Automatic learning of textual entailments with cross-pair similarities

Fabio Massimo Zanzotto
DISCo
University of Milano-Bicocca
Milan, Italy
zanzotto@disco.unimib.it

Alessandro Moschitti
Department of Computer Science
University of Rome "Tor Vergata"
Rome, Italy
moschitti@info.uniroma2.it

Abstract

In this paper we define a novel similarity measure between examples of textual entailments and we use it as a kernel function in Support Vector Machines (SVMs). This allows us to automatically learn the *rewrite rules* that describe a non trivial set of entailment cases. The experiments with the data sets of the RTE 2005 challenge show an improvement of 4.4% over the state-of-the-art methods.

1 Introduction

Recently, textual entailment recognition has been receiving a lot of attention. The main reason is that the understanding of the basic entailment processes will allow us to model more accurate semantic theories of natural languages (Chierchia and McConnell-Ginet, 2001) and design important applications (Dagan and Glickman, 2004), e.g., Question Answering and Information Extraction.

However, previous work (e.g., (Zaenen et al., 2005)) suggests that determining whether or not a text T entails a hypothesis H is quite complex even when all the needed information is explicitly asserted. For example, the sentence T_1: "*At the end of the year, all solid companies pay dividends.*" entails the hypothesis H_1: "*At the end of the year, all solid insurance companies pay dividends.*" but it does not entail the hypothesis H_2: "*At the end of the year, all solid companies pay cash dividends.*"

Although these implications are uncontroversial, their automatic recognition is complex if we rely on models based on lexical distance (or similarity) between hypothesis and text, e.g., (Corley and Mihalcea, 2005). Indeed, according to such approaches, the hypotheses H_1 and H_2 are very similar and seem to be similarly related to T_1. This suggests that we should study the properties and differences of such two examples (negative and positive) to derive more accurate entailment models. For example, if we consider the following entailment:

$T_3 \Rightarrow H_3$?	
T_3	*"All wild animals eat plants that have scientifically proven medicinal properties."*
H_3	*"All wild mountain animals eat plants that have scientifically proven medicinal properties."*

we note that T_3 is structurally (and somehow lexically similar) to T_1 and H_3 is more similar to H_1 than to H_2. Thus, from $T_1 \Rightarrow H_1$ we may extract rules to derive that $T_3 \Rightarrow H_3$.

The above example suggests that we should rely not only on a *intra-pair* similarity between T and H but also on a *cross-pair* similarity between two pairs (T', H') and (T'', H''). The latter similarity measure along with a set of annotated examples allows a learning algorithm to automatically derive syntactic and lexical rules that can solve complex entailment cases.

In this paper, we define a new cross-pair similarity measure based on text and hypothesis syntactic trees and we use such similarity with traditional *intra-pair* similarities to define a novel semantic kernel function. We experimented with such kernel using Support Vector Machines (Vapnik, 1995) on the test tests of the Recognizing Textual Entailment (RTE) challenges (Dagan et al., 2005; Bar Haim et al., 2006). The comparative results show that (a) we have designed an effective way to automatically learn entailment rules from examples and (b) our approach is highly accurate and exceeds the accuracy of the current state-of-the-art

Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL, pages 401–408,
Sydney, July 2006.

models (Glickman et al., 2005; Bayer et al., 2005) by about 4.4% (i.e. 63% vs. 58.6%) on the RTE 1 test set (Dagan et al., 2005).

In the remainder of this paper, Sec. 2 illustrates the related work, Sec. 3 introduces the complexity of learning entailments from examples, Sec. 4 describes our models, Sec. 6 shows the experimental results and finally Sec. 7 derives the conclusions.

2 Related work

Although the textual entailment recognition problem is not new, most of the automatic approaches have been proposed only recently. This has been mainly due to the RTE challenge events (Dagan et al., 2005; Bar Haim et al., 2006). In the following we report some of such researches.

A first class of methods defines measures of the distance or similarity between T and H either assuming the independence between words (Corley and Mihalcea, 2005; Glickman et al., 2005) in a bag-of-word fashion or exploiting syntactic interpretations (Kouylekov and Magnini, 2005). A pair (T, H) is then in entailment when $sim(T, H) > \alpha$. These approaches can hardly determine whether the entailment holds in the examples of the previous section. From the point of view of bag-of-word methods, the pairs (T_1, H_1) and (T_1, H_2) have both the same intra-pair similarity since the sentences of T_1 and H_1 as well as those of T_1 and H_2 differ by a noun, *insurance* and *cash*, respectively. At syntactic level, also, we cannot capture the required information as such nouns are both noun modifiers: *insurance* modifies *companies* and *cash* modifies *dividends*.

