Semantic Convolution Kernels Over Dependency Trees

Danilo Croce
DII
University of Tor Vergata
00133 Roma, Italy
croce@info.uniroma2.it

Alessandro Moschitti
DISI
University of Trento
38123 Povo (TN), Italy
moschitti@disi.unitn.it

Roberto Basili
DII
University of Tor Vergata
00133 Roma, Italy
basili@info.uniroma2.it

ABSTRACT

In recent years, natural language processing techniques have been used more and more in IR. Among other syntactic and semantic parsing are effective methods for the design of complex applications like for example question answering and sentiment analysis. Unfortunately, extracting feature representations suitable for machine learning algorithms from linguistic structures is typically difficult. In this paper, we describe one of the most advanced piece of technology for automatic engineering of syntactic and semantic patterns. This method merges together convolution dependency tree kernels with lexical similarities. It can efficiently and effectively measure the similarity between dependency structures, whose lexical nodes are in part or completely different. Its use in powerful algorithm such as Support Vector Machines (SVMs) allows for fast design of accurate automatic systems. We report some experiments on question classification, which show an unprecedented result, e.g. 41% of error reduction of the former state-of-the-art, along with the analysis of the nice properties of the approach.

1. INTRODUCTION

Recent years have shown that syntactic and semantic structures are becoming essential for solving complex IR tasks, e.g., in question answering [17, 15, 2, 16] and opinion mining [10, 9, 11].

Tree kernels are a valid approach to avoid the difficulty of manually designing effective features from linguistic structures [14]. Indeed, they can directly define a similarity between data points in terms of all possible substructures in an implicit vector space. However, when the availability of training data is scarce, lexical data in the structures above should be generalized to obtain more general structural patterns.

In this perspective, one interesting approach, proposed in [3, 4], encoded lexical similarity in tree kernels. The model is essentially the Syntactic Tree Kernel (STK) proposed in [6], in which syntactic fragments from constituency trees can be matched even if they differ in the leaf nodes (i.e., they are constituted by related words with different surface forms). This kernel uses matching scores between fragments (i.e., features) lower than one, depending on the semantic similarity of the corresponding leaves in the syntactic fragments.

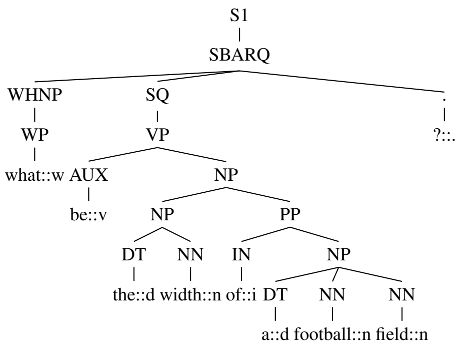

Figure 1: The Constituent Tree (CT) of a question.

Although, such idea is promising and interesting, shows clear limitations: (i) rather limited possibility to exploit semantic smoothing, e.g., trivially the syntactic structure associated with *the big beautiful apple* will only match *apple/orange* when compared to *a nice large orange*; and (ii) STK cannot be effectively applied to dependency structures, e.g., see experiments and motivation in [14]. To overcome such issues, in [7], we augmented the tree kernel in [14], namely the Partial Tree Kernel (PTK), which generalizes STK, with node similarity, e.g. between the lexical nodes. This allows for using any tree and any similarity between nodes in any position of the tree (not just on the leaves as in [4]). In other words, the new Smoothed PTK (SPTK) can automatically provide the learning algorithm, e.g. Support Vector Machines (SVMs), with a huge set of generalized structural patterns by simply applying it to the structural representation of instances of the target task.

In this paper, we analyze SPTK in terms of accuracy, efficiency and error analysis by also comparing it with previous kernels. The extensive experimentation on the question classification (QC) dataset shows that SPTK:

- outperforms any previous kernels achieving an unprecedented result of 41% of error reduction with respect to the former state-of-the-art; and
- is rather efficient to be applied to typical machine learning tasks.

It should be also noted that the most important property of SPTK is its generalization ability, which of course is extremely useful in scarce training data condition.

