A Tree Kernel approach to Question and Answer Classification in Question Answering Systems

Alessandro Moschitti and Roberto Basili

University of Rome Tor Vergata,
Department of Computer Science, Systems and Production,
00133 Roma (Italy),
{moschitti, basili}@info.uniroma2.it

Abstract

A critical step in Question Answering design is the definition of the models for question focus identification and answer extraction. In case of factoid questions, we can use a question classifier (trained according to a target taxonomy) and a named entity recognizer. Unfortunately, this latter cannot be applied to generate answers related to non-factoid questions. In this paper, we tackle such problem by designing classifiers of *non-factoid answers*. As the feature design for this learning task is very complex, we take advantage of tree kernels to generate large feature set from the syntactic parse trees of passages relevant to the target question. Such kernels encode syntactic and lexical information in Support Vector Machines which can decide if a sentence focuses on a target taxonomy subject. The experiments with SVMs on the TREC 10 dataset show that our approach is an interesting future research.

1. Introduction

Among other Information Retrieval paradigms, Question Answering (QA) (Maybury, 2004) has been shown to be very effective to find the desired information. The possibility of expressing the query in natural language and the pointwise answer to such queries make QA systems very interesting for any kind of textual corpus data analysis.

From a modeling point of view, the main difference between QA and document retrieval systems is the chance (of the former) to classify the query (i.e. the question) type. This simple information allows us to raffinate the traditional search engines by designing models that look for a specific information type in a more specific text fragment. As a result, the retrieval accuracy increases and the returned information is contained in much smaller text snippets than in whole documents.

Traditional approaches to question classification rely on the manual extraction of semantic and syntactic features (Li and Roth, 2002) which aim to capture the properties of each different class of a target question taxonomy. Such features may be used to manually handcraft classification rules or used in machine learning algorithms to automatically derive a classification model.

It should be noted that different taxonomies may require different features and different rules, consequently, the machine learning approach seems better suited to study the question classification problem on different application domains since it is easier to manually re-organize questions in a different taxonomy rather than manually re-design the classification rules for a new taxonomy. Since many experimental taxonomies can be made available with their related training data, the major problems relate to the design of effective features to learn the target question categories.

Several researches (Maybury, 2004) have shown that syntactic and lexical information seems essential to achieve an accurate classification. A viable alternative to the manual feature design is thus the tree kernel approach (Collins and Duffy, 2002). Intuitively, tree kernel functions allow the learning machine to use as feature vector components all the subtrees extracted from the syntactic-parse trees of the training set. For example, Figure 1 shows a small syntactic parse tree along with all features extracted from it. The total number of the tree fragments extracted from the training data can be very large, but the kernel function limits such complexity. It avoids to generate all features by carrying out the scalar product between two vectors only implicitly. As the scalar product is the only view of the data that several machine learning models have, the entire learning can be carried out without building the whole huge feature space.

Tree Kernels have successfully been applied to derive feature space automatically for several tasks, e.g. parse-tree re-ranking (Collins and Duffy, 2002), relation extraction (Culotta and Sorensen, 2004), semantic role labeling (Moschitti, 2004) and question classification (Zhang and Lee, 2003). In particular, in (Zhang and Lee, 2003) a question classifier based on tree kernels has been shown to achieve a high accuracy; greater than the simple *bag-of-words* approach.

As tree kernels are very useful to derive syntactic/lexical features automatically, we may use them for another more difficult task, i.e. the answer classification. Once, we know the category of a question, we can search the related answer in the target document collection by exploiting the type defined by the category information. In case of factoid questions, this task is simple as their type can indicate just the Named Entity class of the complex nominal that should be provided as the answer. Consequently, the application of a Named Entity recognizer on the relevant passages provides an effective answer selection strategy. On the contrary, when the question asks for a description, we need to recognize the passages in the available documents that contain descriptive or defining information. Such task appears very complex from a machine learning point of view as it is very difficult to design features suitable for it.

