
Learning to Classify Questions

Olga Feiguina and Balázs Kégl

Département d'informatique et de recherche opérationnelle

Université de Montréal

{feiguino, kegl}@iro.umontreal.ca

Abstract

An automatic classifier of questions in terms of their expected answer type is a desirable component of many question-answering systems. It eliminates the manual labour and the lack of portability of classifying them semi-automatically. We explore the performance of several learning algorithms (SVM, neural networks, boosting) based on two purely lexical feature sets on a dataset of almost 2000 TREC questions. We compare the performance of these methods on the questions in English and their translation in French.

1 Introduction

Many question-answering (QA) systems include a component that classifies natural language questions in terms of the type of answer expected. In most (but not all, see Related Work) systems, this is done using hand-crafted lexico-syntactic patterns. It is desirable to use learning algorithms to perform this classification to decrease the required manual effort and increase language portability, ideally maintaining the accuracy. Since no standard test set of questions nor even a standard set of question categories exist, it is hard to perform comparisons. A rough estimate of the coverage achieved by semi-automatic methods on TREC questions is 90% [3].

2 Data

We took a set of 1893 TREC questions in English and in French, the latter a translation of the former done at NRC by professional translators. The questions are largely factual, requiring a brief answer, e.g. “Where was George Washington born?” or “What are animals without a backbone called?”. We manually classified them into eight classes listed in Table 1. Since no standard set of categories exists in the domain, we chose a rather coarse set that seemed appropriate to us. The proportion of each type of questions in the corpus is presented in the table as well; the uneven distribution of the data over classes complicates the classification problem. It’s worth noting that these classes are not necessarily mutually exclusive, but we assume each question belongs to only one of them to simplify the annotation and learning procedures.

3 Features

Given a set of questions, there are three main ways of characterizing them: lexically, syntactically, semantically. Extraction of syntactic and semantic information usually requires language-specific tools. Moreover, many tools, such as parsers, are not well suited for processing questions since they are trained on corpora that don’t contain many questions. For example, the Abney chunker labeled “Name” as a Noun in “Name an ani-

mal...”. The most portable and accessible characterization is lexical.

As a simple version of lexical patterns, we considered the uni-, bi-, and tri-grams in the whole corpus. Using uni- and bi-grams resulted in slightly worse performance (of the decision trees) than using them together with tri-grams. Using uni-grams alone resulted in inferior performance to combining them with other n-grams.

Our only pre-processing step was the replacement of all numbers by *NUM*, which can be done easily for any language. Upon examination, some n-grams looked very useful (e.g. “another word for”), while others looked bland (e.g. “of the”). We experimented with using various sets of n-grams based on their frequency, and found the performance (of the SVM) wasn’t affected: the important n-grams must be those that have medium range frequency.

We decided to proceed with two sets: one with all n-grams but those that occur only once, the other limited to medium range n-grams (excluding top 5 and bottom 15/5/5 for uni/bi/tri-grams). The size of the former was 2835 for English and 3303 for French, and of the latter - 275 for English and 362 for French. We noticed that the number of n-grams for the French set was always higher, probably because French words get inflected more than English ones. Since our feature vectors were quite sparse, we applied Principal Component Analysis (95%) to most of the data: 2835 dimensions for the large English set were reduced to 891; 275 dimensions for the small English set - to 154, and 362 for the small French set - to 187 (we did not apply PCA to the large French set because we saw that it brought no improvements on the large English set, and the performances on the smaller sets were similar between the two languages).

4 Learning

4.1 Baselines & Setup

The lowest baseline of random guessing for an eight-class classification task is 12.5%. However, we devised a baseline by

Table 1: Question classes’ meanings and proportions.

Code	Name	Description	Proportion in the corpus
0	Explanation	E.g. “How does X affect Y?”	2.9%
1	Living entity	People, animals, Gods, organizations, etc.	19.8%
2	Numeric	Cardinality, measurements, time, etc.	25.6%
3	Specialization	Asking for a term given its description	19.8%
4	Location	E.g. “Where can I find X?”	17.6%
5	Persondef	E.g. “Why is X famous?”	16.9%
6	Definition	E.g. “What is an X?”	10.7%
7	Synonym	E.g. “What is another word for X?”	18.5%

classifying the questions using basic rules a human can come up with without thoroughly examining a set of questions, such as “where” implies class “location”. This simplistic classification method gave us $\approx 55\%$ accuracy. This was only performed for the English dataset.

