule set, which can lead to further classifying function. WordNet has provided us with semantic relation, examples, explanation, etc. The present study only investigates the semantic relation of hyponymy. There are still much to be done in the future to further the research on QC using WordNet.

References

[1] Lin, D. 1998. "Dependency-based Evaluation of MINIPAR." In Workshop on the Evaluation of Parsing Systems. Granada, Spain.

[2] Miller, G. 1995. "WordNet: A Lexical Database for English." Communications of the ACM 38(11):39-41.

[3] Miller, G. and McDonnell, J. S. 2003. "WordNet 2.0." Princeton University's Cognitive Science Laboratory. http://WordNet.princeton.edu/

[4] Li, X. and Roth, D. 2002. "Learning Question Classifiers." In, Proceedings of the 19th International Conference on Computational Linguistics, (COLING'02). Taipei.

[5] Moldovan, D., PASCA, M. and HARABAGIU, S. 2003. "Performance Issues and Error Analysis in an Open-Domain Question Answering System." ACM Transactions on Information Systems Vol.21:133-154.

[6] Hovy, E., Hermjakob, U., Lin, C.-Y. and Ravichandran, D. 2002. "Using Knowledge to Facilitate Factoid Answer Pinpointing." In, Proceedings of the COLING-2002 Conference. Taipei, Taiwan.

[7] Ittycheriah, A., Franz, M., Zhu, W.-J. and Ratnaparkhi, A. 2000. "IBM's Statistical Question Answering System." In, Proceedings of the TREC-9 Conference. Gaithersburg, MD: NIST, p. 229.

[8] Zhang, D. and Lee, W. S. 2003. "Question Classification using Support Vector Machines." In, Proceedings of ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'03). Toronto, Canada: ACM.

[9] Sutcliffe R.F.E. (2003). Question Answering Using the DLT System at TREC 2002. In Proceedings of The Eleventh Text Retrieval Conference (TREC 2002). Gaithersburg, Maryland: NIST Special Publication SP 500-251.

[10] Plamondon L. & Lapalme G. & Kosseim L. (2003). The QUANTUM Question Answering System at TREC 11. In Proceedings of The Eleventh Text Retrieval Conference (TREC 2002). Gaithersburg, Maryland: NIST Special Publication SP 500-251.

[11] Burger J.D. & Ferro L. & Greiff W. & Henderson J. & Mardis S. & Morgan A. (2003). MITRE's Qanda at TREC-11. In Proceedings of The Eleventh Text Retrieval Conference (TREC 2002). Gaithersburg, Maryland: NIST Special Publication SP 500-251.

[12] Bellot P. & Crestan E. & El-Bèze M. & Gillard L. & de Loupy C. (2003). Coupling Named Entity Recognition, Vector-Space Model and Knowledge Bases for TREC-11 Question Answering Track. In Proceedings of The Eleventh Text Retrieval Conference (TREC 2002). Gaithersburg, Maryland: NIST Special Publication SP 500-251.

[13] Yang H. & Chua T.-S. (2003). The Integration of Lexical Knowledge and External Resources for Question Answering. In Proceedings of The Eleventh Text Retrieval Conference (TREC 2002), Gaithersburg, Maryland: NIST Special Publication SP 500-251.

[14] Béchet F. & Nasr A. & Genet F. (2000). Tagging Unknown Proper Names Using Decision Trees. In Proceedings of ACL'2000 (pp.77–84). Hong-Kong, China: ACL. Burger J. et al. (2000). Issues, tasks and program structures to roadmap research in Question Answering. In Proceedings of The Ninth Text Retrieval Conference (TREC 2000). TREC-9, Gaithersburg, Maryland: NIST Special Publication SP 500-251.

[15] Kuhn R. & de Mori R. (1996). The application of semantic decision trees to natural language understanding. In IEEE Transactions on Pattern Analysis and Machine Intelligence (pp. 449–460) 17(5).

[16] Chang, C.-C. and Lin, C.-J. 2001. " LIBSVM: a library for support vector machines." available at: http://www.csie.ntu.edu.tw/~cjlin/libsvm

[17] Brill, E. 1995. "Transformation-based error-driven learning and natural language processing: a case study part-of-speech tagging." Computational Linguistics 2:543-565.

[18] Schapire, R. E. 1997. "Using output codes to boost multiclass learning problems." In Machine Learning: Proceedings of the Fourteenth International Conference. pp. 313-321.

[19] Schapire, R. E. 1999. "Theoretical views of boosting and applications." In 10th International Conference on Algorithmic Learning Theory.