Learning Shallow Semantic Rules for Textual Entailment

Marco Pennacchiotti[⋆] & Fabio Massimo Zanzotto[⋄]
[⋆] Computational Linguistics, Saarland University, Saarbrücken, Germany, *pennacchiotti@coli.uni-sb.de*
[⋄] DISP - University of Roma Tor Vergata, Roma, Italy, *zanzotto@info.uniroma2.it*

Abstract

In this paper we present a novel technique for integrating lexical-semantic knowledge in systems for learning textual entailment recognition rules: the *typed anchors*. These describe the semantic relations between words across an entailment pair. We integrate our approach in the *cross-pair similarity* model. Experimental results show that our approach increases performance of *cross-pair similarity* learning systems.

1 Introduction

The Recognizing Textual Entailment (RTE) task has recently received growing attention, as a means to computationally model textual inference in Natural Language Processing (NLP) applications. Formally, given a pair of text fragments, the *Text T* and the *Hypothesis H*, the goal of an RTE system is to recognize if T *entails* H. Textual entailment is a key component of many NLP applications. For example, consider a Question Answering system which has to answer the question: "*When did John Lennon died?*". The system could find the answer from the snippet "*In 1980 Chapman killed John Lennon*", by recognizing the following implication:

T_1	"*In 1980 Chapman killed John Lennon.*"	(E_1)
H_1	"*John Lennon died in 1980.*"	

In the last few years, RTE challenges [1] have been organized to compare the performance of different RTE systems over a common and balanced corpus of entailment pairs (T, H). Most strategies for RTE fall into these three categories: *lexical overlap* (e.g. [4]), *syntactic matching* (e.g. [14, 11, 9], *entailment triggering* (e.g. [15, 5, 6]). All these approaches are plausible and effective. Also, they are fairly complementary as they recognize different set of entailment pairs [2]. Yet, today it is still not clear which approach is most appropriate for RTE; so far, only few systems successfully integrated them in a common model (e.g. [10, 5]). This lack of integration is one of the reasons of the low recognition performance (the average accuracy at the RTE-2 challenge was 0.59).

Recently, an original machine learning approach for RTE has been proposed in [16]. Its aim is to integrate *lexical overlap* and *entailment triggering*, in order to leverage complementarity and boost performance. The key idea is a similarity between pairs of texts and hypotheses, the *cross-pair similarity*, that considers the relations between words in T and H. These relations are captured using *placeholders*. This allows the system to automatically exploit *rewrite rules*. Yet, the system suffers a major problem which highly limits its performance. Placeholders align two words if they are semantically similar, but the relation between them is not explicitly represented. This limitation can lead the learning algorithm to exploit erroneous rewrite rules.

In this paper, we present a novel method to solve the above mentioned limitation, by introducing the notion of *typed anchors*. The idea is to adopt placeholders with a semantic tag expressing the semantic relation standing between the lexicals. This intuition allows the system to exploit more semantically principled rewrite rules, which should avoid misclassifications and significantly improve performance. For example in the pair E_1, the learning algorithm would exploit the correct rule: *if the object of T aligns to the subject of H, and* ***the verbs are in causation relation****, then entailment holds.*

The paper is organized as follows. Sec. 2 reviews the cross-pair similarity model and analyzes its limits. In Sec. 3, we introduce our model for *typed anchors* aiming at integrating semantic information. Finally, in Sec. 4 we empirically assess that the use of typed anchors significantly outperforms approaches based on simple placeholders and approaches based on *lexical overlap*, *syntactic matching*, and *entailment triggering*.

2 Cross-pair similarity and its limits

In this section we firstly review the cross-pair similarity model used to exploit textual entailment recognition *rewrite rules*. We then analyze its limits observing how poorly defined relations among words may generate wrong rewrite rules.

2.1 Learning entailment rules with syntactic cross-pair similarity

The *cross-pair similarity* model [16] proposes a feature space of entailment pairs (T, H) where similarity-based learning model can exploit *rewrite rules* defined in training examples. The key idea is to define a *cross-pair similarity* $K_S((T', H'), (T'', H''))$ that takes into account relations among words within a pair. This is done using *placeholders*. A *placeholder* co-indexes two substructures in the parse trees of T and H, indicating that such substructures are related. At the word level (i.e. leaves) placeholders link pairs of words which are highly similar: these pairs are called *anchors*. For example, the sentence pair, "*All companies file annual* *reports*" implies "*All insurance companies file annual reports*", would be represented as follows:

T_2	(S (NP$\boxed{1}$ (DT All) (NNS$\boxed{1}$ companies)) (VP$\boxed{2}$ (VBP$\boxed{2}$ file) (NP$\boxed{3}$ (JJ$\boxed{3}$ annual) (NNS$\boxed{3}$ reports))))	(E_2)
H_2	(S (NP$\boxed{1}$ (DT All) (NNP Fortune) (CD 50) (NNS$\boxed{1}$ companies)) (VP$\boxed{2}$ (VBP$\boxed{2}$ file) (NP$\boxed{3}$ (JJ$\boxed{3}$ annual) (NNS$\boxed{3}$ reports))))	

where the placeholders $\boxed{1}$, $\boxed{2}$, and $\boxed{3}$ indicate the relations between the structures of T and those of H, and *companies/companies* is an example of anchor.

Placeholders help to determine if two pairs share the same *rewrite rule* by looking at the subtrees that they have in common. For example, suppose we have to determine if "*In autumn, all leaves fall*" implies "*In autumn, all maple leaves fall*". The related co-indexed representation is:

T_3	(S (PP (IN In) (NP (NN$\boxed{a}$ automn))) (, ,) (NP$\boxed{b}$ (DT all) (NNS$\boxed{b}$ leaves)) (VP$\boxed{c}$ (VBP$\boxed{c}$ fall)))	(E_3)
H_3	(S (PP (IN In) (NP$\boxed{a}$ (NN$\boxed{a}$ automn))) (, ,) (NP$\boxed{b}$ (DT all) (NN maple) (NNS$\boxed{a}$ leaves)) (VP$\boxed{c}$ (VBP$\boxed{c}$ fall)))	

E_2 and E_3 share the following subtrees:

T_4	(S (NP$\boxed{x}$ (DT all) (NNS$\boxed{x}$)) (VP$\boxed{y}$ (VBP$\boxed{y}$)))	(R_4)
H_4	(S (NP$\boxed{x}$ (DT all) (NN) (NNS$\boxed{x}$)) (VP$\boxed{x}$ (VBP$\boxed{x}$)))	

These subtrees represent the *rewrite rule* that E_2 and E_3 have in common. Then, E_3 can be likely classyfied as a valid entailment, as it shares the rule with the valid entailment E_2.

More details on the *cross-pair similarity* model can be found in [16] and an efficient algorithm for its computation is described in [13].

2.2 Limits of the syntactic cross-pair similarity

Learning from examples using cross-pair similarity is an attractive and effective approach, as results of the RTE-2 challenge show [1]. Yet, the cross-pair similarity strategy, as any machine learning approach, is highly sensitive on how the examples are represented in the feature space. An incorrect or inaccurate feature modelling can strongly bias the performance of the classifier.

This problem is even more evident in kernel-based methods, where the feature space is implicit, and the classifier can only rely on the syntactic structure of the examples. Then, as in the cross-pair similarity approach placeholders play an important role within the syntactic tree, the classifier can then be highly biased, if they convey incomplete or incorrect information.

Consider for example the following text-hypothesis pair, which can lead to an incorrect rule, if misused.

T_5	"*For my younger readers, Chapman killed John Lennon more than twenty years ago.*"	(E_5)
H_5	"*John Lennon died more than twenty years ago.*"	

In the basic cross-pair similarity model, the decision process can use rules like the following:

T_6	(S (NP:$\boxed{x}$) (VP:$\boxed{y}$ (VBD:$\boxed{y}$) (NP:$\boxed{z}$) (ADVP:$\boxed{k}$)))	(R_6)
H_6	(S (NP:$\boxed{z}$) (VP:$\boxed{y}$ (VBD:$\boxed{y}$) (ADVP:$\boxed{k}$)))	

where *kill* and *die* are anchored by the $\boxed{y}$ placeholder. This rule is useful to classify examples like:

T_7	"*Cows are vegetarian but, to save money on mass-production, farmers fed cows animal extracts.*"	(E_7)
H_7	"*Cows have eaten animal extracts.*"	

but it will clearly fail when used for:

T_8	"*FDA warns migraine medicine makers that they are illegally selling migraine medicines without federal approval.*"	(E_8)
H_8	"*Migraine medicine makers declared that their medicines have been approved.*"	

where *warn* and *declare* are anchored as generically similar verbs.

The limitation of the cross-pair similarity measure is then that placeholders do not convey the semantic knowledge needed in cases such as the above, where the semantic relation between connected verbs is essential.