In the reminder of this paper, Section 2 illustrates our representation of questions by means of syntactic structures. Section 3

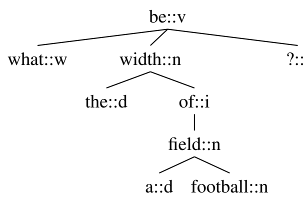

Figure 2: Lexical Only Centered Tree (LOCT).

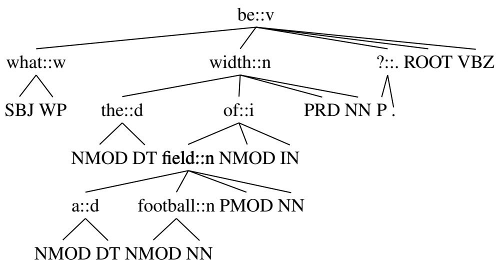

Figure 3: Lexical Centered Tree (LCT).

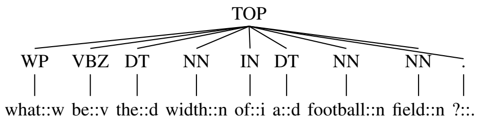
TOP
WP VBZ DT NN IN DT NN NN .
what::w be::v the::d width::n of::i a::d football::n field::n ?::.

Figure 4: Lexical and PoS-Tag Sequences Tree (LPST).

presents the experimental evaluation for QC and Section 4 derives the conclusions.

2. COMPUTATIONAL STRUCTURES FOR QUESTION CLASSIFICATION

Thanks to structural kernel similarity, a question classification (QC) task can be easily modeled by representing questions, i.e., the classification objects, with their parse trees. Several syntactic representations exist, we report the most interesting and effective structures that we proposed in [7]. Given the following sentence:

(s1) *What is the width of a football field?*

the representation tree according to a phrase structure paradigm, i.e. constituency tree (CT), is in Figure 1. We apply lemmatization to the lexicals to improve generalization and, at the same time, we add a generalized PoS-tag, i.e. noun (n::), verb (v::), adjective (::a), determiner (::d) and so on, to them. This is useful to measure similarity between lexicals belonging to the same grammatical category. Our conversion of dependency structures in dependency trees is done in two steps:

- we generate the tree that includes only lexicals, where the edges encode their dependencies. We call it the Lexical Only Centered Tree (LOCT), e.g. see Figure 2.
- To each lexical node, we add two leftmost children, which encode the grammatical function and POS-Tag, i.e. node features. We call this structure the Lexical Centered Tree (LCT), e.g. see Figure 3.

Additionally, for comparative purposes, we define a flat structure, the Lexical and PoS-tag Sequences Tree (LPST), e.g. see Figure 4, which ignores the syntactic structure of the sentence being a simple sequence of PoS-tag nodes, where lexicals are simply added as children.

	STK	PTK	SPTK(LSA)
CT	91.20%	90.80%	91.00%
LOCT	-	89.20%	93.20%
LCT	-	90.80%	**94.80%**
LPST	-	89.40%	89.60%
BOW		88.80%	

Table 1: Accuracy of structural kernels applied to different structures on QC

3. EXPERIMENTS

The aim of the experiments is to analyze the role of lexical similarity embedded in syntactic structures. For this purpose, we present results on QC and the related error analysis.

3.1 Setup

Our referring corpus is the UIUC dataset [13]. It is composed by a training set of 5,452 questions and a test set of 500 questions[1]. The latter are organized in six coarse-grained classes, i.e., ABBREVIATION, ENTITY, DESCRIPTION, HUMAN, LOCATION and NUMBER.

For learning our models, we extended the SVM-LightTK software[2] [14, 15] (which includes structural kernels, i.e., STK and PTK in SVMLight [8]) with the smooth match between tree nodes, i.e. the SPTK defined in [7].

For generating constituency trees, we used Charniak's parser [5] whereas we applied LTH syntactic parser (described in [12]) to generate dependency trees.

The lexical similarity was designed with LSA applied to ukWak [1], which is a large scale document collection made by 2 billion tokens (see [7] for more details). We implemented multi-classification using *one-vs-all* scheme and selecting the category associated with the maximum SVM margin.

