Exploiting Syntactic and Shallow Semantic Kernels for Question/Answer Classification

Alessandro Moschitti
University of Trento
38050 Povo di Trento
Italy
moschitti@dit.unitn.it

Silvia Quarteroni
The University of York
York YO10 5DD
United Kingdom
silvia@cs.york.ac.uk

Roberto Basili
"Tor Vergata" University
Via del Politecnico 1
00133 Rome, Italy
basili@info.uniroma2.it

Suresh Manandhar
The University of York
York YO10 5DD
United Kingdom
suresh@cs.york.ac.uk

Abstract

We study the impact of syntactic and shallow semantic information in automatic classification of questions and answers and answer re-ranking. We define (a) new tree structures based on shallow semantics encoded in Predicate Argument Structures (PASs) and (b) new kernel functions to exploit the representational power of such structures with Support Vector Machines. Our experiments suggest that syntactic information helps tasks such as question/answer classification and that shallow semantics gives remarkable contribution when a reliable set of PASs can be extracted, e.g. from answers.

1 Introduction

Question answering (QA) is as a form of information retrieval where one or more answers are returned to a question in natural language in the form of sentences or phrases. The typical QA system architecture consists of three phases: question processing, document retrieval and answer extraction (Kwok et al., 2001).

Question processing is often centered on question classification, which selects one of k expected answer classes. Most accurate models apply supervised machine learning techniques, e.g. SNoW (Li and Roth, 2005), where questions are encoded using various lexical, syntactic and semantic features. The retrieval and answer extraction phases consist in retrieving relevant documents (Collins-Thompson et al., 2004) and selecting candidate answer passages from them. A further answer re-ranking phase is optionally applied. Here, too, the syntactic structure of a sentence appears to provide more useful information than a bag of words (Chen et al., 2006), although the correct way to exploit it is still an open problem.

An effective way to integrate syntactic structures in machine learning algorithms is the use of tree kernel (TK) functions (Collins and Duffy, 2002), which have been successfully applied to question classification (Zhang and Lee, 2003; Moschitti, 2006) and other tasks, e.g. relation extraction (Zelenko et al., 2003; Moschitti, 2006). In more complex tasks such as computing the relatedness between questions and answers in answer re-ranking, to our knowledge no study uses kernel functions to encode syntactic information. Moreover, the study of shallow semantic information such as predicate argument structures annotated in the PropBank (PB) project (Kingsbury and Palmer, 2002) (www.cis.upenn.edu/~ace) is a promising research direction. We argue that semantic structures can be used to characterize the relation between a question and a candidate answer.

In this paper, we extensively study new structural representations, encoding parse trees, bag-of-words, POS tags and predicate argument structures (PASs) for question classification and answer re-ranking. We define new tree representations for both simple and nested PASs, i.e. PASs whose arguments are other predicates (Section 2). Moreover, we define new kernel functions to exploit PASs, which we automatically derive with our SRL system (Moschitti et al., 2005) (Section 3).

Our experiments using SVMs and the above ker-

Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, pages 776–783,
Prague, Czech Republic, June 2007. ©2007 Association for Computational Linguistics

nels and data (Section 4) shows the following: (a) our approach reaches state-of-the-art accuracy on question classification. (b) PB predicative structures are not effective for question classification but show promising results for answer classification on a corpus of answers to TREC-QA 2001 description questions. We created such dataset by using YourQA (Quarteroni and Manandhar, 2006), our basic Web-based QA system[1]. (c) The answer classifier increases the ranking accuracy of our QA system by about 25%.

Our results show that PAS and syntactic parsing are promising methods to address tasks affected by data sparseness like question/answer categorization.

2 Encoding Shallow Semantic Structures

Traditionally, information retrieval techniques are based on the *bag-of-words* (BOW) approach augmented by language modeling (Allan et al., 2002). When the task requires the use of more complex semantics, the above approaches are often inadequate to perform fine-level textual analysis.

An improvement on BOW is given by the use of syntactic parse trees, e.g. for question classification (Zhang and Lee, 2003), but these, too are inadequate when dealing with definitional answers expressed by long and articulated sentences or even paragraphs. On the contrary, shallow semantic representations, bearing a more "compact" information, could prevent the sparseness of deep structural approaches and the weakness of BOW models.

