Encoding Tree Pair-based Graphs in Learning Algorithms: the Textual Entailment Recognition Case

Alessandro Moschitti
DISI, University of Trento
Via Sommarive 14
38100 POVO (TN) - Italy
moschitti@dit.unitn.it

Fabio Massimo Zanzotto
DISP, University of Rome "Tor Vergata"
Via del Politecnico 1
00133 Roma, Italy
zanzotto@info.uniroma2.it

Abstract

In this paper, we provide a statistical machine learning representation of textual entailment via syntactic graphs constituted by tree pairs. We show that the natural way of representing the syntactic relations between text and hypothesis consists in the huge feature space of all possible syntactic tree fragment pairs, which can only be managed using kernel methods. Experiments with Support Vector Machines and our new kernels for paired trees show the validity of our interpretation.

1 Introduction

Recently, a lot of valuable work on the recognition of textual entailment (RTE) has been carried out (Bar Haim et al., 2006). The aim is to detect implications between sentences like:

$T_1 \Rightarrow H_1$	
T_1	*"Wanadoo bought KStones"*
H_1	*"Wanadoo owns KStones"*

where T_1 and H_1 stand for text and hypothesis, respectively.

Several models, ranging from the simple lexical similarity between T and H to advanced Logic Form Representations, have been proposed (Corley and Mihalcea, 2005; Glickman and Dagan, 2004; de Salvo Braz et al., 2005; Bos and Markert, 2005). However, since a linguistic theory able to analytically show how to computationally solve the RTE problem has not been developed yet, to design accurate systems, we should rely upon the application of machine learning. In this perspective, TE training examples have to be represented in terms of statistical feature distributions. These typically consist in word sequences (along with their lexical similarity) and the syntactic structures of both text and hypothesis (e.g. their parse trees). The interesting aspect with respect to other natural language problems is that, in TE, features useful at describing an example are composed by pairs of features from Text and Hypothesis.

For example, using a word representation, a text and hypothesis pair, $\langle T, H\rangle$, can be represented by the sequences of words of the two sentences, i.e. $\langle t_1, .., t_n\rangle$ and $\langle h_1, .., h_m\rangle$, respectively. If we carry out a blind and complete statistical correlation analysis of the two sequences, the entailment property would be described by the set of subsequence pairs from T and H, i.e. the set $R = \{\langle s_t, s_h\rangle : s_t = \langle t_{i_1}, .., t_{i_l}\rangle, s_h = \langle h_{j_1}, .., h_{j_r}\rangle, l \leq n, r \leq m\}$. The relation set R constitutes a naive and complete representation of the example $\langle T, H\rangle$ in the feature space $\{\langle v, w\rangle : v, w \in V^*\}$, where V is the corpus vocabulary[1].

Although the above representation is correct and complete from a statistically point of view, it suffers from two practical drawbacks: (a) it is exponential in V and (b) it is subject to high degree of data sparseness which may prevent to carry out effective learning. The traditional solution for this problem relates to consider the syntactic structure of word sequences which provides their generalization.

The use of syntactic trees poses the problem of representing structures in learning algorithms. For this purpose, kernel methods, and in particular tree kernels allow for representing trees in

[1] V^* is larger than the actual space, which is the one of all possible subsequences with gaps, i.e. it only contains all possible concatenations of words respecting their order.

terms of all possible subtrees (Collins and Duffy, 2002). Unfortunately, the representation in entailment recognition problems requires the definition of kernels over graphs constituted by tree pairs, which are in general different from kernels applied to single trees. In (Zanzotto and Moschitti, 2006), this has been addressed by introducing semantic links (placeholders) between text and hypothesis parse trees and evaluating two distinct tree kernels for the trees of texts and for those of hypotheses. In order to make such disjoint kernel combination effective, all possible assignments between the placeholders of the first and the second entailment pair were generated causing a remarkable slowdown.

In this paper, we describe the feature space of all possible tree fragment pairs and we show that it can be evaluated with a much simpler kernel than the one used in previous work, both in terms of design and computational complexity. Moreover, the experiments on the RTE datasets show that our proposed kernel provides higher accuracy than the simple union of tree kernel spaces.