A second class of methods can give a solution to the previous problem. These methods generally combine a similarity measure with a set of possible transformations $\mathcal{T}$ applied over syntactic and semantic interpretations. The entailment between T and H is detected when there is a transformation $r \in \mathcal{T}$ so that $sim(r(T), H) > \alpha$. These transformations are logical rules in (Bos and Markert, 2005) or sequences of allowed *rewrite rules* in (de Salvo Braz et al., 2005). The disadvantage is that such rules have to be manually designed. Moreover, they generally model better positive implications than negative ones and they do not consider errors in syntactic parsing and semantic analysis.

3 Challenges in learning from examples

In the introductory section, we have shown that, to carry out automatic learning from examples, we need to define a cross-pair similarity measure. Its definition is not straightforward as it should detect whether two pairs (T', H') and (T'', H'') realize the same *rewrite rules*. This measure should consider pairs similar when: (1) T' and H' are structurally similar to T'' and H'', respectively and (2) the lexical relations within the pair (T', H') are compatible with those in (T'', H''). Typically, T and H show a certain degree of overlapping, thus, lexical relations (e.g., between the same words) determine *word movements* from T to H (or vice versa). This is important to model the syntactic/lexical similarity between example pairs. Indeed, if we encode such movements in the syntactic parse trees of texts and hypotheses, we can use interesting similarity measures defined for syntactic parsing, e.g., the tree kernel devised in (Collins and Duffy, 2002).

To consider structural and lexical relation similarity, we augment syntactic trees with *placeholders* which identify linked words. More in detail:
- We detect links between words w_t in T that are equal, similar, or semantically dependent on words w_h in H. We call *anchors* the pairs (w_t, w_h) and we associate them with *placeholders*. For example, in Fig. 1, the placeholder 2" indicates the *(companies,companies)* anchor between T_1 and H_1. This allows us to derive the word movements between text and hypothesis.
- We align the trees of the two texts T' and T'' as well as the tree of the two hypotheses H' and H'' by considering the *word movements*. We find a correct mapping between placeholders of the two hypothesis H' and H'' and apply it to the tree of H'' to substitute its placeholders. The same mapping is used to substitute the placeholders in T''. This mapping should maximize the structural *similarity* between the four trees by considering that placeholders augment the node labels. Hence, the cross-pair similarity computation is reduced to the tree similarity computation.

The above steps define an effective cross-pair similarity that can be applied to the example in Fig. 1: T_1 and T_3 share the subtree in bold starting with **S** $\rightarrow$ **NP VP**. The lexicals in T_3 and H_3 are quite different from those T_1 and H_1, but we can rely on the structural properties expressed by their bold subtrees. These are more similar to the subtrees of T_1 and H_1 than those of T_1 and H_2, respectively. Indeed, H_1 and H_3 share the production **NP** $\rightarrow$ **DT JJ NN NNS** while H_2 and H_3 do