3.2 Classification Results

The F1 of SVMs using (i) STK applied to CT and (ii) PTK and SPTK applied to the several structures for QC is reported in Table 1. The first column shows the different structures described in Section 2. The first row lists the tree kernel models. The last row reports the accuracy of bag-of-words (BOW), which is a linear kernel applied to lexical vectors.

It is worth nothing that:

- BOW produces high accuracy, i.e. 88.8% but it is improved by STK, current state-of-the-art[3] in QC [18, 17];
- PTK applied to the same tree of STK (i.e. CT) produces a slightly lower value (non-statistically significant difference); and
- PTK applied to LCT, which contains structures but also grammatical functions and PoS-tags, achieves higher accuracy than when applied to LOCT (no grammatical/syntactic features) or to LPST (no structure).

[1]http://cogcomp.cs.illinois.edu/Data/QA/QC/
[2]http://disi.unitn.it/moschitti/Tree-Kernel.htm
[3]Note that higher accuracy values for smoothed STK are shown in [4] but the one optimizing a validation set is not shown.

Most importantly, SPTK (using LSA similarity):

- improves PTK on all the syntactic-based structures;
- (ii) gets the lowest improvement when applied to CT; and
- (iii) achieves the impressive result of 94.80% with LCT, i.e more than 41% of relative error reduction.

This suggests that the sequences of similar lexicals when selected by syntactic structures produce accurate features. Indeed, when syntax is missing such as for the unstructured lexical sequences of LPST, the accuracy does not highly improve.

To better understand the role of lexical similarity in syntactic structures, in the next section, we will outline the benefit of SPTK over the other models by analyzing the classification outcome.

3.3 Error Analysis

Table 2 shows four questions (from 1 to 4) along with: the flags +/- asserting the presence or not of an error, the classification model, the true label of the question and the one given by the classifier.

The first question is not correctly classified by BOW, which assigned ENTITY category. Indeed, from a bag-of-words perspective ENTITY and LOCATION are equally probable for question (1), e.g., the question *What is the French cognac produced in province?* uses the same words but asks for an entity. However, it is enough to add some syntactic cues to solve the above ambiguity. For example, the previous question would generate a dependency between *What* and *is* whereas question (1) generates a dependency between *What* and *French province*. These syntactic features are contained in both tree representations; indeed, all the syntactic based models correctly classify this example.

The second question is mistaken by all models but SPTK. The explanation is that without knowing that *ruler* is a person, it is not possible to infer the expected category of the answer. Indeed, if *ruler* had been a synonym of *army* or *region* the expected answer type would have been ENTITY. SPTK can disambiguate between ENTITY and HUMAN by exploiting the similarity with the training question *What Mexican leader was shot dead in 1923 ?*. This shows some structural similarity, which is reinforced by the lexical similarity between *French* and *Mexican*, between *ruler* and *leader*, and between *defeat* and *shot*.

A similar rationale applies to the third question: without knowing that *peninsula* indicates a geographic location the most probable category could be ENTITY. In contrast, SPTK can provide the correct answer by measuring the structural similarity of question (3) with the training question: *What island group is Guadalcanal a part of ?* along with the lexical similarity between *peninsula* and *island* and between *Spain* and *Guadalcanal*.

The last question is correctly classified only using the dependency structure. The BOW model is too influenced by words such as *least, amount* and *per*. The costituency structure fails as well, probably because it does not contain structures encoding lexical trigrams like *what-state-have*, which instead are subtrees of LCT.

Finally, the errors of SPTK refer to questions like *What did Jesse Jackson organize?*, where the classifier selected ENTITY instead of HUMAN category, or *What is the melting point of copper ?* where ENTITY is selected instead of the correct NUMBER. These are clear examples where specific background knowledge is needed to provide the correct answer.