2 Fragments of Tree Pair-based Graphs

The previous section has pointed out that RTE can be seen as a relational problem between word sequences of Text and Hypothesis. The syntactic structures embedded in such sequences can be generalized by natural language grammars. Such generalization is very important since it is evident that entailment cases depend on the syntactic structures of Text and Hypothesis. More specifically, the set R described in the previous section can be extended and generalized by considering syntactic derivations[2] that generate word sequences in the training examples. This corresponds to the following set of tree fragment pairs:

$$R^{\tau} = \{\langle \tau_t, \tau_h \rangle : \tau_t \in \mathcal{F}(T), \tau_h \in \mathcal{F}(H)\}, \quad (1)$$

where $\mathcal{F}(\cdot)$ indicates the set of tree fragments of a parse tree (i.e. the one of the text T or of the hypothesis H). R^{τ} contains less sparse relations than R. For instance, given T_1 and H_1 of the previous section, we would have the following relational description:

[2] By cutting derivation at different depth, different degrees of generalization can be obtained.

$R^{\tau} = \{$ ⟨ NP(NNP) , NP(NNP) ⟩ , ⟨ S(NP VP) , S(NP VP) ⟩ ,
⟨ S(NP(NNP), VP(VBP(bought), NP(NNP))) , S(NP(NNP), VP(VBP(owns), NP(NNP))) ⟩ ,
⟨ VP(VBP(bought), NP(NNP)) , VP(VBP(owns), NP(NNP)) ⟩ , .. $\}$

These features (relational pairs) generalize the entailment property, e.g. the pair ⟨[VP [VBP bought] [NP]], [VP [VBP own] [NP]]⟩ generalizes many word sequences, i.e. those external to the verbal phrases and internal to the NPs.

We can improve this space by adding semantic links between the tree fragments. Such links or placeholders have been firstly proposed in (Zanzotto and Moschitti, 2006). A placeholder assigned to a node of τ_t and a node of τ_h states that such nodes dominate the same (or similar) information. In particular, placeholders are assigned to nodes whose words t_i in T are equal, similar, or semantically dependent on words h_j in H. Using placeholders, we obtain a richer fragment pair based representation that we call $R^{\tau p}$, exemplified hereafter:

$\{$ ⟨ S(NP(NNP X), VP(VBP(bought), NP(NNP Y))) , S(NP(NNP X), VP(VBP(owns), NP(NNP Y))) ⟩
, ⟨ S(NP, VP(VBP(bought), NP(NNP Y))) , S(NP, VP(VBP(owns), NP(NNP Y))) ⟩
, ⟨ S(NP VP) , S(NP VP) ⟩ , ... $\}$

The placeholders (or variables) indicated with X and Y specify that the NNPs labeled by the same variables dominate similar or identical words. Therefore, an automatic algorithm that assigns placeholders to semantically similar con-

stituents is needed. Moreover, although $R^{\tau p}$ contains more semantic and less sparse features than both R^{τ} and R, its cardinality is still exponential in the number of the words of T and H. This means that standard machine learning algorithms cannot be applied. In contrast, tree kernels (Collins and Duffy, 2002) can be used to efficiently generate the huge space of tree fragments but, to generate the space of pairs of tree fragments, a new kernel function has to be defined.

The next section provides a solution to both problems. i.e. an algorithm for placeholders assignments and for the computation of paired tree kernels which generates R^{τ} and $R^{\tau p}$ representations.

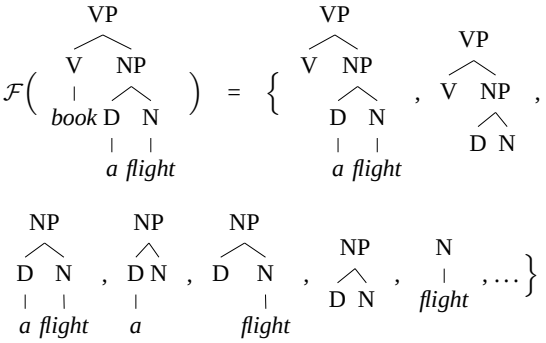

Figure 1: A syntactic parse tree.