Initiatives such as PropBank (PB) (Kingsbury and Palmer, 2002) have made possible the design of accurate automatic Semantic Role Labeling (SRL) systems (Carreras and Màrquez, 2005). Attempting an application of SRL to QA hence seems natural, as pinpointing the answer to a question relies on a deep understanding of the semantics of both.

Let us consider the PB annotation: [$_{ARG1}$ Antigens] were [$_{AM-TMP}$ originally] [$_{rel}$ defined] [$_{ARG2}$ as non-self molecules].

Such annotation can be used to design a shallow semantic representation that can be matched against other semantically similar sentences, e.g. [$_{ARG0}$ Researchers] [$_{rel}$ describe] [$_{ARG1}$ antigens] [$_{ARG2}$ as foreign molecules] [$_{ARGM-LOC}$ in the body].

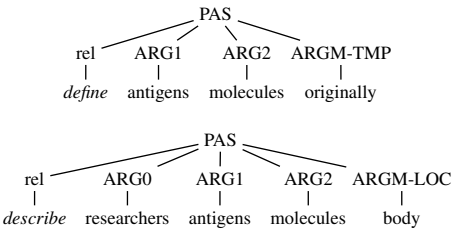

Figure 1: Compact predicate argument structures of two different sentences.

For this purpose, we can represent the above annotated sentences using the tree structures described in Figure 1. In this compact representation, hereafter Predicate-Argument Structures (PAS), arguments are replaced with their most important word – often referred to as the semantic head. This reduces data sparseness with respect to a typical BOW representation.

However, sentences rarely contain a single predicate; it happens more generally that propositions contain one or more subordinate clauses. For instance let us consider a slight modification of the first sentence: "Antigens were originally defined as non-self molecules *which bound specifically to antibodies*[2]." Here, the main predicate is "defined", followed by a subordinate predicate "bound". Our SRL system outputs the following *two* annotations:

(1) [$_{ARG1}$ Antigens] were [$_{ARGM-TMP}$ originally] [$_{rel}$ defined] [$_{ARG2}$ as non-self molecules which bound specifically to antibodies].

(2) Antigens were originally defined as [$_{ARG1}$ non-self molecules] [$_{R-A1}$ which] [$_{rel}$ bound] [$_{ARGM-MNR}$ specifically] [$_{ARG2}$ to antibodies].

giving the PASs in Figure 2.(a) resp. 2.(b).

As visible in Figure 2.(a), when an argument node corresponds to an entire subordinate clause, we label its leaf with PAS, e.g. the leaf of ARG2. Such PAS node is actually the root of the subordinate clause in Figure 2.(b). Taken as standalone, such PASs do not express the whole meaning of the sentence; it is more accurate to define a single structure encoding the dependency between the two predicates as in

[1]Demo at: http://cs.york.ac.uk/aig/aqua.

[2]This is an actual answer to "What are antibodies?" from our question answering system, YourQA.

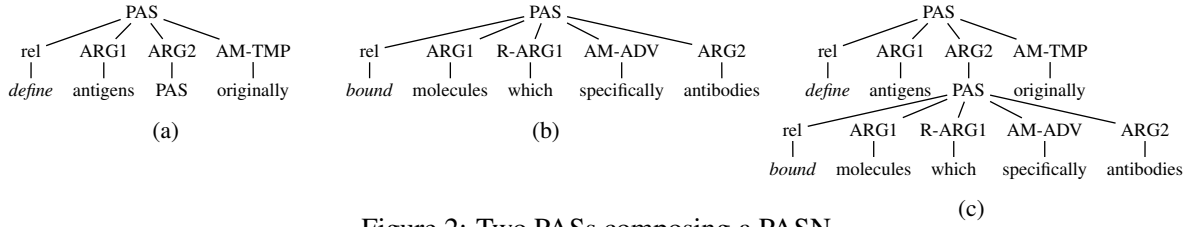

Figure 2: Two PASs composing a PASN

Figure 2.(c). We refer to nested PASs as PASNs.