3.4 Kernel generalization and efficiency

To understand the role of syntactic/semantic kernels, it is interesting to study their impact on the SVM generalization. For this purpose, Fig. 5 reports the learning curve of BOW, of STK and

What French province is cognac produced in? (1)			
-	BOW	LOC	ENTY
+	CT-STK		LOC
+	LCT-PTK		LOC
+	LCT-SPTK		LOC
What French ruler was defeated at the battle of Waterloo? (2)			
-	BOW	HUM	ENTY
-	CT-STK		ENTY
-	LCT-PTK		ENTY
+	LCT-SPTK		HUM
What peninsula is Spain part of? (3)			
-	BOW	LOC	ENTY
-	CT-STK		ENTY
-	LCT-PTK		ENTY
+	LCT-SPTK		LOC
What state has the least amount of rain per year? (4)			
-	BOW	LOC	NUM
-	CT-STK		NUM
+	LCT-PTK		LOC
+	LCT-SPTK		LOC

Table 2: Some interesting question classification case

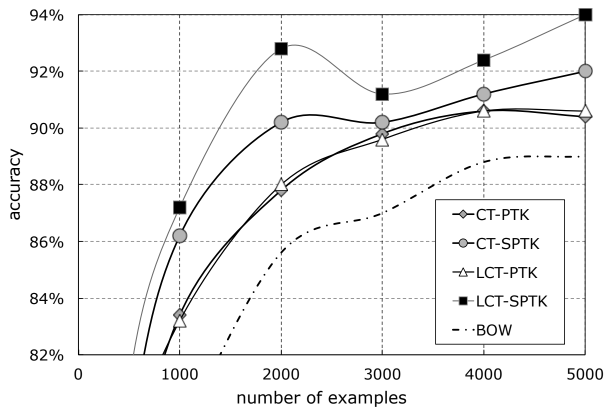

Figure 5: Learning curves: comparison with no similarity

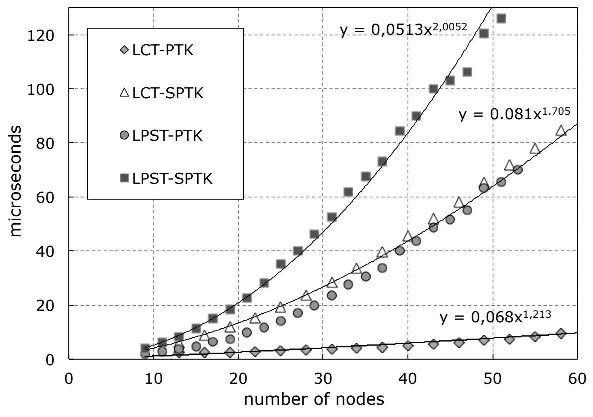

Figure 6: Micro-seconds for each kernel computation

PTK applied to CT and of PTK and SPTK applied to LCT. We note that:

- BOW learning curve is clearly below those of syntactic-based models;
- LCT and CT (dependency and constituency paradigms) show very similar curves, suggesting that the two syntactic representations are equivalent; and
- when SPTK is used (i.e., CT-SPTK and LCT-SPTK) the generalization of SVMs meaningful increases.
- SPTK applied to LCT shows the steepest curve, achieving with just 1/5 of the available data the same accuracy of BOW on all data.

The benefits in terms of accuracy of SPTK are clear thus it is important to demonstrate that it can be efficiently applied to large datasets. For this purpose, we plotted the average running time of each computation of PTK and SPTK applied to the different structures. We divided the examples from QC based on the number of nodes in each example. Figure 6 shows the elapsed time in function of the number of nodes for different tree representations. We note that: (i) LCT-PTK is very fast as we used the fast algorithm designed in [14]; (ii) LCT-SPTK is also very fast as it uses the same algorithm of PTK but it tends to match many more tree fragments thus its complexity increases. However, the equation of the curve fit, shown in the figure, suggests that the trend is sub-quadratic , i.e., $x^{1.7}$. This efficiency is due to the tree structure, which imposes hierarchical matching of subtrees. (iii) Only when SPTK is applied to LPST, which has no structure, it matches all possible similar subsequences of nodes. This increases its computational complexity, which results in an order higher than 2.

4. CONCLUSIONS

This paper has investigated the properties of a novel tree kernel, namely SPTK, which can encode generalized syntactic patterns from dependency or constituency structures. The main characteristic of SPTK is its ability to measure the similarity between syntactic structures, which are partially similar and whose lexical nodes can be different but related, e.g., Mexican and Spain. This allows SVMs to exploit large feature spaces, automatically generated from dependency substructures.