As a third-level baseline, we used the K-Nearest-Neighbours algorithm (using WEKA¹); the optimal K value was determined via cross-validation, and the 9-fold cross-validation accuracy was: **73% for English and 72% for French** using the *small feature set*, and **73% for English and 76% for French** using the *large feature set*. These are well above the 55% baseline even with this simple algorithm. Notably, the accuracy is about the same for the two languages. Using post-PCA data resulted in accuracies lower by several percent.

In our experiments, the validation set size was 200-250 samples (or $\approx 10\%$ of the corpus). We ensured that all classes were represented in the validation set used to set the parameters. Since the dataset is small, 9-fold cross-validation was used for final tests with optimized parameters.

4.2 SVM

Our first experiment was carried out using SVM Torch². It does one vs. all multi-class classification and is designed to handle large datasets. Using a simple validation set, we found that the radial basis function kernel was best.

With the *small feature set*, this method attained an accuracy of **77.3% for English and 79.2% for French**. Again, the accuracy is similar across the two languages. It is also 22% above our hand-crafted baseline and 5-7% above the accuracy achieved by KNN.

With the *big feature set*, it attained an accuracy of **80.6% in English, and 80.8% for French**. This is 26% above our hand-crafted baseline and 5-8% above the accuracy achieved by KNN on this feature set.

Again, the accuracy is similar across the two languages for both sets. Using post-PCA data resulted in accuracies lower by several percent. Note that although the larger feature set gave superior performance, it is ten times bigger than the small feature set, and the improvements are only 2-4%. With a bigger dataset, this difference would likely be even smaller and possibly negligible.

¹<http://www.cs.waikato.ac.nz/ml/weka/>

²<http://www.idiap.ch/bengio/projects/SVM Torch.html>

4.3 Neural Networks

Our next experiment involved using a two-layer sigmoid backpropagation-trained network (available in WEKA) with as many input nodes as there are features (i.e. 154 for English, 187 for French) and as many output nodes as there are classes (i.e. 8). Because of training times, only the post-PCA smaller feature set was used in this experiment.

The optimized parameters were found using a simple validation set: 100 nodes in the only hidden layer, 0.3 learning rate, 0.2 momentum, weight decay. This network setup resulted in **75.6% accuracy for English, and 74.7% for French**. Yet again, the accuracies are approximately the same across the two languages. We are able to achieve good accuracy using rather small networks. Further experimentation with the networks was attractive but darkened by the training times in light of having to optimize parameters.

4.4 Boosting

We experimented with AdaBoost.M1 ([5], available in WEKA) using reweighting (as opposed to resampling). AdaBoost.M1 is the simplest generalization of AdaBoost to the multi-class case: it uses base classifiers that are capable of multi-class classification.

Since even KNN gives accuracy above 70%, we had trouble finding a weak algorithm to use as the base classifier. Decision stumps achieved only 30-40% on either feature set, so boosting couldn’t be based on them. Boosting KNN resulted in a marginal decrease of accuracy.

We also attempted boosting decision trees despite the fact that on their own, they resulted in 79.3% accuracy with the large feature set (for English), and 76.2% with the small feature set. A boosted C4 decision tree resulted in **80.1%** with the large set, and **78.3%** with the small set, bringing these improvements after five iterations. **For French**, using the large feature set resulted in **81.7%** accuracy, and the small one - **77.2%**.

We see that M1 boosting can improve the accuracy of even fairly strong algorithms, but only marginally. Using post-PCA data resulted in slightly lower accuracies, but reduced the training time.

Table 2: Confusion matrix for English (En) and French (Fr).