It is worth to note that semantically equivalent sentences syntactically expressed in different ways share the same PB arguments and the same PASs, whereas semantically different sentences result in different PASs. For example, the sentence: "Antigens were originally defined as *antibodies* which bound specifically to *non-self molecules*", uses the same words as (2) but has different meaning. Its PB annotation:

(3) `Antigens were originally defined as [`$_{ARG1}$ `antibodies] [`$_{R-A1}$ `which] [`$_{rel}$ `bound] [`$_{ARGM-MNR}$ `specifically] [`$_{ARG2}$ `to non-self molecules]`,

clearly differs from (2), as ARG2 is now *non-self molecules*; consequently, the PASs are also different.

Once we have assumed that parse trees and PASs can improve on the simple BOW representation, we face the problem of representing tree structures in learning machines. Section 3 introduces a viable approach based on tree kernels.

3 Syntactic and Semantic Kernels for Text

As mentioned above, encoding syntactic/semantic information represented by means of tree structures in the learning algorithm is problematic. A first solution is to use all its possible substructures as features. Given the combinatorial explosion of considering subparts, the resulting feature space is usually very large. A tree kernel (TK) function which computes the number of common subtrees between two syntactic parse trees has been given in (Collins and Duffy, 2002). Unfortunately, such subtrees are subject to the constraint that their nodes are taken with all or none of the children they have in the original tree. This makes the TK function not well suited for the PAS trees defined above. For instance, although the two PASs of Figure 1 share most of the subtrees rooted in the PAS node, Collins and Duffy's kernel would compute no match.

In the next section we describe a new kernel derived from the above tree kernel, able to evaluate the meaningful substructures for PAS trees. Moreover, as a single PAS may not be sufficient for text representation, we propose a new kernel that combines the contributions of different PASs.

3.1 Tree kernels

Given two trees T_1 and T_2, let $\{f_1, f_2, ..\} = \mathcal{F}$ be the set of substructures (fragments) and $I_i(n)$ be equal to 1 if f_i is rooted at node n, 0 otherwise. Collins and Duffy's kernel is defined as

$$TK(T_1, T_2) = \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \Delta(n_1, n_2), \quad (1)$$

where N_{T_1} and N_{T_2} are the sets of nodes in T_1 and T_2, respectively and $\Delta(n_1, n_2) = \sum_{i=1}^{|\mathcal{F}|} I_i(n_1) I_i(n_2)$. The latter is equal to the number of common fragments rooted in nodes n_1 and n_2. Δ can be computed as follows:

(1) if the productions (i.e. the nodes with their direct children) at n_1 and n_2 are different then $\Delta(n_1, n_2) = 0$;

(2) if the productions at n_1 and n_2 are the same, and n_1 and n_2 only have leaf children (i.e. they are pre-terminal symbols) then $\Delta(n_1, n_2) = 1$;

(3) if the productions at n_1 and n_2 are the same, and n_1 and n_2 are not pre-terminals then $\Delta(n_1, n_2) = \prod_{j=1}^{nc(n_1)} (1 + \Delta(c_{n_1}^j, c_{n_2}^j))$, where $nc(n_1)$ is the number of children of n_1 and c_n^j is the j-th child of n.

Such tree kernel can be normalized and a λ factor can be added to reduce the weight of large structures (refer to (Collins and Duffy, 2002) for a complete description). The critical aspect of steps (1), (2) and (3) is that the productions of two evaluated nodes have to be identical to allow the match of further descendants. This means that common substructures cannot be composed by a node with only some of its

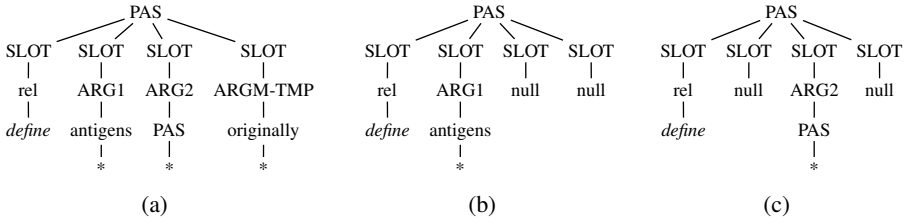

Figure 3: A PAS with some of its fragments.

children as an effective PAS representation would require. We solve this problem by designing the Shallow Semantic Tree Kernel (SSTK) which allows to match portions of a PAS.