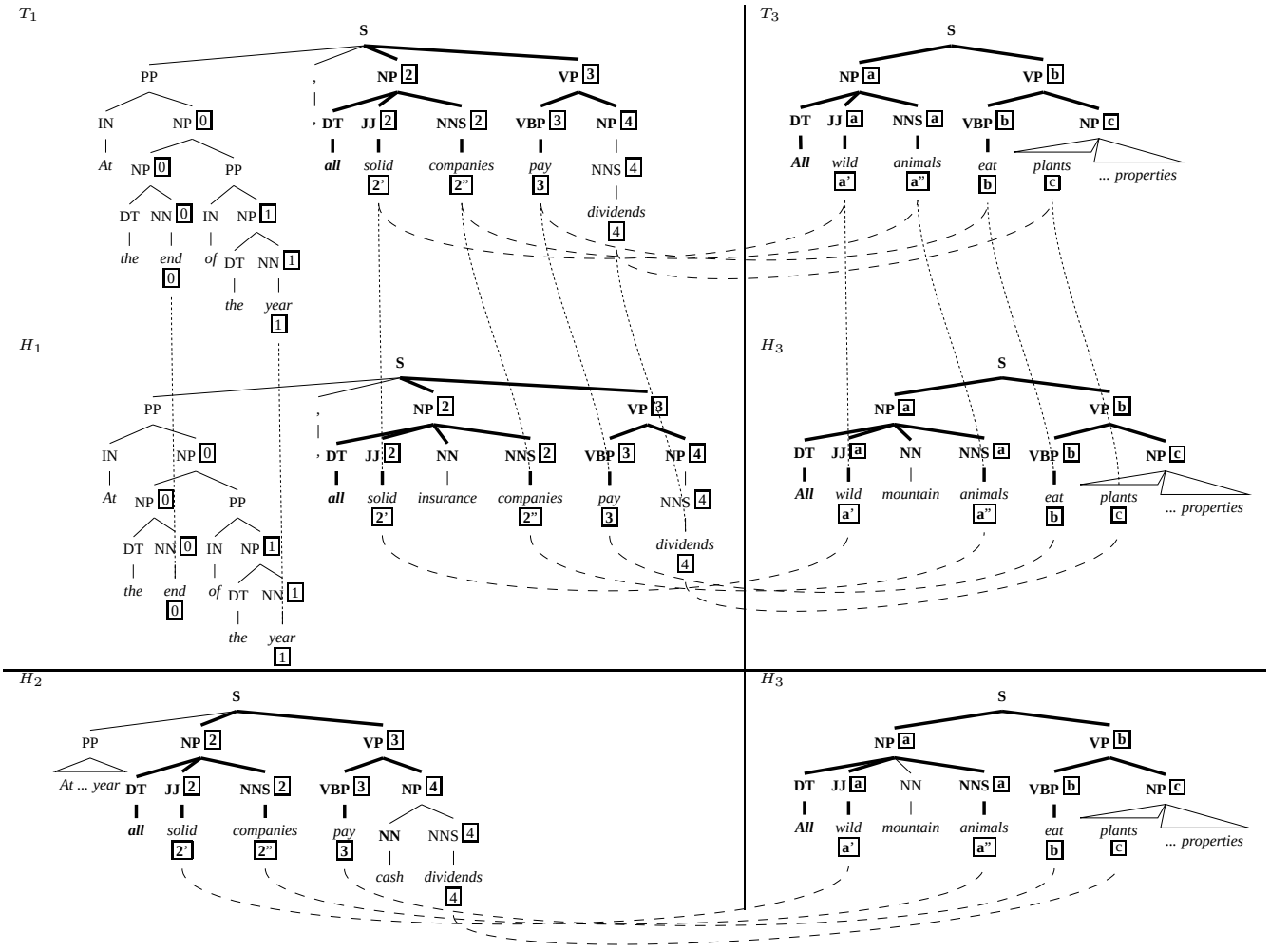

Figure 1: Relations between (T_1, H_1), (T_1, H_2), and (T_3, H_3).

not. Consequently, to decide if (T_3,H_3) is a valid entailment, we should rely on the decision made for (T_1, H_1). Note also that the dashed lines connecting placeholders of two texts (hypotheses) indicate structurally equivalent nodes. For instance, the dashed line between 3 and b links the main verbs both in the texts T_1 and T_3 and in the hypotheses H_1 and H_3. After substituting 3 with b and 2 with a, we can detect if T_1 and T_3 share the bold subtree **S** $\rightarrow$ **NP**2 **VP**3. As such subtree is shared also by H_1 and H_3, the words within the pair (T_1, H_1) are correlated similarly to the words in (T_3, H_3).

The above example emphasizes that we need to derive the *best* mapping between placeholder sets. It can be obtained as follows: let A' and A'' be the placeholders of (T', H') and (T'', H''), 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.

More precisely, let C be the set of all bijective mappings from $a' \subseteq A' : |a'| = |A''|$ to A'', an element $c \in C$ is a substitution function. We define as the best alignment the one determined by

$$c_{max} = argmax_{c \in C}(K_T(t(H', c), t(H'', i)) + K_T(t(T', c), t(T'', i)) \quad (1)$$

where (a) $t(S, c)$ returns the syntactic tree of the hypothesis (text) S with placeholders replaced by means of the substitution c, (b) i is the identity substitution and (c) $K_T(t_1, t_2)$ is a function that measures the similarity between the two trees t_1 and t_2 (for more details see Sec. 4.2). For example, the c_{max} between (T_1, H_1) and (T_3, H_3) is $\{(2', a'), (2'', a''), (3, b), (4, c)\}$.

4 Similarity Models

In this section we describe how anchors are found at the level of a single pair (T, H) (Sec. 4.1). The anchoring process gives the direct possibility of

implementing an inter-pair similarity that can be used as a baseline approach or in combination with the cross-pair similarity. This latter will be implemented with tree kernel functions over syntactic structures (Sec. 4.2).