3 Kernels over Semantic Tree Pair-based Graphs

The previous section has shown that placeholders enrich a tree-based graph with relational information, which, in turn, can be captured by means of word semantic similarities $sim_w(w_t, w_h)$, e.g. (Corley and Mihalcea, 2005; Glickman et al., 2005). More specifically, we use a two-step greedy algorithm to anchor the content words (verbs, nouns, adjectives, and adverbs) in the hypothesis W_H to words in the text W_T.

In the first step, each word w_h in W_H is connected to all words w_t in W_T that have the maximum similarity $sim_w(w_t, w_h)$ with it (more than one w_t can have the maximum similarity with w_h). As result, we have a set of anchors $A \subset W_T \times W_H$. $sim_w(w_t, w_h)$ is computed by means of three techniques:

1. Two words are maximally similar if they have the same surface form $w_t = w_h$.
2. Otherwise, WordNet (Miller, 1995) similarities (as in (Corley and Mihalcea, 2005)) and different relation between words such as verb entailment and derivational morphology are applied.
3. The edit distance measure is finally used to capture the similarity between words that are missed by the previous analysis (for misspelling errors or for the lack of derivational forms in WordNet).

In the second step, we select the final anchor set $A' \subseteq A$, such that $\forall w_t$ (or w_h) $\exists! \langle w_t, w_h \rangle \in A'$. The selection is based on a simple greedy algorithm that given two pairs $\langle w_t, w_h \rangle$ and $\langle w'_t, w_h \rangle$ to be selected and a pair $\langle s_t, s_h \rangle$ already selected, considers word proximity (in terms of number of words) between w_t and s_t and between w'_t and s_t; the nearest word will be chosen.

Once the graph has been enriched with semantic information we need to represent it in the learning algorithm; for this purpose, an interesting approach is based on kernel methods. Since the considered graphs are composed by only two trees, we can carried out a simplified computation of a graph kernel based on tree kernel pairs.

3.1 Tree Kernels

Tree Kernels (e.g. see NLP applications in (Giuglea and Moschitti, 2006; Zanzotto and Moschitti, 2006; Moschitti et al., 2007; Moschitti et al., 2006; Moschitti and Bejan, 2004)) represent trees in terms of their substructures (fragments) which are mapped into feature vector spaces, *e.g.* $\Re^n$. The kernel function measures the similarity between two trees by counting the number of their common fragments. For example, Figure 1 shows some substructures for the parse tree of the sentence `"book a flight"`. The main advantage of tree kernels is that, to compute the substructures shared by two trees τ_1 and τ_2, the whole fragment space is not used. In the following, we report the formal definition presented in (Collins and Duffy, 2002).

Given the set of fragments $\{f_1, f_2, ..\} = \mathcal{F}$, the indicator function $I_i(n)$ is equal 1 if the target f_i is rooted at node n and 0 otherwise. A tree kernel is then defined as:

$$TK(\tau_1, \tau_2) = \sum_{n_1 \in N_{\tau_1}} \sum_{n_2 \in N_{\tau_2}} \Delta(n_1, n_2) \quad (2)$$

where N_{τ_1} and N_{τ_2} are the sets of the τ_1's and τ_2's nodes, 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 the n_1 and n_2 nodes and Δ can be evaluated with the following algorithm:

1. if the productions 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 have only leaf children (i.e. they are pre-terminals 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

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

where $nc(n_1)$ is the number of the children of n_1 and c_n^j is the j-th child of the node n. Note that since the productions are the same, $nc(n_1) = nc(n_2)$.

Additionally, we add the decay factor λ by modifying steps (2) and (3) as follows[3]:

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

The computational complexity of Eq. 2 is $O(|N_{\tau_1}| \times |N_{\tau_2}|)$ although the average running time tends to be linear (Moschitti, 2006).

3.2 Tree-based Graph Kernels

The above tree kernel function can be applied to the parse trees of two texts or those of the two hypotheses to measure their similarity in terms of the shared fragments. If we sum the contributions of the two kernels (for texts and for hypotheses) as proposed in (Zanzotto and Moschitti, 2006), we just obtain the feature space of the union of the fragments which is completely different from the space of the tree fragments pairs, i.e. R^τ. Note that the union space is not useful to describe which grammatical and lexical property is at the same time held by T and H to trig the implication.