Intuitively, the role of syntax is critical to the detection of

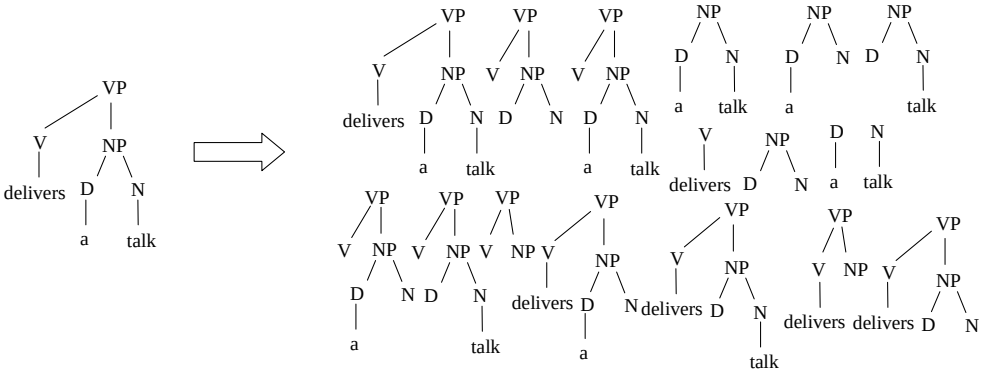

Figure 1: A syntactic parse tree with its representing features produced by the tree kernel function.

the descriptive passages, although the manual design of effective syntactic features seems to be very hard. Therefore, we may rely on the use of tree kernels to generate many syntactic fragments and study their impact on the classification accuracy of passages.

In the remainder of this paper Section 2. introduces our question/answer classification models whereas Section 3. report the experiments on such models. Finally, Section 4. summarizes the conclusions.

2. Tree Kernel Model for Question/Answer Classification

Tree kernels represent parse trees in a subtree space that they implicitly generate. For example, Figure 1 shows a small subtree (on the left) and its representation based on tree fragments (on the right). Such representation is then mapped in a vector space where each component is associated with a different fragment. The overall space is the union of the fragments of any parse tree of the adopted corpus.

Many learning algorithms like kernel-based machines just use a similarity measure, e.g. the scalar product, between the examples to carry out the training process, consequently, what is really important is to define the similarity measure or kernel function.

Formally, let us define a subtree space $\mathcal{F} = \{f_1, f_2, \ldots, f_{|\mathcal{F}|}\}$ and the indicator function $I_i(n)$ such that it is equal to 1 if the target f_i is rooted at node n and equal to 0 otherwise. We can define a tree-kernel function $K_T(t_1, t_2)$ over two trees t_1 and t_2 equal to $\sum_{n_1 \in N_{t_1}} \sum_{n_2 \in N_{t_2}} \Delta(n_1, n_2)$, where N_{t_1} and N_{t_2} are the sets of the t_1's and t_2's nodes, respectively. In turn $\Delta(n_1, n_2) = \sum_{i=1}^{|\mathcal{F}|} \lambda^{l(f_i)} I_i(n_1) I_i(n_2)$, where $0 \leq \lambda \leq 1$ and $l(f_i)$ is the number of levels of the subtree f_i. Thus $\lambda^{l(f_i)}$ assigns a lower weight to larger fragments. When $\lambda = 1$, Δ is equal to the number of common fragments rooted at nodes n_1 and n_2. To evaluate Δ efficiently (i.e. in $O(|N_{t_1}| \times |N_{t_2}|)$), we use the algorithm described in (Collins and Duffy, 2002).

The kernel function can be used to classify both questions and answers as we can generate parse trees for both. For example, Figure 2 shows the syntactic parse of the question `What does HTML stand for?`, which will be used directly in the kernel function.

