Question Classification

Gaurav Aggarwal, Richa Bhayani, Tejaswi Tenneti

December 10, 2009

1 Abstract

Question-Classification is the task of identifying the required answer type for any question posed to a Question-Answering system. The space of required answer types often varies based on the purposes of the Q&A system. So, it is more useful to do a two-level classification of the Question. We present a machine learning model which achieves this goal using various surface features present in a question. We have tested various features and analyzed the usefulness of each of these for classification purposes. Using a Maxent classifier our system is able to achieve an accuracy of 92% on the coarse-level classes and 86% on finer-level classes.

2 Introduction

Any question answering system consists of three major modules: 1) Question Classifier 2) Relevant documents retrieval 3) Pinpointing and scoring answer candidates. A Question classifier addresses the issue of determining the required 'answer type' for the question. An answer type can be a generic category like "humans", "location", "time", "definition" and can be classified into finer categories based on the requirements of that Question-Answering system. A simple method to achieve question classification (henceforth referred to as QC) would be to consider a bag-of-words model and use cue words like "who", "when" and so on to determine the answer type. However, question-answering systems can be domain specific (say, for lawyer or amongst the medical community), it follows that the required answer-types differ for each of the question answering systems. Hence, rule-based/template-based approaches suffer from poor scalability.

Furthermore, questions can be paraphrased in many ways and the answer-type is not immediately obvious. It is easy to see that this is a problem in questions starting with "What" and "How". We can see the difference between the questions "How many jews were killed in the holocaust ?"(required answer type is number) versus "How do the Nazis justify the killings of jews in the Holocaust?" (required answer type is manner). Thus we might need syntactic information of the question sentence for good classification. However, consider a "what" question like "What do bats eat ?" where the intended answer-type (which is subtle to determine) is of 'food' category. Here the syntactic structure of a sentence might not be useful for determining the focus in the question. Thus we require features at many levels i.e lower-order features containing syntactic information, and higher-order features like encoding of world-knowledge (like saying that 'a country is a kind of location'). Many such issues stand in the way of accurate Question-classification which makes it an interesting problem to tackle.

3 Dataset

The Cognitive Computation Group at UIUC has a dataset for question classification experiments : http://l2r.cs.uiuc.edu/ cogcomp/Data/QA/QC/. It contains 6 coarse grained classes (Abbreviation, Entity, Description, Human, Location and Numeric) and about 60 fine grained classes. There are 5500 labelled questions available for training and 500 test questions. Lists of semantically related words were also provided in the data set, which could also be obtained tools like word-net.

4 Approach

One goal of this project was to figure out what level of features would be useful to obtain a good classification at each of the levels described in the Motivation section. Just relying on surface features like n-grams, bag of words is not expected to perform well as they would not be able to capture the semantics of the question. So we have used higher order language features like POS tags, stems, synonyms/related words, interrogative pronouns, word category for sparse words, question focus.

4.1 Features

We use syntactic features like Named Entity Tags, Part-Of-Speech tags, and semantic features Related concepts to words present in the question. We are using Stanford's Named Entity Recognizer which tags each word in the question with one of 'Person', 'Location', 'Organization', 'Other' category. The POS tags features was used to capture the syntax patterns that similar kind of questions might share. The related concepts features is also a very useful feature to capture some amount of semantics inside the question. For example, 'away' and 'distant' belong to a list of words semantically related to the class 'distance'. The intuition is that some semantic classes might be good indicators of specific question categories. Other features include N-grams(uni,bi and tri).

4.2 Classifiers

We decided to choose Naive Bayes and Maxent for this stage of analysis, since NB often works well on text classification problems , and Maxent is a representative for conditional models.

4.2.1 Naive Bayes

Used a simple multiclass multinormial NB model.It assumes each feature is conditional independent to other features given the class. That is,

$$P(c|t) = \frac{P(c)P(t|c)}{P(t)} \tag{1}$$

where c is a specific class and t is text we want to classify. $P(c)$ and $P(t)$ is the prior probabilities of this class and this text. And $P(t|c)$ is the probability the text appears given this class. To avoid the problem of having zero frequencies. we make use of some smothing techniques. Otherwise, the likelihood will be 0 if there is an unseen word when it making prediction. We simply use add-1 smoothing in our project and it works well.