4.1 Anchoring and Lexical Similarity

The algorithm that we design to find the anchors is based on similarity functions between words or more complex expressions. Our approach is in line with many other researches (e.g., (Corley and Mihalcea, 2005; Glickman et al., 2005)).

Given the set of content words (verbs, nouns, adjectives, and adverbs) W_T and W_H of the two sentences T and H, respectively, the set of anchors $A \subset W_T \times W_H$ is built using a similarity measure between two words $sim_w(w_t, w_h)$. Each element $w_h \in W_H$ will be part of a pair $(w_t, w_h) \in A$ if:
1) $sim_w(w_t, w_h) \neq 0$
2) $sim_w(w_t, w_h) = \max_{w'_t \in W_T} sim_w(w'_t, w_h)$
According to these properties, elements in W_H can participate in more than one anchor and conversely more than one element in W_H can be linked to a single element $w \in W_T$.

The similarity $sim_w(w_t, w_h)$ can be defined using different indicators and resources. First of all, two words are maximally similar if these have the same surface form $w_t = w_h$. Second, we can use one of the WordNet (Miller, 1995) similarities indicated with $d(l_w, l_{w'})$ (in line with what was done in (Corley and Mihalcea, 2005)) and different relation between words such as the lexical entailment between verbs (Ent) and derivationally relation between words (Der). Finally, we use the edit distance measure $lev(w_t, w_h)$ to capture the similarity between words that are missed by the previous analysis for misspelling errors or for the lack of derivationally forms not coded in WordNet.

As result, given the syntactic category $c_w \in \{noun, verb, adjective, adverb\}$ and the lemmatized form l_w of a word w, the similarity measure between two words w and w' is defined as follows:

$$sim_w(w, w') = \begin{cases} 1 & \textbf{if } w = w' \vee \\ & l_w = l_{w'} \wedge c_w = c_{w'} \vee \\ & ((l_w, c_w), (l_{w'}, c_{w'})) \in Ent \vee \\ & ((l_w, c_w), (l_{w'}, c_{w'})) \in Der \vee \\ & lev(w, w') = 1 \\ d(l_w, l_{w'}) & \textbf{if } c_w = c_{w'} \wedge d(l_w, l_{w'}) > 0.2 \\ 0 & \textbf{otherwise} \end{cases} \quad (2)$$

It is worth noticing that, the above measure is not a *pure* similarity measure as it includes the entailment relation that does not represent synonymy or similarity between verbs. To emphasize the contribution of each used resource, in the experimental section, we will compare Eq. 2 with some versions that exclude some word relations.

The above word similarity measure can be used to compute the similarity between T and H. In line with (Corley and Mihalcea, 2005), we define it as:

$$s_1(T, H) = \frac{\sum_{(w_t, w_h) \in A} sim_w(w_t, w_h) \times idf(w_h)}{\sum_{w_h \in W_H} idf(w_h)} \quad (3)$$

where $idf(w)$ is the inverse document frequency of the word w. For sake of comparison, we consider also the corresponding more classical version that does not apply the inverse document frequency

$$s_2(T, H) = \sum_{(w_t, w_h) \in A} sim_w(w_t, w_h)/|W_H| \quad (4)$$

¿From the above intra-pair similarities, s_1 and s_2, we can obtain the baseline *cross-pair* similarities based on only lexical information:

$$K_i((T', H'), (T'', H'')) = s_i(T', H') \times s_i(T'', H''), \quad (5)$$

where $i \in \{1, 2\}$. In the next section we define a novel cross-pair similarity that takes into account syntactic evidence by means of tree kernel functions.

4.2 Cross-pair syntactic kernels

Section 3 has shown that to measure the syntactic similarity between two pairs, (T', H') and (T'', H''), we should capture the number of common subtrees between texts and hypotheses that share the same anchoring scheme. The best alignment between anchor sets, i.e. the best substitution c_{max}, can be found with Eq. 1. As the corresponding maximum quantifies the *alignment degree*, we could define a cross-pair similarity as follows:

$$K_s((T', H'), (T'', H'')) = \max_{c \in C} \Big(K_T(t(H', c), t(H'', i)) + K_T(t(T', c), t(T'', i)) \Big), \quad (6)$$

where as $K_T(t_1, t_2)$ we use the tree kernel function defined in (Collins and Duffy, 2002). This evaluates the number of subtrees shared by t_1 and t_2, thus defining an implicit substructure space.