3 Adding semantic information to cross-pair similarity

In the previous section we showed that the cross-pair similarity approach lacks the lexical-semantic knowledge for anchoring words. In the examples, the missed knowledge is the type of semantic relation between the main verbs. The relation that links *kill* and *die* is not a generic similarity, as a WordNet based similarity measure would suggest, but a more specific causal relation. The exploited rewrite rule R_6 holds only for verbs in such relation. It is correctly applied in example E_7, as *feed* causes *eat*. Yet, it gives a wrong suggestion in example E_8, as *warn* and *declare* are related by a generic similarity relation.

The type of relation that links two words (*anchor type*) seems to be mandatory, in order to exploit correct rules. The problem is then to encode this information in the syntactic trees along with the placeholders.

In this section we describe how we encode the anchor types in the syntactic trees, by using two models: the *typed anchor* (*ta*) and the *propagated typed anchor* (*tap*) models. As anchoring words of H with words in T is the basic step, before describing the models (Sec. 3.2), we shortly revise how anchors are selected and how they are encoded in the trees (Sec. 3.1).

3.1 Anchors and Placeholders

As many other approaches (e.g., [4]), our anchoring model is based on a similarity measure between words $sim_w(w_t, w_h)$. 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 highest similarity $sim_w(w_t, w_h)$. As result, we have a set of anchors $A \subset W_T \times W_H$ and the subset $W'_T \subseteq W_T$ of words in T connected with a word in H. In the second step, we select the final anchor set $A' \subseteq A$, as the bijective relation between W_H and W'_T that mostly satisfies a locality criterion: whenever possible, words of constituent in H should be related to words of a constituent in T. See [16] for more details on the adopted word similarity $sim_w(w_t, w_h)$.

Once the set A' is found, anchors are encoded in the syntactic trees with placeholders. Placeholders are put on the pre-terminal nodes of the anchored words. Then, they climb up in the tree according to this rule: *constituent nodes in the syntactic trees take the placeholder of their semantic heads*. This latter step guarantees that any subtree has the relational information. The final tree explicitly indicates how T relates to H using co-indexing (see E_2).

3.2 Typing anchors and placeholders

Our goal is to augment the co-indexed syntactic trees with typed anchors. To do that, we first have to decide what type of semantic relations we want to represent in the typed anchors (Sec. 3.2.1). Then, we need to define how to encode this information in the syntactic trees (Sec. 3.2.2).

3.2.1 Defining anchor types

The idea of introducing anchor types is in principle very simple. Yet, this may be not effective: attempts to introduce semantic information in RTE systems have often failed. A main reason for this failure is that any model using semantic information has the problem of dealing with ambiguity.

To investigate the validity of our idea, we then need to focus on a small set of relevant relation types. A valuable source of relation types among words is WordNet. We choose to integrate in our system three relations: *part-of*, *antinomy*, and *verb entailment*. This small set seems to be a correct choice, as it is relevant for many entailment cases such as those presented in Sec. 2.

We also define two more general anchor types: *similarity* and *surface matching*. The first type links words which are similar according to the WordNet similarity measure described in [7]. This type is intended to capture *synonymy* and *hyperonymy*. The second type is activated when words or lemmas match: it captures semantically equivalent words. The complete set of relation types used in the experiments is given in Table 1.

3.2.2 Augmenting placeholders with anchor types

Once anchor types have been defined, it is necessary to decide how to integrate their information in the syntactic trees. We apply a strategy similar to that adopted for placeholders, described in Sec. 3.1. However, the main problem is then to decide how the semantic information should be encoded. We experiment two possible models:

Rank	***Relation Type***	***Symbol***
1.	*antinomy*	$\leftrightarrow$
2.	*part-of*	$\subset$
3.	*verb entailment*	$\leftarrow$
4.	*similarity*	$\approx$
5.	*surface matching*	$=$

Table 1: *Ranked anchor types*

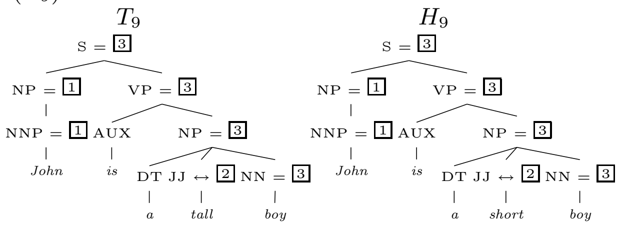
typed anchor model (ta) : anchor types augment only the pre-terminal nodes of the syntactic tree;

propagated typed anchor model (tap) : anchors climb up in the syntactic tree according to some specific *climbing-up rules*, similarly to what done for placeholders.