Therefore to generate the space of the fragment pairs we need to define the kernel between two pairs of entailment examples $\langle T_1, H_1 \rangle$ and $\langle T_2, H_2 \rangle$ as

$$K_p(\langle T_1, H_1 \rangle, \langle T_2, H_2 \rangle) = \\ = \sum_{n_1 \in T_1} \sum_{n_2 \in T_2} \sum_{n_3 \in H_1} \sum_{n_4 \in H_2} \Delta(n_1, n_2, n_3, n_4),$$

where Δ evaluates the number of subtrees rooted in n_1 and n_2 combined with those rooted in n_3 and n_4. More specifically, each fragment rooted into the nodes of the two texts' trees is combined with each fragment rooted in the two hypotheses' trees. Now, since the number of subtrees rooted in the texts is independent of the number of trees rooted in the hypotheses,

$$\Delta(n_1, n_2, n_3, n_4) = \Delta(n_1, n_2)\Delta(n_3, n_4).$$

Therefore, we can rewrite K_p as:

$$K_p(\langle T_1, H_1 \rangle, \langle T_2, H_2 \rangle) = \\ = \sum_{n_1 \in T_1} \sum_{n_2 \in T_2} \sum_{n_3 \in H_1} \sum_{n_4 \in H_2} \Delta(n_1, n_2)\Delta(n_3, n_4) = \\ = \sum_{n_1 \in T_1} \sum_{n_2 \in T_2} \Delta(n_1, n_2) \sum_{n_3 \in H_1} \sum_{n_4 \in H_2} \Delta(n_3, n_4) = \\ = K_t(T_1, T_2) \times K_t(H_1, H_2). \quad (4)$$

This result shows that the natural kernel to represent textual entailment sentences is the kernel product, which corresponds to the set of all possible syntactic fragment pairs. Note that, such kernel can be also used to evaluate the space of fragment pairs for trees enriched with relational information, i.e. by placeholders.

4 Approximated Graph Kernel

The feature space described in the previous section correctly encodes the fragment pairs. However, such huge space may result inadequate also for algorithms such as SVMs, which are in general robust to many irrelevant features. An approximation of the fragment pair space is given by the kernel described in (Zanzotto and Moschitti, 2006). Hereafter we illustrate its main points.

[3] To have a similarity score between 0 and 1, we also apply the normalization in the kernel space, i.e. $K'(\tau_1, \tau_2) = \frac{TK(\tau_1, \tau_2)}{\sqrt{TK(\tau_1, \tau_1) \times TK(\tau_2, \tau_2)}}$.

First, tree kernels applied to two texts or two hypotheses match identical fragments. When placeholders are added to trees, the labeled fragments are matched only if the basic fragments and the assigned placeholders match. This means that we should use the same placeholders for all texts and all hypotheses of the corpus. Moreover, they should be assigned in a way that similar syntactic structures and similar relational information between two entailment pairs can be matched, i.e. same placeholders should be assigned to the potentially similar fragments.

Second, the above task cannot be carried out at pre-processing time, i.e. when placeholders are assigned to trees. At the running time, instead, we can look at the comparing trees and make a more consistent decision on the type and order of placeholders. Although, there may be several approaches to accomplish this task, we apply a basic heuristic which is very intuitive:

Choose the placeholder assignment that maximizes the tree kernel function over all possible correspondences

More formally, let A and A' be the placeholder sets of $\langle T, H\rangle$ and $\langle T', H'\rangle$, respectively, without loss of generality, we consider $|A| \geq |A'|$ and we align a subset of A to A'. The best alignment is the one that maximizes the syntactic and lexical overlapping of the two subtrees induced by the aligned set of anchors. By calling C the set of all bijective mappings from $S \subseteq A$, with $|S| = |A'|$, to A', an element $c \in C$ is a substitution function. We define the best alignment c_{max} the one determined by

$$c_{max} = argmax_{c\in C}(TK(t(T,c),t(T',i))+ TK(t(H,c),t(H',i)),$$

where (1) $t(\cdot, c)$ returns the syntactic tree enriched with placeholders replaced by means of the substitution c, (2) i is the identity substitution and (3) $TK(\tau_1, \tau_2)$ is a tree kernel function (e.g. the one specified by Eq. 2) applied to the two trees τ_1 and τ_2.