Formally, given a subtree space $\mathcal{F} = \{f_1, f_2, \ldots, f_{|\mathcal{F}|}\}$, the indicator function $I_i(n)$ is equal to 1 if the target f_i is rooted at node n and equal to 0 otherwise. A tree-kernel function over t_1 and t_2 is $K_T(t_1, t_2) = \sum_{n_1 \in N_{t_1}} \sum_{n_2 \in N_{t_2}} \Delta(n_1, n_2)$, 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. As described in (Collins and Duffy, 2002), Δ can be computed in $O(|N_{t_1}| \times |N_{t_2}|)$.

The K_T function has been proven to be a valid kernel, i.e. its associated *Gram* matrix is positive-semidefinite. Some basic operations on kernel functions, e.g. the sum, are closed with respect to the set of valid kernels. Thus, if the maximum held such property, Eq. 6 would be a valid kernel and we could use it in kernel based machines like SVMs. Unfortunately, a counterexample illustrated in (Boughorbel et al., 2004) shows that the *max* function does not produce valid kernels in general.

However, we observe that: (1) $K_s((T', H'), (T'', H''))$ is a symmetric function since the set of transformation C are always computed with respect to the pair that has the largest anchor set; (2) in (Haasdonk, 2005), it is shown that when kernel functions are not positive semidefinite, SVMs still solve a data separation problem in pseudo Euclidean spaces. The drawback is that the solution may be only a local optimum. Therefore, we can experiment Eq. 6 with SVMs and observe if the empirical results are satisfactory. Section 6 shows that the solutions found by Eq. 6 produce accuracy higher than those evaluated on previous automatic textual entailment recognition approaches.

5 Refining cross-pair syntactic similarity

In the previous section we have defined the intra and the cross pair similarity. The former does not show relevant implementation issues whereas the latter should be optimized to favor its applicability with SVMs. The Eq. 6 improvement depends on three factors: (1) its computation complexity; (2) a correct marking of tree nodes with placeholders; and, (3) the pruning of irrelevant information in large syntactic trees.

5.1 Controlling the computational cost

The computational cost of cross-pair similarity between two tree pairs (Eq. 6) depends on the size of C. This is combinatorial in the size of A' and A'', i.e. $|C| = (|A'| - |A''|)!|A''|!$ if $|A'| \geq |A''|$. Thus we should keep the sizes of A' and A'' reasonably small.

To reduce the number of placeholders, we consider the notion of *chunk* defined in (Abney, 1996), i.e., *not recursive kernels* of noun, verb, adjective, and adverb phrases. When placeholders are in a single chunk both in the text and hypothesis we assign them the same name. For example, Fig. 1 shows the placeholders 2' and 2" that are substituted by the placeholder 2. The placeholder reduction procedure also gives the possibility of resolving the ambiguity still present in the anchor set A (see Sec. 4.1). A way to eliminate the ambiguous anchors is to select the ones that reduce the final number of placeholders.

5.2 Augmenting tree nodes with placeholders

Anchors are mainly used to extract relevant syntactic subtrees between pairs of text and hypothesis. We also use them to characterize the syntactic information expressed by such subtrees. Indeed, Eq. 6 depends on the number of common subtrees between two pairs. Such subtrees are matched when they have the same node labels. Thus, to keep track of the argument movements, we augment the node labels with placeholders. The larger number of placeholders two hypotheses (texts) match the larger the number of their common substructures is (i.e. higher similarity). Thus, it is really important where placeholders are inserted.

For example, the sentences in the pair (T_1, H_1) have related subjects 2 and related main verbs 3. The same occurs in the sentences of the pair (T_3, H_3), respectively a and b. To obtain such node marking, the placeholders are propagated in the syntactic tree, from the leaves[1] to the target nodes according to the head of constituents. The example of Fig. 1 shows that the placeholder 0 climbs up to the node governing all the NPs.

5.3 Pruning irrelevant information in large *text* trees

Often only a portion of the parse trees is relevant to detect entailments. For instance, let us consider the following pair from the RTE 2005 corpus:

[1]To increase the generalization capacity of the tree kernel function we choose not to assign any placeholder to the leaves.

$T \Rightarrow H$ (id: 929)	
T	***"Ron Gainsford, chief executive of the TSI,*** *said: "It is a major concern to us that parents could be unwittingly exposing their children to the risk of sun damage, thinking they are better protected than they actually are."*
H	*"Ron Gainsford is the chief executive of the TSI."*

Only the bold part of T supports the implication; the rest is useless and also misleading: if we used it to compute the similarity it would reduce the importance of the relevant part. Moreover, as we normalize the syntactic tree kernel (K_T) with respect to the size of the two trees, we need to focus only on the part relevant to the implication.

