Question Classification with Support Vector Machines and Error Correcting Codes

Kadri Hacioglu, Wayne Ward
Center for Spoken Language Research
University of Colorado at Boulder
{hacioglu,whw}@cslr.colorado.edu

Abstract

In this paper we consider a machine learning technique for question classification. The goal is to replace our regular expression based classifier with a classifier that learns from a set of labeled questions. We have realized that an enourmous amount of time is required to create a rich collection of patterns and keywords for a good coverage of questions in an open-domain application. We decided to use support vector machines, since they have been successfully used for a number of benchmark problems. Although the support vector machines are inherently binary classifiers, it is possible to extend their use as multi-class classifiers using binary codes. We represent questions as frequency weighted vectors of salient terms. We compare our approcah to related work that uses relatively complex syntactic/semantic processing to create features and a sparse network of linear units to classify questions. We provide results to show performance of the method.

1 Introduction

In current Question Answering (QA) systems it has been widely accepted that the classification of questions into one of several question types is very important. Question type can be used for both query generation and answer extraction to improve the efficiency and accuracy of the overall sytsem. Approaches to question classification (QC) can be considered in two broad classes; namely, rule-based and statistical. In a rule-based approach a domain expert comes up with a number of regular expressions and keywords. In a probabilistic approach, the expert knowledge is replaced by a sufficiently large collection of labelled questions. Then a model is assumed and trained with the hope that the useful patterns for classification will be automatically captured. The latter approach has the advantage of saving expensive expert labor and having easier portability to other domains.

We assume that a user seeks an answer that fulfills his/her information need about an entity. Examples to entity types are PERSON and CITY as in the queries "Who invented the lamp?" and "Which city is the capital of Portugal?", respectively. We do not focus on questions like "Who is Bill Clinton?" , "How the earth rotates around sun?" etc., which are not asking for named entities, and we map all to a general category called NA (not applicable).

The first step in statistical QC is to design a taxonomy of question (equivalently, answer) types. One can distinguish among taxonomies of having flat and hiererchical structures, or taxonomies of having a small (10-30) and large number of categories (above 50). After analyzing several thousands of questions we created the Q/A entity type taxonomy shown in figure 1. Although it provides a detailed hierarcy of question types, one can collapse some categories or introduce new categories depending on the specifity required, coverage of the named entity recognizer over the documents, availability of training data, and required accuracy.

The second step is to create a corpus of labelled questions and train a classifier in a supervised manner. In this step the choice of features (to represent the questions) and the classifier (to assign questions into one of several categories) are very important. Features may vary from simple count statistics of words to detailed syntatic/semantic features derived using computationally demanding linguistic analysis. Classifiers can be implemented in several ways. Possible choices are naive Bayes classifiers, Neural network classifiers, vector-based classifiers, language model based classifiers and support vector machines (SVMs) (Sebastiani, 2002). We have chosen support vector machines for QC, motivated by the fact that

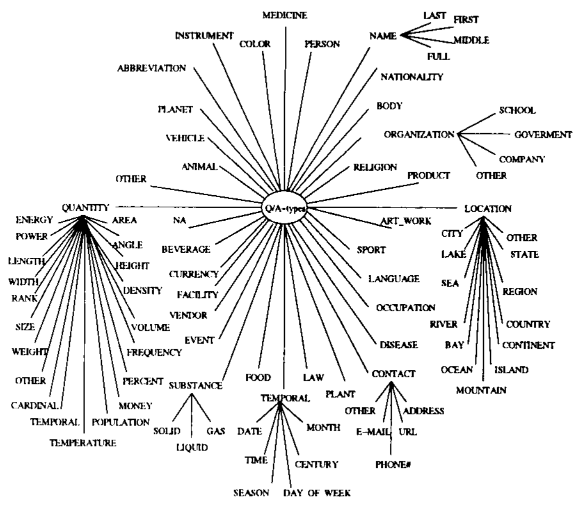

Figure 1: Question/Answer type taxonomy

they consistently outperformed other machine learning techniques in several tasks including the text classification (Rennie and Rifkin, 2001).