At the same time, the desired similarity value to be used in the learning algorithm is given by the kernel sum: $TK(t(T, c_{max}), t(T', i)) + TK(t(H, c_{max}), t(H', i))$, i.e. by solving the following optimization problem:

$$K_s(\langle T, H\rangle, \langle T', H'\rangle) = max_{c\in C}(TK(t(T,c),t(T',i))+ TK(t(H,c),t(H',i)), \quad (5)$$

For example, let us compare the following two pairs (T_1, H_1) and (T_2, H_2) in Fig. 2.

To assign the placeholders 1, 2 and 3 of (T_2, H_2) to those of (T_1, H_1), i.e. X and Y, we need to maximize the similarity between the two texts' trees and between the two hypotheses' trees. It is straightforward to derive that X=1 and Y=3 allow more substructures (i.e. large part of the trees) to be identical, e.g. [S [NP 1 X VP]] , [VP [VBP NP 3 Y]]], [S [NP 1 X VP [VBP NP 3 Y]]]].

Finally, it should be noted that, (a) $K_s(\langle T, H\rangle, \langle T', H'\rangle)$ is a symmetric function since the set of derivation C are always computed with respect to the pair that has the largest anchor set and (b) it is not a valid kernel as the max function does not in general produce valid kernels. However, in (Haasdonk, 2005), it is shown that when kernel functions are not positive semidefinite like in this case, SVMs still solve a data separation problem in pseudo Euclidean spaces. The drawback is that the solution may be only a local optimum. Nevertheless, such solution can still be valuable as the problem is modeled with a very rich feature space.

Regarding the computational complexity, running the above kernel on a large training set may result very expensive. To overcome this drawback, in (Moschitti and Zanzotto, 2007), it has been designed an algorithm to factorize the evaluation of tree subparts with respect to the different substitution. The resulting speed-up makes the application of such kernel feasible for datasets of ten of thousands of instances.

5 Experiments

The aim of the experiments is to show that the space of tree fragment pairs is the most effective to represent Tree Pair-based Graphs for the design of Textual Entailment classifiers.

5.1 Experimental Setup

To compare our model with previous work we implemented the following kernels in SVM-light (Joachims, 1999):

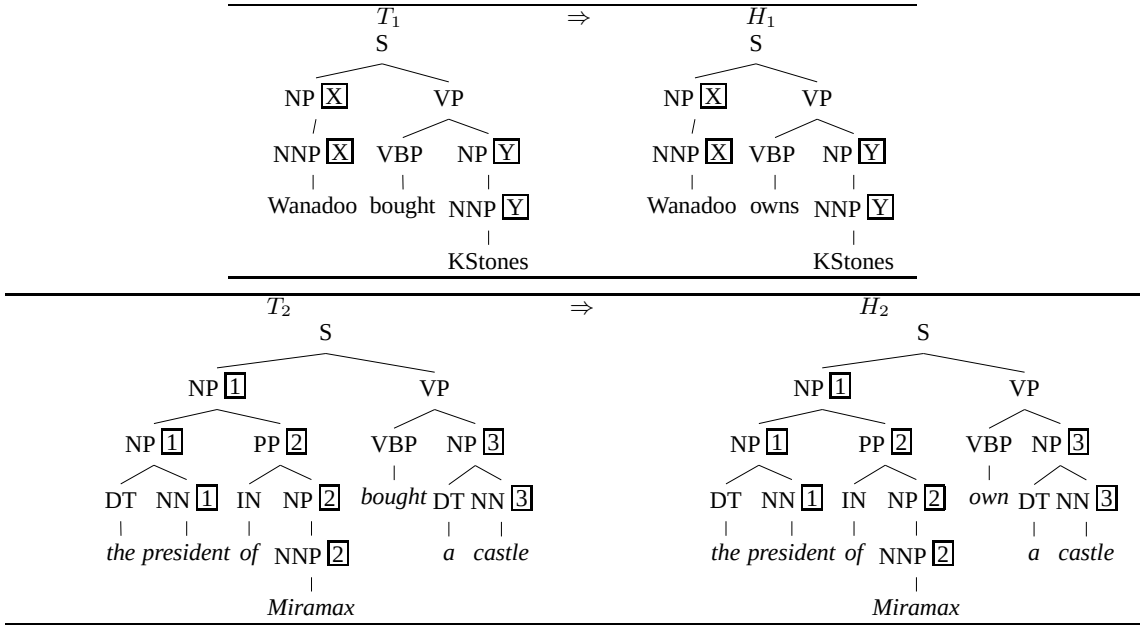

Figure 2: The problem of finding the correct mapping between placeholders

- $K_s(e_1, e_2) = K_t(T_1, T_2) + K_t(H_1, H_2)$, where $e_1 = \langle T_1, H_1 \rangle$ and $e_2 = \langle T_2, H_2 \rangle$ are two text and hypothesis pairs and K_t is the syntactic tree kernel (Collins and Duffy, 2002) presented in the previous section.