The anchored leaves are good indicators of relevant parts but also some other parts may be very relevant. For example, the function word *not* plays an important role. Another example is given by the word *insurance* in H_1 and *mountain* in H_3 (see Fig. 1). They support the implication $T_1 \Rightarrow H_1$ and $T_1 \Rightarrow H_3$ as well as *cash* supports $T_1 \not\Rightarrow H_2$. By removing these words and the related structures, we cannot determine the correct implications of the first two and the incorrect implication of the second one. Thus, we keep all the words that are immediately related to relevant constituents.

The reduction procedure can be formally expressed as follows: given a syntactic tree t, the set of its nodes $N(t)$, and a set of anchors, we build a tree t' with all the nodes N' that are anchors or ancestors of any anchor. Moreover, we add to t' the leaf nodes of the original tree t that are direct children of the nodes in N'. We apply such procedure only to the syntactic trees of texts before the computation of the kernel function.

6 Experimental investigation

The aim of the experiments is twofold: we show that (a) entailment recognition rules can be learned from examples and (b) our kernel functions over syntactic structures are effective to derive syntactic properties. The above goals can be achieved by comparing the different intra and cross pair similarity measures.

6.1 Experimental settings

For the experiments, we used the Recognizing Textual Entailment Challenge data sets, which we name as follows:

- $D1$, $T1$ and $D2$, $T2$, are the development and the test sets of the first (Dagan et al., 2005) and second (Bar Haim et al., 2006) challenges, respectively. $D1$ contains 567 examples whereas $T1$, $D2$ and $T2$ have all the same size, i.e. 800 training/testing instances. The positive examples constitute the 50% of the data.
- ALL is the union of $D1$, $D2$, and $T1$, which we also split in 70%-30%. This set is useful to test if we can learn entailments from the data prepared in the two different challenges.
- $D2(50\%)'$ and $D2(50\%)''$ is a random split of $D2$. It is possible that the data sets of the two competitions are quite different thus we created this *homogeneous* split.

We also used the following resources:

- The Charniak parser (Charniak, 2000) and the `morpha` lemmatiser (Minnen et al., 2001) to carry out the syntactic and morphological analysis.
- WordNet 2.0 (Miller, 1995) to extract both the verbs in entailment, Ent set, and the derivationally related words, Der set.
- The `wn::similarity` package (Pedersen et al., 2004) to compute the Jiang&Conrath (J&C) distance (Jiang and Conrath, 1997) as in (Corley and Mihalcea, 2005). This is one of the best figure method which provides a similarity score in the $[0, 1]$ interval. We used it to implement the $d(l_w, l_{w'})$ function.
- A selected portion of the British National Corpus[2] to compute the inverse document frequency (idf). We assigned the maximum idf to words not found in the BNC.
- SVM-light-TK[3] (Moschitti, 2006) which encodes the basic tree kernel function, K_T, in SVM-light (Joachims, 1999). We used such software to implement K_s (Eq. 6), K_1, K_2 (Eq. 5) and $K_s + K_i$ kernels. The latter combines our new kernel with traditional approaches ($i \in \{1, 2\}$).

6.2 Results and analysis

Table 1 reports the results of different similarity kernels on the different training and test splits described in the previous section. The table is organized as follows:

The first 5 rows (*Experiment settings*) report the intra-pair similarity measures defined in Section 4.1, the 6th row refers to only the idf similarity metric whereas the following two rows report the cross-pair similarity carried out with Eq. 6 with (*Synt Trees with placeholders*) and without (*Only Synt Trees*) augmenting the trees with placeholders, respectively. Each column in the *Experiment*