3.2 The Shallow Semantic Tree Kernel (SSTK)

The SSTK is based on two ideas: first, we change the PAS, as shown in Figure 3.(a) by adding *SLOT* nodes. These accommodate argument labels in a specific order, i.e. we provide a fixed number of slots, possibly filled with *null* arguments, that encode all possible predicate arguments. For simplicity, the figure shows a structure of just 4 arguments, but more can be added to accommodate the maximum number of arguments a predicate can have. Leaf nodes are filled with the wildcard character $\star$ but they may alternatively accommodate additional information.

The slot nodes are used in such a way that the adopted TK function can generate fragments containing one or more children like for example those shown in frames (b) and (c) of Figure 3. As previously pointed out, if the arguments were directly attached to the root node, the kernel function would only generate the structure with all children (or the structure with no children, i.e. empty).

Second, as the original tree kernel would generate many matches with slots filled with the null label, we have set a new step 0:

(0) if n_1 (or n_2) is a pre-terminal node and its child label is *null*, $\Delta(n_1, n_2) = 0$;

and subtract one unit to $\Delta(n_1, n_2)$, in step 3:

(3) $\Delta(n_1, n_2) = \prod_{j=1}^{nc(n_1)}(1 + \Delta(c_{n_1}^j, c_{n_2}^j)) - 1$,

The above changes generate a new Δ which, when substituted (in place of the original Δ) in Eq. 1, gives the new Shallow Semantic Tree Kernel. To show that SSTK is effective in counting the number of relations shared by two PASs, we propose the following:

Proposition 1 *The new Δ function applied to the modified PAS counts the number of all possible k-ary relations derivable from a set of k arguments, i.e. $\sum_{i=1}^{k}\binom{k}{i}$ relations of arity from 1 to k (the predicate being considered as a special argument).*

Proof We observe that a kernel applied to a tree and itself computes all its substructures, thus if we evaluate SSTK between a PAS and itself we must obtain the number of generated k-ary relations. We prove by induction the above claim.

For the base case ($k = 0$): we use a PAS with no arguments, i.e. all its slots are filled with null labels. Let r be the PAS root; since r is not a pre-terminal, step 3 is selected and Δ is recursively applied to all r's children, i.e. the slot nodes. For the latter, step 0 assigns $\Delta(c_r^j, c_r^j) = 0$. As a result, $\Delta(r, r) = \prod_{j=1}^{nc(r)}(1 + 0) - 1 = 0$ and the base case holds.

For the general case, r is the root of a PAS with $k+1$ arguments. $\Delta(r, r) = \prod_{j=1}^{nc(r)}(1 + \Delta(c_r^j, c_r^j)) - 1$ $=\prod_{j=1}^{k}(1+\Delta(c_r^j, c_r^j))\times(1+\Delta(c_r^{k+1}, c_r^{k+1}))-1$. For k arguments, we assume by induction that $\prod_{j=1}^{k}(1+\Delta(c_r^j, c_r^j)) - 1 = \sum_{i=1}^{k}\binom{k}{i}$, i.e. the number of k-ary relations. Moreover, $(1 + \Delta(c_r^{k+1}, c_r^{k+1})) = 2$, thus $\Delta(r, r) = \sum_{i=1}^{k}\binom{k}{i} \times 2 = 2^k \times 2 = 2^{k+1} = \sum_{i=1}^{k+1}\binom{k+1}{i}$, i.e. all the relations until arity $k + 1$ □

TK functions can be applied to sentence parse trees, therefore their usefulness for text processing applications, e.g. question classification, is evident. On the contrary, the SSTK applied to one PAS extracted from a text fragment may not be meaningful since its representation needs to take into account all the PASs that it contains. We address such problem

by defining a kernel on *multiple* PASs.

Let P_t and $P_{t'}$ be the sets of PASs extracted from the text fragment t and t'. We define:

$$K_{\texttt{all}}(P_t, P_{t'}) = \sum_{p \in P_t} \sum_{p' \in P_{t'}} SSTK(p, p'), \quad (2)$$

While during the experiments (Sect. 4) the $K_{\texttt{all}}$ kernel is used to handle predicate argument structures, TK (Eq. 1) is used to process parse trees and the linear kernel to handle POS and BOW features.