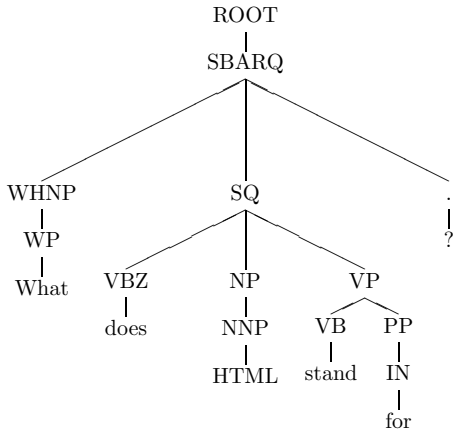

Figure 2: A syntactic parse tree with its representing features produced by the tree kernel function.

Similarly, by parsing the sentences used in the correct answer to a certain type of question, we can generate a collection of trees to train the answer type classifier.

3. Experiments

The aim of these experiments is to show that tree kernels can be used to learn the classification of questions as well as answers with respect to a given taxonomy.

For question classification, we used the data set available at `http://l2r.cs.uiuc.edu/~cogcomp/Data/QA/QC/`. This contains 5,500 training and 500 test questions from the TREC 10 QA competition. We used a subpart of the question taxonomy available at `l2r.cs.uiuc.edu/~cogcomp/Data/QA/QC/definition.html`, i.e. *definition*, *Description*, *Entity*, *Human*, *Location*, *Manner*, *Numeric* and *Organization*. Our results can be compared with (Zhang and Lee, 2003; Li and Roth, 2005) in some extents but we used a mixed coarse grained taxonomy, e.g. *definition* and *description* subclasses *Description*, that constitutes a more difficult classification task.

Type	a	b	c	d	Precision	Recall	Acc.	F1
definition	62	10	3	611	86,11	95,38	98,10	90,51
description	41	10	29	606	80,39	58,57	94,31	67,77
entity	87	56	26	517	60,84	76,99	88,05	67,97
human	102	24	16	544	80,95	86,44	94,17	83,61
location	101	15	11	559	87,07	90,18	96,21	88,60
manner	43	1	13	629	97,73	76,79	97,96	86,00
numeric	116	6	13	551	95,08	89,92	97,23	92,43
organization	11	1	12	662	91,67	47,83	98,10	62,86

Table 1: Results of individual question classifiers.

	a	b	c	d	Precision	Recall	Acc.	F1
Multiclassifier	563	123	123	0	0,8207	0,8207	0,8207	0,8207

Table 2: Question multi-classifier accuracy.

Table 1 reports the performance of the different individual classifiers on the test set (obtained by applying the ONE-vs-ALL scheme (Rifkin and Klautau, 2004)). Column 1 shows the type of the question, columns from 2 to 5 report the number of (a) *correct*, (b) *incorrect*, (c) *missed* and (d) *correctly not classified* questions, respectively and columns from 6 to 9 illustrates the Precision, Recall and F1 measure, respectively. We note that *entity* and *organization* show the lower F1s as they tend to confuse each other.
Table 2 shows the accuracy of the overall SVM multi-classifier obtained by choosing the question class associated with the highest score among of the set of binary SVMs. The resulting accuracy is enough satisfactory.
For answer classification, we applied the following steps:

1. Selection of 30 questions of type definition, description or manner from those of the Trec 10 competition (`http://trec.nist.gov/pubs/trec9/t9_proceedings.html`).
2. Querying our question answering system with the above questions to retrieve the most relevant paragraphs.
3. Labeling of the 1,000 paragraphs obtained in the previous step according to description and other categories. If a paragraph does not contain answer that can be categorized in one of the tree categories, we label it as *noclass*.
4. Application of the Charniak parser (Charniak, 2000) to the above paragraphs to convert them in syntactic parse trees.
5. Learning the Support Vector Machine Classifiers using the tree kernel on 900 parse trees and testing on the remaining 100 trees.