SVMs are binary classifiers (Burges, 1998). Although QC is a multi-class classification problem, it can be converted into a number of binary classification problems (Allwein et al., 2000). Then SVMs can be applied to learn each of these binary problems. Error correcting output coding has been found promising to convert a multi-class problem into a number of two-class problems (Dietterich and Bakiri, 1991; Berger, 1999).

In a related work (Li and Roth, 2002), words, part-of-speech (POS) tags, non-overlapping phrases (chunks), named entities (NEs), head chunks and semantically related words are derived from questions and used as primitive features. Then a set of operators were used to compose more complex features from the primitive features. The classifier is based on the SNoW (sparse Network of Winnows) learning architecture. It is a sparse network of linear units. The authors have made their training and test data publicly available. We tested our system on the same data and compared to their reported results. Using only words to create features, we have reached a performance level comparable with the result in (Li and Roth, 2002). The main advantage of our approach is that computationally expensive linguistic analysis and labor intensive creation of linguistic knowledge has been avoided.

2 Description of the Classifier

Our approach to classification is illustrated in Figure 1. The interesting aspect of this system is that all of its modules are constructed using readily available free software and a question corpus. We believe that our results would encourage many people to use these resources to deploy a descent statistical classifier in their systems.

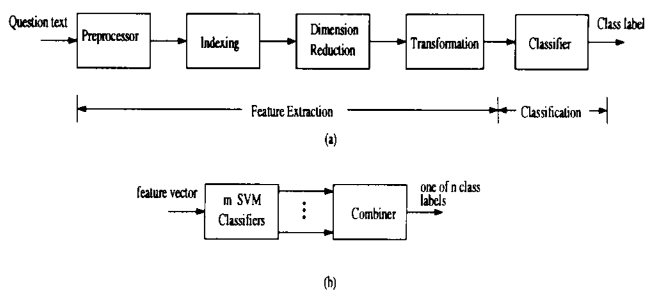

Figure 2: Classification scheme: (a) Overall system, (b) Classifier

The preprocessor tasks are (1) skipping of everything but a-z and A-Z characters (2) changing of each character to lower case, (3) excluding of each word in a user-defined "stoplist" and (4) stemming. One can augment the preprocessor with much sophisticated processing using a named entity tagger, part of speech tagger, chunker etc. The input to the indexer is a sequence of words and/or tags, and/or chunks (if a tagger or chunker is used). Based on the specification of a *term*, the indexer groups tokens into terms and accumulates their statistics. Often the dimensionality of the term space is unnecessarily high. Because of this, dimensionality reduction is very common before classification. It helps to reduce noise in the input space, avoid overfitting during training with sparse data and lower computational complexity. However, it should be done with care to minimize the loss of useful information for classification. Several techniques based on heuristics and information theory have been proposed. The final reduced space has basis defined by terms, and those basis may not be the optimal for the classification task. So it might be useful to transform the term space into another space. This can be done by computing transformations using singular value decomposition (SVD) , independent component analysis etc. (Srinivasan, 2002).

Question classification is a multi-class classification problem and SVMs are binary classifiers. An extension of SVMs to QC is possible using *codes*. Each class is assigned a codeword of 1's and -1's of length m. Here, m can be selected equal or greater than the number of classes. This splits the multi-class data into m binary class data. Therefore, one can design m SVM classifiers, combine their outputs and minimize a loss function to predict multi-class labels. One-vs-all, random and error correcting codes can be used for this purpose.

We used Rainbow (McCallum, 1996) for preprocessing, indexing and term reduction, SVDPACKC [1] for

[1]http://www.netlib.org/svdpack/svdpackc.tgz

transformation, SvmFu [2] for classification, and BCH codes [3] as error correcting codes. We used the same training data as in (Li and Roth, 2002) to train the classifier.