4 Experiments

The purpose of our experiments is to study the impact of the new representations introduced earlier for QA tasks. In particular, we focus on question classification and answer re-ranking for Web-based QA systems.

In the question classification task, we extend previous studies, e.g. (Zhang and Lee, 2003; Moschitti, 2006), by testing a set of previously designed kernels and their combination with our new Shallow Semantic Tree Kernel. In the answer re-ranking task, we approach the problem of detecting description answers, among the most complex in the literature (Cui et al., 2005; Kazawa et al., 2001).

The representations that we adopt are: bag-of-words (BOW), bag-of-POS tags (POS), parse tree (PT), predicate argument structure (PAS) and nested PAS (PASN). BOW and POS are processed by means of a linear kernel, PT is processed with TK, PAS and PASN are processed by SSTK. We implemented the proposed kernels in the SVM-light-TK software available at `ai-nlp.info.uniroma2.it/moschitti/` which encodes tree kernel functions in SVM-light (Joachims, 1999).

4.1 Question classification

As a first experiment, we focus on question classification, for which benchmarks and baseline results are available (Zhang and Lee, 2003; Li and Roth, 2005). We design a question multi-classifier by combining n binary SVMs[3] according to the ONE-vs-ALL scheme, where the final output class is the one associated with the most probable prediction. The PASs were automatically derived by our SRL system which achieves a 76% F1-measure (Moschitti et al., 2005).

As benchmark data, we use the question training and test set available at: `l2r.cs.uiuc.edu/~cogcomp/Data/QA/QC/`, where the test set are the 500 TREC 2001 test questions (Voorhees, 2001). We refer to this split as UIUC. The performance of the multi-classifier and the individual binary classifiers is measured with accuracy resp. F1-measure. To collect statistically significant information, we run 10-fold cross validation on the 6,000 questions.

Features	Accuracy (UIUC)	Accuracy (c.v.)
PT	90.4	84.8±1.2
BOW	90.6	84.7±1.2
PAS	34.2	43.0±1.9
POS	26.4	32.4±2.1
PT+BOW	**91.8**	**86.1±1.1**
PT+BOW+POS	91.8	84.7±1.5
PAS+BOW	90.0	82.1±1.3
PAS+BOW+POS	88.8	81.0±1.5

Table 1: Accuracy of the question classifier with different feature combinations

Question classification results Table 1 shows the accuracy of different question representations on the UIUC split (Column 1) and the average accuracy ± the corresponding confidence limit (at 90% significance) on the cross validation splits (Column 2).(i) The TK on PT and the linear kernel on BOW produce a very high result, i.e. about 90.5%. This is higher than the best outcome derived in (Zhang and Lee, 2003), i.e. 90%, obtained with a kernel combining BOW and PT on the same data. Combined with PT, BOW reaches 91.8%, very close to the 92.5% accuracy reached in (Li and Roth, 2005) using complex semantic information from external resources.
(ii) The PAS feature provides no improvement. This is mainly because at least half of the training and test questions only contain the predicate "to be", for which a PAS cannot be derived by a PB-based shallow semantic parser.
(iii) The 10-fold cross-validation experiments confirm the trends observed in the UIUC split. The best model (according to statistical significance) is PT+BOW, achieving an 86.1% average accuracy[4].

[3] We adopted the default regularization parameter (i.e., the average of $1/||\vec{x}||$) and tried a few cost-factor values to adjust the rate between Precision and Recall on the development set.

[4] This value is lower than the UIUC split one as the UIUC test set is not consistent with the training set (it contains the

4.2 Answer classification

Question classification does not allow to fully exploit the PAS potential since questions tend to be short and with few verbal predicates (i.e. the only ones that our SRL system can extract). A different scenario is answer classification, i.e. deciding if a passage/sentence correctly answers a question. Here, the semantics to be generated by the classifier are not constrained to a small taxonomy and answer length may make the PT-based representation too sparse.

We learn answer classification with a binary SVM which determines if an answer is correct for the target question: here, the classification instances are ⟨question, answer⟩ pairs. Each pair component can be encoded with PT, BOW, PAS and PASN representations (processed by previous kernels).