Table 3 reports the classification performance of passages of the definition, description, manner and noclass classifiers. The F1 measures are quite low but we should consider that the three answer classes are difficult to separate. To show this, we run an experiment in which we built a

Type	a	b	c	d	Precision	Recall	Acc.	F1
definition	3	0	10	87	100,00	23,08	90,00	37,50
description	6	0	9	85	100,00	40,00	91,00	57,14
manner	2	0	8	90	100,00	20,00	92,00	33,33
noClass	59	21	3	17	73,75	95,16	76,00	83,10

Table 3: F1 of the answer classifiers for definition, description and manner categories.

Type	a	b	c	d	Precision	Recall	Acc.	F1
def.-descr.-man.	21	8	17	54	72,41	55,26	75,00	62,69

Table 4: F1 of the answer classifier for the grouped category (definition, description and manner category) vs. the not-a-class classifier.

classifier that only decides if a question belongs to one of the three categories or is a noclass. Table 4 reports an F1 of 62.69% for such classifier. This is a quite good result as such classifier can help a traditional answer extraction module to select relevant passages. Moreover, as described in (Moschitti, 2004), we can combine tree kernels with manual features to boost the *Answer Classification* accuracy.

4. Discussion and Conclusions

The preliminary results illustrated in this paper suggest three important consideartions:

First, tree kernels, by automatically generating several syntactic features, alleviate the work of the feature designer. The effort to find the relevant and irrelevant features can be avoided as it is also carried out by SVMs automatically.

Second, the good accuracy of the kernel approach in question classification shows the benefit of using tree kernels. It should be noted that there are several kind of tree kernel functions, e.g. (Moschitti et al., 2005; Basili and Moschitti, 2005; Moschitti, 2006), but the subset tree kernel (also called all subtrees) is the more adequate for constituent parsing. Indeed, to generate syntactic features, it follows the prior knowledge given by a grammar, thus it captures a more precise information. We could draw a parallel with the bag of word kernel and the string kernel. It has been proven that the former is slightly more accurate as the features are formed with the prior knowledge that words should be more meaningful than other character sequences. Of course, the large feature set produced by all subtree kernel could be highly reduced by designing some specific features manually. This constitutes an exciting and complex future research.

Finally, the answer classification results suggest that it is possible to extract syntax/semantic-based features to learn answer taxonomies. The study of different kernel types on different sources of information, e.g. constituent and dependency parsing, is an interesting short term research. Also the integration of a more explicit source of semantic cues such as semantic roles is a promising research line that we would like to pursue.

5. References

Roberto Basili and Alessandro Moschitti. 2005. *Automatic Text Categorization: from Information Retrieval to Support Vector Learning*. Aracne.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the 1st Meeting of the North American Chapter of the ACL*, pages 132–139.

Michael Collins and Nigel Duffy. 2002. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *ACL02*.

Aron Culotta and Jeffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 423–429, Barcelona, Spain, July.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *Proceedings of COLING'02*.

X. Li and D. Roth. 2005. Learning question classifiers: The role of semantic information. *Journal of Natural Language Engineering*, 11(4).

Mark T. Maybury. 2004. *New Directions in Question Answering*. AAAI Press.

Alessandro Moschitti, Bonaventura Coppola, Daniele Pighin, and Roberto Basili. 2005. Engineering of syntactic features for shallow semantic parsing. In *of the ACL05 Workshop on Feature Engineering for Machine Learning in Natural Language Processing*, USA.

Alessandro Moschitti. 2004. A study on convolution kernel for shallow semantic parsing. In *proceedings of the 42^{th} Conference on Association for Computational Linguistic (ACL-2004)*, Barcelona, Spain.

Alessandro Moschitti. 2006. Making tree kernels practical for natural language learning. In *Proceedings of EACL'06*, Trento, Italy.

R. Rifkin and A. Klautau. Vol. 5, 101-141, 2004. In defense of one-vs-all classification. *Journal of Machine Learning Research*.

Dell Zhang and Wee Sun Lee. 2003. 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. ACM Press.