3 Experimental Results

The training data consists of 5500 labeled questions which have been made publicly available [4]. The total number of Q/A types is 50. The test data is 500 labeled questions from TREC-10. The final term space is created by indexing word tokens as N-grams and selecting the first N_I most informative terms. In addition to "words only" experiments, we carried out experiments using Identifinder, a named entity (NE) tagger developed at BBN (Bikel et al., 1997) with 7 and 29 tags. We used SVD to transform the term space into a new space spanned by SVD derived basis. We tried 50 × 50 one-vs-all (OVA) and 63-50 BCH codes. SVMs are trained using linear kernels. The *hinge* loss function is used in combining SVM outputs into a multi-class label.

Table 1: Question classification results with and without named entity (NE) tagger for different n-grams. The first most informative 2000 terms were retained in the feature space.

Method	1-gram	2-gram	3-gram
No NE	79.4%	80.2% (77.8%)	78.4%
NE-7	81.4%	82.0% (81.2%)	80.2%
NE-29	75.4	78.6% (79.2%)	78.8%

Inspection of a set of questions has shown that the inferior performance of NE-29 is due to (a) the 29-NE has a tagging performance worse than that of 7-NE, and (b) some words which are very discriminative across several question types are collapsed into a single broad tag (e.g. GPE_DESC includes words like city, state, country etc.) Results in parantheses show slightly inferior performance of the 50 × 50 OVA code.

The best result in (Li and Roth, 2002) for their flat classifier is 84.0%, using all features derived from words, NEs, POS tags, chunks, heads and related words. It should be noted that their "words only" performance is 52.4%.

Table 2: Question classification results with SVD transformation, NE-7 and BCH codes

Default,2000	Default 1000	SVD, 1000	SVD, 500
82.0%	81.8%	79.8%	79.2%

[2]http://five-percent-nation.mit.edu/SvmFu

[3]http://www.cs.cmu.edu/~ rayid/ecoc/ecoc-codes.tar.gz

[4]http://l2r.cs.uiuc.edu/~ cogcomp/Data/QA/QC

In Table 2 we compare the results obtained using the default terms and SVD vectors as basis for the feature representation. Its seems that the default terms as basis outperforms the SVD based basis. This agrees with the results reported in (Srinivasan, 2002).

4 Conclusions

We have introduced a statistical question/answer type classifier with a similar structure that has been recently considered for text categorization. We have augmented the system with a named entity tagger and SVD based transformation. Although the NE tagging has improved the performance, we have observed some performance loss with SVD transformation. A comparison with related work has shown that the approach is very promising yielding comparable performance with minimal and even with no linguistic analysis.

References

Erin L. Allwein, Robert E. Schapire, and Yoram Singer. 2000. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research*, 1:113-141.

A. Berger. 1999. Error-correcting output coding for text classification. *Workshop on Machine Learnibg for Information Filtering*, Stockholm, Sweeden.

D. Bikel, S. Miller, R. Schwartz, and R. Weischedel 1997 Nymble: a High-Performance Learning Name-finder. *In Fifth Conference on Applied Natural Language Processing*, pages 194-201.

C. J. Burges 1998. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*. 2(2), pages 121-167, 1998.

Tom G. Dietterich and Ghulum Bakiri. 2002. Error correcting output codes: A general method for improving multiclass inductive learning programs. *Ninth National Conference on Artificial Intelligence*, pages 572-577, Anaheim, CA, AAAI Press.

A. Kachites MCCallum. 1996. Bow: A toolkit for statistical language modeling, text retrival, classification and clustering. *http://www.cs.cmu.edu/~ mccalum/bow*.

Xin Li, Dan Roth 2002. Learning Question Classifiers. *COLING'02*.

J.D.M. Rennie, R. Rifkin 2001. Improving multiclass text classification with the support vector machine. *MIT Artificial Intelligence Laboratory Publications*, AIM-2001-026..

F. Sebastiani 2002. Machine learning in automated text categorization. *ACM Computing Surveys*, Vol. 34, No. 1, pages 1-47.

S. H. Srinivasan 2002. Features for unsupervised document classification. *Proceedings of CoNLL-2002*, pages 36-42. Taipei, Taiwan,. 2(2):121-167, 1998.