We have tested SPTK on the question classification (QC) task by also comparing with previous state-of-the-art models. The results show that SPTK with SVMs achieves an unprecedented result for QC, i.e., 94.8% of accuracy.

The error analysis has revealed that syntactic structures are needed to disambiguate the lexical semantic of questions. However, they may be either too general if lexicals are not part of them or too sparse if they are based on several lexicals. Therefore, SPTK, generalizing the latter, provides a compromise that improves accuracy and generalization ability of SVMs.

Finally, we have also investigated the computational complexity of SPTK by empirically showing that it can easily scale to large datasets. Such result enables many promising future research directions: the most important being the use of SPTK for many IR tasks with many different similarities. It is also interesting to note that SPTK can be applied to trees completely different from syntactic parses, e.g., XML trees, on which a general semantic similarity can be defined between nodes.

Acknowledgements

This work has been partially supported by the EC project FP247758: Trustworthy Eternal Systems via Evolving Software, Data and Knowledge (EternalS).

5. REFERENCES

[1] M. Baroni, S. Bernardini, A. Ferraresi, and E. Zanchetta. The wacky wide web: a collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226, 2009.

[2] M. W. Bilotti, J. L. Elsas, J. Carbonell, and E. Nyberg. Rank learning for factoid question answering with linguistic and semantic constraints. In *Proceedings of ACM CIKM*, 2010.

[3] S. Bloehdorn and A. Moschitti. Combined syntactic and semantic kernels for text classification. In *Proceedings of ECIR 2007, Rome, Italy*, 2007.

[4] S. Bloehdorn and A. Moschitti. Structure and semantics for expressive text kernels. In *Proceedings of CIKM*, 2007.

[5] E. Charniak. A maximum-entropy-inspired parser. In *Proceedings of NAACL'00*, 2000.

[6] M. Collins and N. Duffy. New Ranking Algorithms for Parsing and Tagging: Kernels over Discrete Structures, and the Voted Perceptron. In *Proceedings of ACL'02*, 2002.

[7] D. Croce, A. Moschitti, and R. Basili. Structured lexical similarity via convolution kernels on dependency trees. In *Proceedings of EMNLP*, Edinburgh, Scotland, UK., 2011.

[8] T. Joachims. Estimating the generalization performance of a SVM efficiently. In *Proceedings of ICML'00*, 2000.

[9] R. Johansson and A. Moschitti. Reranking models in fine-grained opinion analysis. In *Proceedings of Coling 2010*, pages 519–527, Beijing, China, 2010.

[10] R. Johansson and A. Moschitti. Syntactic and semantic structure for opinion expression detection. In *Proceedings of CoNLL*, pages 67–76, Uppsala, Sweden, 2010.

[11] R. Johansson and A. Moschitti. Extracting opinion expressions and their polarities – exploration of pipelines and joint models. In *Proceedings of ACL-HLT*, Portland, Oregon, USA, 2011.

[12] R. Johansson and P. Nugues. Dependency-based syntactic–semantic analysis with PropBank and NomBank. In *Proceedings of CoNLL*, Manchester, United Kingdom, 2008.

[13] X. Li and D. Roth. Learning question classifiers. In *Proceedings of ACL'02*, 2002.

[14] A. Moschitti. Efficient convolution kernels for dependency and constituent syntactic trees. In *Proceedings of ECML'06*, pages 318–329, 2006.

[15] A. Moschitti. Kernel methods, syntax and semantics for relational text categorization. In *Proceeding of CIKM '08*, NY, USA, 2008.

[16] A. Moschitti, J. Chu-carroll, S. Patwardhan, J. Fan, and G. Riccardi. Using syntactic and semantic structural kernels for classifying definition questions in jeopardy! In *Proceedings of EMNLP*, Edinburgh, Scotland, UK., 2011.

[17] A. Moschitti, S. Quarteroni, R. Basili, and S. Manandhar. Exploiting syntactic and shallow semantic kernels for question/answer classification. In *Proceedings of ACL'07*, 2007.

[18] D. Zhang and W. S. Lee. Question classification using support vector machines. In *Proceedings of ACM SIGIR*. ACM Press, 2003.