4.2.2 MaxEnt

We made use of Stanford classifier to code the Maxent section. The idea behind MaxEnt classifiers is that we should prefer the most uniform models that satisfy any given constraint. MaxEnt models are feature based models. We use these features to find a distribution over the different classes using logistic regression. The probability of a particular data point belonging to a particular class is calculated as follows:

$$P(c|d,\lambda) = \frac{exp\sum_i \lambda_i f_i(c,d)}{\sum_c exp\sum_i \lambda_i f_i(c,d)} \tag{2}$$

Where, c is the class, d is the data point we are looking at, and λ is a weight vector.MaxEnt makes no independence assumptions for its features, unlike Nave Bayes. This means we can add features like bigrams and phrases to MaxEnt without worrying about feature overlapping.

We also implemented SVM, however the results were not significantly better than Maxent.

5 Results

We ran the classifiers mentioned in 4.2 on our training set. Accuracy obtained on the test set is reported in table 1. As seen from the table, MaxEnt classifier performed the best.

5.1 Naive Bayes

Naive Bayes does well for coarse-grained classes but is only as good as a random classifier in case of fine-grained classification. Besides, in the latter case, there are many classes for which the classifier completely ignores and does not classify even a single question into those classes.

Training Set	Coarse-grained		Fine-grained	
	NB	MaxEnt	NB	MaxEnt
1000	0.754	0.848	0.500	0.732
2000	0.832	0.908	0.510	0.818
3000	0.828	0.904	0.518	0.822
4000	0.83	0.912	0.526	0.826
5500	0.862	0.918	0.534	0.856

Table 1: Accuracy for different classifiers

5.2 MaxEnt

MaxEnt was our best classifier achieving good accuracy for both coarse and fine-grained classification. It is also interesting to look at precision and recall scores achieved by MaxEnt on individual classes when trained on 5500 questions. This is illustrated in tables 2 and 3. For some fine-grained classes, the classifier does not predict that class at all. Such classes have `NaN` in the F1-score in the table. Figure 1 shows the learning curve for both class types.

Class	Precision	Recall	F-1
ENTY	0.89	0.78	0.83
DESC	0.89	0.98	0.93
NUM	0.96	0.96	0.96
ABBR	1.00	0.78	0.88
HUM	0.95	0.95	0.95
LOC	0.89	0.90	0.90

Table 2: Precision and Recall scores of MaxEnt for individual coarse classes

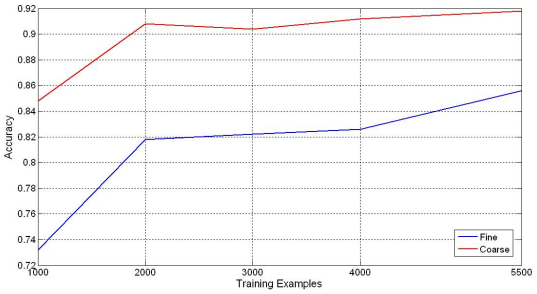

Figure 1: Learning Curve for MaxEnt

5.3 Feature Selection

In order to identify the best performing features, we did feature selection using *forward feature search* algorithm as seen in class. Tables 4 and 5 show successive iterations of the selection algorithm. At each iteration, we pick the best remaining feature and add it to the feature set. From the tables, one can conclude that 'part-of-speech' tag features (POS and POSCOMB) and 'related word' feature (RELWORD) correspond to the maximum gain in accuracy.

It was a bit un-intuitive to see the POS feature performing the best in the first iteration. We felt that since POS tag for the current word lack contextual information, POS feature would not provide useful signals. With this in mind, we had added the POSCOMB feature so that it captures syntactic structure of the sentence. But suprisingly it did very well. On the other hand, it makes sense to add semantic features, so we expected RELWORD feature to do well.

6 Analysis and Future Work

We also tried to do some error analysis on the MaxEnt classifier. For coarse grained classes, it achieves 90 plus accuracy for all classes except Abbreviation (ABBR) and Entity (ENTY). We observed that the classifier often confuses ABBR and ENTY class questions with Description (DESC) class. For example, *What is TMJ?* and *What does the technical term ISDN mean?* are both misclassified as DESC. Also, *What is* feature is common to many classes like Definition and Entity which might lead to some errors.