Class Name	Code	0		1		2		3		4		5		6		7		Accuracy%	
		En	Fr	En	Fr	En	Fr	En	Fr	En	Fr	En	Fr	En	Fr	En	Fr	En	Fr
Exp.	0	22	25	-	2	4	-	19	20	2	3	3	2	4	2	-	-	41	46
Liv.ent.	1	1	-	295	297	3	3	45	52	11	7	12	10	1	1	7	5	79	78
Num.	2	4	2	2	3	435	434	32	30	4	13	-	-	7	2	1	1	90	90
Spec.	3	10	3	35	43	16	21	252	274	27	23	1	1	29	7	5	3	67	73
Loc.	4	2	2	5	9	5	5	36	26	280	291	-	-	4	1	2	-	84	87
P-def.	5	2	1	-	1	-	-	-	2	-	-	30	28	-	-	-	-	94	88
Def.	6	2	3	1	-	3	-	10	17	2	2	-	-	185	181	-	-	91	89
Syn.	7	1	1	6	7	-	-	10	9	-	-	-	-	1	1	17	17	49	49

5 Error Analysis

To analyze the errors, we looked at the confusion matrix of boosting a C4 decision tree based on 9-fold cross validation because it gave us one of the highest accuracies of 80.1% for English and 81.7% for French, based on the large feature set. We're interested in which type of questions were hardest to classify and which got confused most often. We also want to see if there is a relationship between the number of training samples available per class and the error rate for it. The matrix is presented in Table 2. The columns correspond to what the questions were classified as automatically, whereas the first entry in every row represents the true classification.

What stands out most is the amount of questions that were incorrectly classified as class 3 (Specialization). Likely, this can be explained by the fact that specialization questions have the most lexical variability - they are descriptions of various terms, so most of their n-grams are uninformative of the question class. Moreover, they usually use the question word "what", the least informative question word in terms of the answer type.

Second, we notice that class 0 (Explanation) and 7 (Synonym) were classified most poorly. For the former, this can likely be explained by under-representation: 2.9% of the corpus. For the latter, it is not clear. Most other classes were handled fairly well, especially considering that some misclassifications can hardly be considered mistakes. For example, if an instance of class 5 (Persondef) is classified as an Explanation question, this is understandable because questions such as "Why is X famous?" can be easily viewed as Explanation questions as well. This goes back to our previous comment about assigning one category per question being a simplification.

Looking at the confusion matrix for French, we see a very similar picture. The minor differences indicate different uniformity of expressions used in certain types of questions in the two languages, but are overall marginal. It's worth noting that the corresponding matrices for other algorithms displayed similar tendencies.

Finally, seeing that the Specialization class was so problematic, we experimented with taking it out of the dataset. That left us with 1518 questions and 7 classes. We ran the C4 decision tree classifier on this data using the small feature set and got an accuracy of 86%, 10% higher than on the whole dataset. This leads to several ideas. First, perhaps some classes can be discovered using lexical features alone, and only the more com-

plex ones need more sophisticated processing. Second, perhaps the Specialization class isn't a good idea in general: for example, it could be replaced by more specific classes.

We considered another form of evaluation - by comparing the automatically generated classification rules to the manual patterns. The algorithm whose rules were easiest to extract is the decision tree. Comparing them turned out to be difficult for a curious reason: the automatically created patterns relied a lot of the absence of n-grams from a given question, whereas the manually created patterns focused on the presence of things. Although the manual patterns do sometimes make use of absences, and the automatic patterns certainly make use of presences, the difference in focus seems to us big enough that a comparison is not possible.

6 Related Work

The problem of classifying TREC questions using learning algorithms has been pursued since 2002. Radev [4] used Ripper to classify questions into 17 classes and achieved an accuracy of about 70%. The next attempt was made by Li&Roth [2], who used SNoW and a large number of features (lexical, syntactic and semantic), some semantic ones constructed *semi-automatically*, and achieved near-perfect accuracy using a dataset of 5,500 questions.

Zhang&Lee [9] tested a variety of machine learning algorithms using bags of words and bags of n-grams as features. They also tried using smoothed bi-grams after generalizing the corpus syntactically (e.g. replacing clauses by CLAUSE), semantically (e.g. replacing animal names by ANIMAL) and lexically (e.g. replacing all numbers by NUMBER). They found that SVM was the most fitting algorithm. They achieved low 80s accuracy on coarse categories using a dataset of a size similar to ours, and high 80s using datasets of up to almost 6000 questions. They also experimented with adding syntactic information, which improved the accuracy to 90% for the coarse division. For fine-grained categories (almost 50), the top accuracy achieved was about 80%. [9] also includes the learning curves of every algorithm they tested. Although in most cases, the performance keeps improving as the set is augmented, the curves become less steep. It would be interesting to see these curves separately for different question categories.