As test data, we collected the 138 TREC 2001 test questions labeled as "description" and for each, we obtained a list of answer paragraphs extracted from Web documents using YourQA. Each paragraph sentence was manually evaluated based on whether it contained an answer to the corresponding question. Moreover, to simplify the classification problem, we isolated for each paragraph the sentence which obtained the maximal judgment (in case more than one sentence in the paragraph had the same judgment, we chose the first one). We collected a corpus containing 1309 sentences, 416 of which – labeled "+1" – answered the question either concisely or with noise; the rest – labeled "-1"– were either irrelevant to the question or contained hints relating to the question but could not be judged as valid answers[5].

Answer classification results To test the impact of our models on answer classification, we ran 5-fold cross-validation, with the constraint that two pairs $\langle q, a_1 \rangle$ and $\langle q, a_2 \rangle$ associated with the same question q could not be split between training and testing. Hence, each reported value is the average over 5 different outcomes. The standard deviations ranged

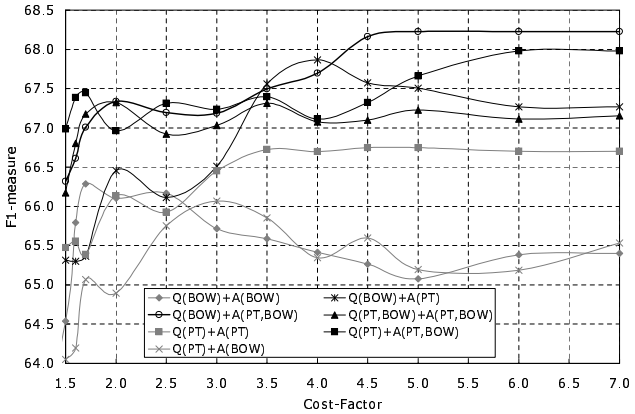

Figure 4: Impact of the BOW and PT features on answer classification

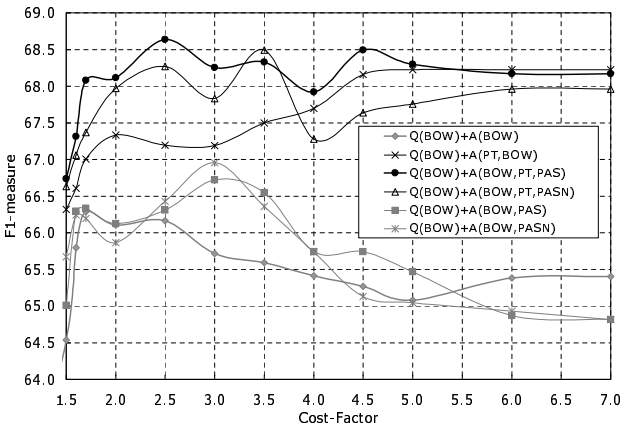

Figure 5: Impact of the PAS and PASN features combined with the BOW and PT features on answer classification

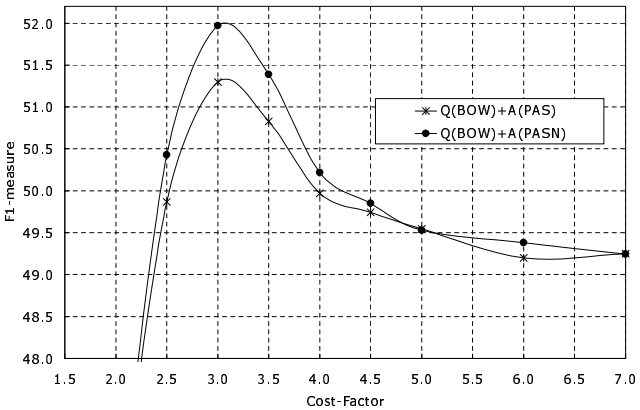

Figure 6: Comparison between PAS and PASN when used as standalone features for the answer on answer classification

TREC 2001 questions) and includes a larger percentage of easily classified question types, e.g. the numeric (22.6%) and description classes (27.6%) whose percentage in training is 16.4% resp. 16.2%.