For abbreviations, it might help to add a binary feature which is activated whenever the question has a word containing all capital letters.

Chunking of entities in the question might also help. Consider this question - *What is the Ohio state*

Class	Precision	Recall	F-1
plant	1.00	0.20	0.33
abb	1.00	1.00	1.00
body	1.00	1.00	1.00
other	0.70	0.84	0.76
weight	1.00	1.00	1.00
desc	0.82	0.90	0.86
count	0.90	1.00	0.95
reason	1.00	1.00	1.00
state	0.64	1.00	0.78
date	0.98	1.00	0.99
lang	1.00	1.00	1.00
city	1.00	0.83	0.91
currency	1.00	0.33	0.50
substance	0.89	0.53	0.67
title	NaN	0.00	NaN
def	0.87	0.99	0.93
ind	0.93	0.98	0.96
event	0.00	0.00	NaN
techmeth	1.00	1.00	1.00
money	0.40	0.67	0.50
perc	1.00	0.67	0.80
animal	0.92	0.75	0.83
speed	1.00	0.50	0.67
instru	1.00	1.00	1.00
sport	1.00	1.00	1.00
period	0.80	1.00	0.89
country	1.00	1.00	1.00
product	NaN	0.00	NaN
exp	1.00	0.63	0.77
veh	1.00	0.50	0.67
termeq	0.78	1.00	0.88
mount	1.00	0.67	0.80
color	1.00	1.00	1.00
food	1.00	0.75	0.86
temp	NaN	0.00	NaN
manner	0.50	1.00	0.67
gr	1.00	0.50	0.67
dismed	1.00	0.50	0.67
dist	1.00	0.56	0.72

Table 3: Precision and Recall scores of MaxEnt for individual fine classes

New Feature Added	Accuracy
POS	87.2
POS-COMB	88.2
RELWORD	90.8
2-GRAM	91.8
3-GRAM	92
NER	91.6
1-GRAM	91

Table 4: Feature Selection for coarse classes

Current Feature Set	Accuracy
POS	81.8
POS-COMB	84.6
RELWORD	84.4
1-GRAM	85.4

Table 5: Feature Selection for fine classes

bird ?. This question definitely belongs to ENTY class as we are looking for the specific bird. However, our classifier labels this as Location probably because of the word 'Ohio'. Using clever NER and chunking algorithms, it may be possible to combine 'Ohio', 'state' and 'bird' into one entity 'Ohio_state_bird'. It would then be easier to identify this chunked question with the class ENTY.

Coarse-grained classifier does much better than the fine-grained one. This is in conjunction with the intuition that the more specific things get, unless we have more features, we will not get a very good performance. Hence, hierarchical classification seems to be a promising approach. We can use our existing coarse-grained classifier to predict the top two coarse classes. Then we train a separate classifier using only the questions that belong to those coarse classes and use that to predict one fine-grained class. In [1], the authors claim that hierarchical classification does not help. However, it would still be worth a try since we use different features than those used in the paper.

We could also try to add more semantic features given the success of the RELWORD feature. Using corpora like WordNet and FrameNet, we could try to more accurately identify the semantic classes of individual words occuring in the question.

In our experiments, SVM with a linear kernel does only as well as the MaxEnt classifier. Using non-linear kernels would be an interesting option to exploit higher dimensional features such as conjunction of current features.

7 Related Work

Dan Roth et al in their seminal paper [1] laid much of the groundwork for the task of Question Classification. The authors also used intelligent text based features and a heriarchical classifier using the SNoW algorithm. We borrowed some of their features like RELWORD and added some more like POS-COMB. However, we experimented with different classifiers and achieve similar accuracy.

In [2], the authors developed a novel *tree kernel* based SVM classifier that attempts to capture the syntactic structure of the questions.

References

[1] Xin Li and Dan Roth. Learning question classifiers. In *Proceedings of the 19th international conference on Computational linguistics*, pages 1–7, Morristown, NJ, USA, 2002. Association for Computational Linguistics.

[2] Dell Zhang and Wee Sun Lee. Question classification using support vector machines. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 26–32, New York, NY, USA, 2003. ACM.