Suzuki et al. [8] attempted classifying questions in Japanese using their HDAG kernel. The highest accuracy of about 95%

Table 3: Summary of our findings: accuracies in percent (baseline 55%).

KNN				SVM				PCA+Neural Nets				AdaBoost.M1			
En		Fr		En		Fr		En		Fr		En		Fr	
Sm	Lg	Sm	Lg	Sm	Lg	Sm	Lg	Sm	Lg	Sm	Lg	Sm	Lg	Sm	Lg
73	73	72	76	77.3	80.6	79.2	80.6	75.1	-	74.4	-	78.3	80.1	77.2	81.7

was achieved using a combination of bag-of-words, named entity labels and other semantic information.

Solorio et al. [7] focused on language-independent question classification. In addition to bags of words and word prefixes, they used Internet search engines by comparing the number of returned documents for queries such as “president is a person”, “president is a date”. They use SVM to perform the classification, and achieve low 80s results in English, Italian and Spanish using a small dataset of 450 questions.

7 Future Work

Automatic question classification has not been studied extensively. The comparison of various approaches is difficult due to the lack of a standard data set and a standard set of question categories. Establishing those would be beneficial both in terms of allowing comparisons and reducing the manual labour of annotation.

What’s obvious by now is that very accurate automatic classification of factual questions is possible using a combination of lexical, syntactic and semantic information. The challenge lies in developing language-independent methods, which may include using language-independent named entity recognition and automatically acquired semantic information. It would also be interesting to see how automatic classification performs on more complex (longer, non-factual) questions. As mentioned above, allowing multiple categories per question would probably improve the accuracy and provide useful information as well.

One relaxation of the language-independence constraint that would be interesting to explore is the use of part-of-speech labels. It would be especially interesting to employ mixed n-grams that contain both words and POS labels, since manually created patterns are mixed in this sense.

In terms of experiments with boosting, our results with AdaBoost.M1 and decision trees suggests exploring the alternating decision trees algorithm proposed by Freud&Mason in [1]. The AdaBoost.MH [6] approach is attractive because of the easy extension to multi-label tasks, but our preliminary single-label experiments resulted in disappointing baseline-level accuracies.

Another avenue to explore in future work would be the use of large unlabelled corpora which can help get around the lack of linguistic processing; it could be used in conjunction with active learning.

8 Conclusion

In this paper, we described our comparison of various learning algorithms with respect to question classification based on word n-grams. We experimented with two feature sets, one of

about 300 features, the other about 10 times bigger, and found that using the latter brings consistent but small accuracy improvements. The best accuracies attained were 80-82% for English and French, using SVM and boosting decision trees. The performance was consistently similar for English and French. We summarize our findings in Table 3.

Acknowledgments

Guy Lapalme for guidance in the setup of the classification task. Norman Casagrande for discussions of multi-class boosting.

References

- [1] Y. Freund and L. Mason. The alternating decision tree learning algorithm. In *Proceedings of the 16th International Conference on Machine Learning*, pages 124–133.
- [2] X. Li and D. Roth. Learning question classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING 2002)*, 2002.
- [3] L. Plamondon and L. Kosseim. Le web et la question-réponse : transformer une question en réponse. In *Journées francophones de la toile (JFT 2003)*, pages 225–234, Tours, France, jul 2003.
- [4] D. R. Radev, W. Fan, H. Qi, H. Wu, and A. Grewal. Probabilistic question answering from the web. In *Proceedings of the 11th International World Wide Web Conference*, 2002.
- [5] R. E. Schapire. A brief introduction to boosting. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, 1999.
- [6] R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.
- [7] T. Solorio, M. Pérez-Coutiño, M. M. y Gómez, L. Vilaseñor-Pineda, and A. López-López. A language independent method for question classification. In *Proceedings of the 20th International Conference on Computational Linguistics, COLING-04, volume II*, pages 1374–1380, 2004.
- [8] J. Suzuki, H. Taira, Y. Sasaki, and E. Maeda. Question classification using hdag kernel. In *Workshop on Multilingual Summarization and Question Answering 2003, (post-conference workshop in conjunction with ACL-2003)*, pages 61–68, 2003.
- [9] D. Zhang and W. S. Lee. Question classification using support vector machines. In *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 Press.