- $K_p(e_1, e_2) = K_t(T_1, T_2) \times K_t(H_1, H_2)$, which (as shown in the previous sections) encodes the tree fragment pairs with and without placeholders.

- $K_{max}(e_1, e_2) = \max_{c \in \mathcal{C}} \big(K_t(\phi_c(T_1), \phi_c(T_2)) + K_t(\phi_c(H_1), \phi_c(H_2))\big)$, where c is a possible placeholder assignment which connects nodes from the first pair with those of the second pair and $\phi_c(\cdot)$ transforms trees according to c.

- $K_{pmx}(e_1, e_2) = \max_{c \in \mathcal{C}} \big(K_t(\phi_c(T_1), \phi_c(T_2)) \times K_t(\phi_c(H_1), \phi_c(H_2))\big)$.

Note that K_{max} is the kernel proposed in (Zanzotto and Moschitti, 2006) and K_{pmx} is a hybrid kernel based on the maximum K_p, which uses the space of tree fragment pairs. For all the above kernels, we set the default cost factor and trade-off parameters and we set λ to 0.4.

To experiment with entailment relations, we used the data sets made available by the first (Dagan et al., 2005) and second (Bar Haim et al., 2006) Recognizing Textual Entailment Challenge. These corpora are divided in the development sets $D1$ and $D2$ and the test sets $T1$ and $T2$. $D1$ contains 567 examples whereas $T1$, $D2$ and $T2$ all have the same size, i.e. 800 instances. Each example is an ordered pair of texts for which the entailment relation has to be decided.

5.2 Evaluation and Discussion

Table 1 shows the results of the above kernels on the split used for the RTE competitions. The first column reports the kernel model. The second and third columns illustrate the model accuracy for RTE1 whereas column 4 and 5 show the accuracy for RTE2. Moreover, ¬ P indicates the use of standard syntactic trees and P the use of trees enriched with placeholders. We note that:

First, the space of tree fragment pairs, generated by K_p improves the one generated by K_s (i.e. the simple union of the fragments of texts and hypotheses) of 4 (58.9% vs 54.9%) and 0.9 (53.5% vs 52.6%) points on RTE1 and RTE2, respectively. This suggests that the fragment pairs are more effective for encoding the syntactic rules describing the entailment concept.

Second, on RTE1, the introduction of placeholders does not improve K_p or K_s suggesting that for their correct exploitation an extension of the space of tree fragment pairs should be modeled.

Third, on RTE2, the impact of placeholders

Kernels	RTE1		RTE2	
	¬ P	P	¬ P	P
K_s	54.9	50.0	52.6	59.5
K_p	**58.9**	55.5	**53.5**	56.0
$K_{\max}$	-	58.25	-	61.0
K_{pmx}	-	50.0	-	56.8

Table 1: Accuracy of different kernel models using (P) and not using (¬ P) placeholder information on RTE1 and RTE2.

seems more important but only K_{max} and K_s are able to fully exploit their semantic contribution. A possible explanation is that in order to use the set of all possible assignments (required by K_{max}), we needed to prune the "too large" syntactic trees as also suggested in (Zanzotto and Moschitti, 2006). This may have negatively biased the statistical distribution of tree fragment pairs.

Finally, although we show that K_p is better suited for RTE than the other kernels, its accuracy is lower than the state-of-the-art in RTE. This is because the latter uses additional models like the lexical similarity between text and hypothesis, which greatly improve accuracy.

6 Conclusion

In this paper, we have provided a statistical machine learning representation of textual entailment via syntactic graphs constituted by tree pairs. We have analytically shown that the natural way of representing the syntactic relations between text and hypothesis in learning algorithms consists in the huge feature space of all possible syntactic tree fragment pairs, which can only be managed using kernel methods.