[5]For instance, given the question "What are invertebrates?", the sentence "At least 99% of all animal species are invertebrates, comprising ..." was labeled "-1" , while "Invertebrates are animals without backbones." was labeled "+1".

approximately between 2.5 and 5. The experiments were organized as follows:

First, we examined the contributions of BOW and PT representations as they proved very important for question classification. Figure 4 reports the plot of the F1-measure of answer classifiers trained with all combinations of the above models according to different values of the cost-factor parameter, adjusting the rate between Precision and Recall. We see here that the most accurate classifiers are the ones using both the answer's BOW and PT feature and either the question's PT or BOW feature (i.e. Q(BOW) + A(PT,BOW) resp. Q(PT) + A(PT,BOW) combinations). When PT is used for the answer the simple BOW model is outperformed by 2 to 3 points. Hence, we infer that both the answer's PT and BOW features are very useful in the classification task. However, PT does not seem to provide additional information to BOW when used for question representation. This can be explained by considering that answer classification (restricted to description questions) does not require question type classification since its main purpose is to detect question/answer relations. In this scenario, the question's syntactic structure does not seem to provide much more information than BOW.

Secondly, we evaluated the impact of the newly defined PAS and PASN features combined with the best performing previous model, i.e. Q(BOW) + A(PT,BOW). Figure 5 illustrates the F1-measure plots again according to the cost-factor parameter. We observe here that model Q(BOW) + A(PT,BOW,PAS) greatly outperforms model Q(BOW) + A(PT,BOW), proving that the PAS feature is very useful for answer classification, i.e. the improvement is about 2 to 3 points while the difference with the BOW model, i.e. Q(BOW) + A(BOW), exceeds 3 points. The Q(BOW) + A(PT,BOW,PASN) model is not more effective than Q(BOW) + A(PT,BOW,PAS). This suggests either that PAS is more effective than PASN or that when the PT information is added, the PASN contribution fades out.

To further investigate the previous issue, we finally compared the contribution of the PAS and PASN when combined with the question's BOW feature alone, i.e. no PT is used. The results, reported in Figure 6, show that this time PASN performs better than PAS. This suggests that the dependencies between the nested PASs are in some way captured by the PT information. Indeed, it should be noted that we join predicates only in case one is subordinate to the other, thus considering only a restricted set of all possible predicate dependencies. However, the improvement over PAS confirms that PASN is the right direction to encode shallow semantics from different sentence predicates.

Baseline	P	R	F1-measure
Gg@5	39.22±3.59	33.15±4.22	35.92±3.95
QA@5	39.72±3.44	34.22±3.63	36.76±3.56
Gg@all	31.58±0.58	100	48.02±0.67
QA@all	31.58±0.58	100	48.02±0.67
	Gg	QA	Re-ranker
MRR	48.97±3.77	56.21±3.18	81.12±2.12

Table 2: Baseline classifiers accuracy and MRR of YourQA (QA), Google (Gg) and the best re-ranker

4.3 Answer re-ranking

The output of the answer classifier can be used to re-rank the list of candidate answers of a QA system. Starting from the top answer, each instance can be classified based on its correctness with respect to the question. If it is classified as correct its rank is unchanged; otherwise it is pushed down, until a lower ranked incorrect answer is found.

We used the answer classifier with the highest F1-measure on the development set according to different cost-factor values[6]. We applied such model to the Google ranks and to the ranks of our Web-based QA system, i.e. YourQA. The latter uses Web documents corresponding to the top 20 Google results for the question. Then, each sentence in each document is compared to the question via a blend of similarity metrics used in the answer extraction phase to select the most relevant sentence. A passage of up to 750 bytes is then created around the sentence and returned as an answer.

Table 2 illustrates the results of the answer classifiers derived by exploiting Google (Gg) and YourQA (QA) ranks: the top N ranked results are considered as correct definitions and the remaining ones as in-

[6]However, by observing the curves in Fig. 5, the selected parameters appear as pessimistic estimates for the best model improvement: the one for BOW is the absolute maximum, but an average one is selected for the best model.

correct for different values of N. We show N = 5 and the maximum N (all), i.e. all the available answers. Each measure is the average of the Precision, Recall and F1-measure from cross validation. The F1-measure of Google and YourQA are greatly outperformed by our answer classifier.