[2] http://www.natcorp.ox.ac.uk/

[3] SVM-light-TK is available at `http://ai-nlp.info.uniroma2.it/moschitti/`

Experiment Settings								
$w = w' \vee l_w = l_{w'} \wedge c_w = c_{w'}$	√	√	√	√	√	√	√	√
$c_w = c_{w'} \wedge d(l_w, l_{w'}) > 0.2$			√	√	√	√	√	√
$((l_w, c_w), (l_{w'}, c_{w'})) \in Der$					√	√	√	√
$((l_w, c_w), (l_{w'}, c_{w'})) \in Ent$					√	√	√	√
$lev(w, w') = 1$						√	√	√
idf		√		√	√	√	√	√
Only Synt Trees							√	
Synt Trees with placeholders								√
Datasets								
"Train:$D1$-Test:$T1$"	0.5388	0.5813	0.5500	0.5788	0.5900	0.5888	0.6213	**0.6300**
"Train:$T1$-Test:$D1$"	0.5714	0.5538	0.5767	0.5450	0.5591	0.5644	0.5732	**0.5838**
"Train:$D2(50\%)'$-Test:$D2(50\%)''$"	0.6034	0.5961	0.6083	0.6010	0.6083	0.6083	0.6156	**0.6350**
"Train:$D2(50\%)''$-Test:$D2(50\%)'$"	0.6452	0.6375	0.6427	0.6350	0.6324	0.6272	0.5861	**0.6607**
"Train:$D2$-Test:$T2$"	0.6000	0.5950	0.6025	0.6050	0.6050	0.6038	0.6238	**0.6388**
Mean	0.5918 (± 0.0396)	0.5927 (± 0.0303)	0.5960 (± 0.0349)	0.5930 (± 0.0335)	0.5990 (± 0.0270)	0.5985 (± 0.0235)	0.6040 (± 0.0229)	**0.6297** (± 0.0282)
"Train:$ALL(70\%)$-Test:$ALL(30\%)$"	0.5902	0.6024	0.6009	-	0.6131	0.6193	0.6086	**0.6376**
"Train:ALL-Test:$T2$"	0.5863	0.5975	0.5975	0.6038	-	-	0.6213	**0.6250**

Table 1: Experimental results of the different methods over different test settings

settings indicates a different intra-pair similarity measure built by means of a combination of basic similarity approaches. These are specified with the check sign $\surd$. For example, Column 5 refers to a model using: the surface word form similarity, the $d(l_w, l_{w'})$ similarity and the idf.

The next 5 rows show the accuracy on the data sets and splits used for the experiments and the next row reports the average and Std. Dev. over the previous 5 results. Finally, the last two rows report the accuracy on ALL dataset split in 70/30% and on the whole ALL dataset used for training and T2 for testing.

¿From the table we note the following aspects:
- First, the lexical-based distance kernels K_1 and K_2 (Eq. 5) show accuracy significantly higher than the random baseline, i.e. 50%. In all the datasets (except for the first one), the $sim_w(T, H)$ similarity based on the lexical overlap (first column) provides an accuracy essentially similar to the best lexical-based distance method.
- Second, the dataset "Train:$D1$-Test:$T1$" allows us to compare our models with the ones of the first RTE challenge (Dagan et al., 2005). The accuracy reported for the best systems, i.e. 58.6% (Glickman et al., 2005; Bayer et al., 2005), is not significantly different from the result obtained with K_1 that uses the idf.
- Third, the dramatic improvement observed in (Corley and Mihalcea, 2005) on the dataset "Train:$D1$-Test:$T1$" is given by the idf rather than the use of the J&C similarity (second vs. third columns). The use of J&C with the idf decreases the accuracy of the idf alone.
- Next, our approach (last column) is significantly better than all the other methods as it provides the best result for each combination of training and test sets. On the "Train:$D1$-Test:$T1$" test set, it exceeds the accuracy of the current state-of-the-art models (Glickman et al., 2005; Bayer et al., 2005) by about 4.4 absolute percent points (63% vs. 58.6%) and 4% over our best lexical similarity measure. By comparing the average on all datasets, our system improves on all the methods by at least 3 absolute percent points.
- Finally, the accuracy produced by *Synt Trees with placeholders* is higher than the one obtained with *Only Synt Trees*. Thus, the use of placeholders is fundamental to automatically learn entailments from examples.

6.2.1 Qualitative analysis

Hereafter we show some instances selected from the first experiment "Train:$T1$-Test:$D1$". They were correctly classified by our overall model (last column) and miss-classified by the models in the seventh and in the eighth columns. The first is an example in entailment:

$T \Rightarrow H$	(id: 35)
T	*"Saudi Arabia, the biggest oil producer in the world, was once a supporter of Osama bin Laden and his associates who led attacks against the United States."*
H	*"Saudi Arabia is the world's biggest oil exporter."*

It was correctly classified by exploiting examples like these two:

$T \Rightarrow H$	(id: 929)
T	*"Ron Gainsford, chief executive of the TSI, said: ..."*
H	*"Ron Gainsford is the chief executive of the TSI."*

$T \Rightarrow H$	(id: 976)
T	*"Harvey Weinstein, the co-chairman of Miramax, who was instrumental in popularizing both independent and foreign films with broad audiences, agrees."*
H	*"Harvey Weinstein is the co-chairman of Miramax."*

The rewrite rule is: *"X, Y, ..."* implies *"X is Y"*. This rule is also described in (Hearst, 1992).