Therefore, we used tree kernels, which allow for representing trees in terms of all possible subtrees. More specifically, we defined a new model for the entailment recognition problems, which requires the definition of kernels over graphs constituted by tree pairs. These are in general different from kernels applied to single trees. We also studied another alternative solution which concerns the use of semantic links (placeholders) between text and hypothesis parse trees (to form relevant semantic fragment pairs) and the evaluation of two distinct tree kernels for the trees of texts and for those of hypotheses. In order to make such disjoint kernel combination effective, all possible assignments between the placeholders of the first and the second entailment pair have to be generated (causing a remarkable slowdown).

Our experiments on the RTE datasets show that our proposed kernel may provide higher accuracy than the simple union of tree kernel spaces with a much simpler and faster algorithm. Future work will be devoted to make the tree fragment pair space more effective, e.g. by using smaller and accurate tree representation for text and hypothesis.

Acknowledgments

We would like to thank the anonymous reviewers for their professional and competent reviews and for their invaluable suggestions.
Alessandro Moschitti would like to thank the European Union project, LUNA (spoken **L**anguage **UN**derstanding in multilingu**A**l communication systems) contract n 33549 for supporting part of his research.

References

Bar Haim, Roy, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The II PASCAL RTE challenge. In *PASCAL Challenges Workshop*, Venice, Italy.

Bos, Johan and Katja Markert. 2005. Recognising textual entailment with logical inference. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 628–635, Vancouver, British Columbia, Canada, October. Association for Computational Linguistics.

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

Corley, Courtney and Rada Mihalcea. 2005. Measuring the semantic similarity of texts. In *Proc. of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, pages 13–18, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Dagan, Ido, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL RTE challenge. In *PASCAL Challenges Workshop*, Southampton, U.K.

de Salvo Braz, R., R. Girju, V. Punyakanok, D. Roth, and M. Sammons. 2005. An inference model for semantic entailment in natural language. In *Proceedings of AAAI*, pages 1678–1679.

Giuglea, Ana-Maria and Alessandro Moschitti. 2006. Semantic role labeling via framenet, verbnet and propbank. In *Proceedings of Coling-ACL*, Sydney, Australia.

Glickman, Oren and Ido Dagan. 2004. Probabilistic textual entailment: Generic applied modeling of language variability. In *Proceedings of the Workshop on Learning Methods for Text Understanding and Mining*, Grenoble, France.

Glickman, Oren, Ido Dagan, and Moshe Koppel. 2005. Web based probabilistic textual entailment. In *Proceedings of the 1st Pascal Challenge Workshop*, Southampton, UK.

Haasdonk, Bernard. 2005. Feature space interpretation of SVMs with indefinite kernels. *IEEE Trans Pattern Anal Mach Intell*, 27(4):482–92, Apr.

Joachims, Thorsten. 1999. Making large-scale svm learning practical. In Schlkopf, B., C. Burges, and A. Smola, editors, *Advances in Kernel Methods-Support Vector Learning*. MIT Press.

Miller, George A. 1995. WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41, November.

Moschitti, Alessandro and Cosmin Adrian Bejan. 2004. A semantic kernel for predicate argument classification. In *CoNLL-2004*, USA.

Moschitti, A. and F. Zanzotto. 2007. Fast and effective kernels for relational learning from texts. In Ghahramani, Zoubin, editor, *Proceedings of the 24th Annual International Conference on Machine Learning (ICML 2007)*.

Moschitti, Alessandro, Daniele Pighin, and Roberto Basili. 2006. Semantic Role Labeling via Tree Kernel Joint Inference. In *Proceedings of CoNLL-X*.

Moschitti, Alessandro, Silvia Quarteroni, Roberto Basili, and Suresh Manandhar. 2007. Exploiting syntactic and shallow semantic kernels for question answer classification. In *Proceedings ACL*, Prague, Czech Republic.

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

Zanzotto, Fabio Massimo and Alessandro Moschitti. 2006. Automatic learning of textual entailments with cross-pair similarities. In *Proceedings of the 21st Coling and 44th ACL*, pages 401–408, Sydney, Australia, July.