The last row of Table 2 reports the MRR[7] achieved by Google, YourQA (QA) and YourQA after re-ranking (Re-ranker). We note that Google is outperformed by YourQA since its ranks are based on whole documents, not on single passages. Thus Google may rank a document containing several sparsely distributed question words higher than documents with several words concentrated in one passage, which are more interesting. When the answer classifier is applied to improve the YourQA ranking, the MRR reaches 81.1%, rising by about 25%.

Finally, it is worth to note that the answer classifier based on Q(BOW)+A(BOW,PT,PAS) model (parameterized as described) gave a 4% higher MRR than the one based on the simple BOW features. As an example, for question "What is foreclosure?", the sentence "Foreclosure means that the lender takes possession of your home and sells it in order to get its money back." was correctly classified by the best model, while BOW failed.

5 Conclusion

In this paper, we have introduced new structures to represent textual information in three question answering tasks: question classification, answer classification and answer re-ranking. We have defined tree structures (PAS and PASN) to represent predicate-argument relations, which we automatically extract using our SRL system. We have also introduced two functions, $SSTK$ and $K_{\texttt{all}}$, to exploit their representative power.

Our experiments with SVMs and the above models suggest that syntactic information helps tasks such as question classification whereas semantic information contained in PAS and PASN gives promising results in answer classification.

In the future, we aim to study ways to capture relations between predicates so that more general semantics can be encoded by PASN. Forms of generalization for predicates and arguments within PASNs like LSA clusters, WordNet synsets and FrameNet (roles and frames) information also appear as a promising research area.

[7]The Mean Reciprocal Rank is defined as: $MRR = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{rank_i}$, where n is the number of questions and $rank_i$ is the rank of the first correct answer to question i.

Acknowledgments

We thank the anonymous reviewers for their helpful suggestions. Alessandro Moschitti would like to thank the AMI2 lab at the University of Trento and the EU project LUNA "spoken Language UNderstanding in multilinguAl communication systems" contract n° 33549 for supporting part of his research.

References

J. Allan, J. Aslam, N. Belkin, and C. Buckley. 2002. Challenges in IR and language modeling. In *Report of a Workshop at the University of Amherst*.

X. Carreras and L. Màrquez. 2005. Introduction to the CoNLL-2005 shared task: SRL. In *CoNLL-2005*.

Y. Chen, M. Zhou, and S. Wang. 2006. Reranking answers from definitional QA using language models. In *ACL'06*.

M. Collins and N. Duffy. 2002. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *ACL'02*.

K. Collins-Thompson, J. Callan, E. Terra, and C. L.A. Clarke. 2004. The effect of document retrieval quality on factoid QA performance. In *SIGIR'04*. ACM.

H. Cui, M. Kan, and T. Chua. 2005. Generic soft pattern models for definitional QA. In *SIGIR'05*. ACM.

T. Joachims. 1999. Making large-scale SVM learning practical. In *Advances in Kernel Methods - Support Vector Learning*.

H. Kazawa, H. Isozaki, and E. Maeda. 2001. NTT question answering system in TREC 2001. In *TREC'01*.

P. Kingsbury and M. Palmer. 2002. From Treebank to PropBank. In *LREC'02*.

C. C. T. Kwok, O. Etzioni, and D. S. Weld. 2001. Scaling question answering to the web. In *WWW'01*.

X. Li and D. Roth. 2005. Learning question classifiers: the role of semantic information. *Journ. Nat. Lang. Eng.*

A. Moschitti, B. Coppola, A. Giuglea, and R. Basili. 2005. Hierarchical semantic role labeling. In *CoNLL 2005 shared task*.

A. Moschitti. 2006. Efficient convolution kernels for dependency and constituent syntactic trees. In *ECML'06*.

S. Quarteroni and S. Manandhar. 2006. User modelling for Adaptive Question Answering and Information Retrieval. In *FLAIRS'06*.

E. M. Voorhees. 2001. Overview of the TREC 2001 QA track. In *TREC'01*.

D. Zelenko, C. Aone, and A. Richardella. 2003. Kernel methods for relation extraction. *Journ. of Mach. Learn. Res.*

D. Zhang and W. Lee. 2003. Question classification using support vector machines. In *SIGIR'03*. ACM.