A more interesting rule relates the following two sentences which are not in entailment:

$T \nRightarrow H$	(id: 2045)
T	*"Mrs. Lane, who has been a Director since 1989, is Special Assistant to the Board of Trustees and to the President of Stanford University."*
H	*"Mrs. Lane is the president of Stanford University."*

It was correctly classified using instances like the following:

$T \nRightarrow H$	(id: 2044)
T	*"Jacqueline B. Wender is Assistant to the President of Stanford University."*
H	*"Jacqueline B. Wender is the President of Stanford University."*

$T \nRightarrow H$	(id: 2069)
T	*"Grieving father Christopher Yavelow hopes to deliver one million letters to the queen of Holland to bring his children home."*
H	*"Christopher Yavelow is the queen of Holland."*

Here, the implicit rule is: *"X (VP (V ...) (NP (to Y) ...)"* does not imply *"X is Y"*.

7 Conclusions

We have presented a model for the automatic learning of rewrite rules for textual entailments from examples. For this purpose, we devised a novel powerful kernel based on cross-pair similarities. We experimented with such kernel using Support Vector Machines on the RTE test sets. The results show that (1) learning entailments from positive and negative examples is a viable approach and (2) our model based on kernel methods is highly accurate and improves on the current state-of-the-art entailment systems.

In the future, we would like to study approaches to improve the computational complexity of our kernel function and to design approximated versions that are valid Mercer's kernels.

References

Steven Abney. 1996. Part-of-speech tagging and partial parsing. In G.Bloothooft K.Church, S.Young, editor, *Corpus-based methods in language and speech*. Kluwer academic publishers, Dordrecht.

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

Samuel Bayer, John Burger, Lisa Ferro, John Henderson, and Alexander Yeh. 2005. MITRE's submissions to the eu PASCAL RTE challenge. In *Proceedings of the 1st RTE Workshop*, Southampton, UK.

Johan Bos and Katja Markert. 2005. Recognising textual entailment with logical inference. In *Proc. of HLT-EMNLP Conference*, Canada.

S. Boughorbel, J-P. Tarel, and F. Fleuret. 2004. Non-mercer kernel for svm object recognition. In *Proceedings of BMVC 2004*.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proc. of the 1st NAACL*,Seattle, Washington.

Gennaro Chierchia and Sally McConnell-Ginet. 2001. *Meaning and Grammar: An introduction to Semantics*. MIT press, Cambridge, MA.

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

Courtney Corley and Rada Mihalcea. 2005. Measuring the semantic similarity of texts. In *Proc. of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, Ann Arbor, Michigan.

Ido Dagan and Oren Glickman. 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.

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

Rodrigo de Salvo Braz, Roxana Girju, Vasin Punyakanok, Dan Roth, and Mark Sammons. 2005. An inference model for semantic entailment in natural language. In *Proc. of the RTE Workshop*, Southampton, U.K.

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

Bernard Haasdonk. 2005. Feature space interpretation of SVMs with indefinite kernels. *IEEE Trans Pattern Anal Mach Intell*.

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proc. of the 15th CoLing*, Nantes, France.

Jay J. Jiang and David W. Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proc. of the 10th ROCLING*, Tapei, Taiwan.

Thorsten Joachims. 1999. Making large-scale svm learning practical. In *Advances in Kernel Methods-Support Vector Learning*. MIT Press.

Milen Kouylekov and Bernardo Magnini. 2005. Tree edit distance for textual entailment. In *Proc. of the RANLP-2005*, Borovets, Bulgaria.

George A. Miller. 1995. WordNet: A lexical database for English. *Communications of the ACM*, November.

Guido Minnen, John Carroll, and Darren Pearce. 2001. Applied morphological processing of English. *Natural Language Engineering*.

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

Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. 2004. Wordnet::similarity - measuring the relatedness of concepts. In *Proc. of 5th NAACL*, Boston, MA.

Vladimir Vapnik. 1995. *The Nature of Statistical Learning Theory*. Springer.

Annie Zaenen, Lauri Karttunen, and Richard Crouch. 2005. Local textual inference: Can it be defined or circumscribed? In *Proc. of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, Ann Arbor, Michigan.