The **ta model** is easy to implement: typed anchors simply augment the pre-terminals of anchored words. The **tap model** is apparently more suitable for our purpose. The anchor type information is repeated in several tree fragments. As tree fragments are compared in the cross-pair similarity, this guarantees that the information is used in the decision process.

Unfortunately, the *tap* model is more complex, as it depends on strategy adopted for the anchor type climbing-up. The strategy must account for how anchors that climb up to the same node should interact. We implement our strategy by using *climbing-up rules*, as done in the case of placeholders. Yet, in this case, rules must consider the semantic information of the typed anchors. The choice of correct climbing-up rules is critical, as an incorrect rule could alter completely the semantics of the tree. In the case of placeholders, the climbing-up rule states that a constituent in the syntactic tree takes the placeholder of its semantic head. It is easy to demonstrate that in the case of typed anchors this rule would have disastrous effects. For example, consider the following false entailment pair:

(E_9)



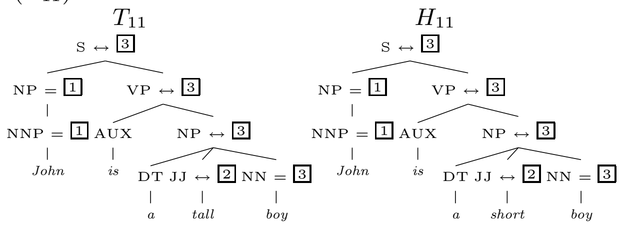
In the example, we apply the abovementioned rule: the typed anchor $=\boxed{3}$ climbs up to the pre-terminal node NP, instead of the typed anchor $\leftrightarrow\boxed{2}$, as it is the head of the constituent. If modelled in this way, this false entailment pair could generate, among others, the incorrect rewrite rule:

T_{10}	(S=$\boxed{3}$ (NP=$\boxed{1}$) (VP=$\boxed{3}$ (AUX is) (NP=$\boxed{2}$)))	(R_{10})
H_{10}	(S=$\boxed{3}$ (NP=$\boxed{1}$) (VP=$\boxed{3}$ (AUX is) (NP=$\boxed{2}$)))	

which states: *if two fragment have the same syntactic structure $\underline{S}(NP, VP(AUX, NP))$, and there is a semantic equivalence (=) on all constituents, then entailment does not hold.* This rule is wrong, as in that case entailment would hold (as all substructures are semantically equivalent).

The problem is that the wrong typed anchor climbed up the tree: we need the antonym anchor on the adjective (*tall/short*) to climb up, instead of the matching anchor on the noun (*boy/boy*), in order to exploit a correct rule. Our strategy must then implement a climbing-up rule producing these trees:

(E_{11})



In this case the pair generates correct rewrite rules, such as:

T_{12}	(S↔$\boxed{3}$ (NP=$\boxed{1}$) (VP↔$\boxed{3}$ (AUX is) (NP↔$\boxed{2}$)))
H_{12}	(S↔$\boxed{3}$ (NP=$\boxed{1}$) (VP↔$\boxed{3}$ (AUX is) (NP↔$\boxed{2}$)))

(R_{12})

The rule states: *if two fragment have the same syntactic structure $\underline{S}(NP_1, VP(AUX, NP_2))$, and there is an antonym type (↔) on the S and NP_2 , then entailment does not hold.*

The above example shows that the anchor type that has to climb up depends on the structure of the constituents. This can lead to a very complex model. Luckily, this intuition can be also captured by a simpler approximation. Instead of having climbing-up rules for each constituent type, we can rely on a ranking of the anchor types (as the one reported in Tab. 1). The anchor type that climbs up is the one that has an higher rank. In the example, this strategy produces the correct solution, as *antinomy* has an higher rank than *surface match*. We then implement in our model the following climbing-up rule: *if two typed anchors climb up to the same node, give precedence to that with the highest ranking in the ordered set of types $\mathcal{T} = (\leftrightarrow, \subset, \leftarrow, \approx, =)$.* Our ordered set $\mathcal{T}$ is consistent with common sense intuitions. In the next section we will empirically demonstrate its validity by reporting experiment evidences.

4 Experimental Results

In this section, we present empirical evidence to support the claims of the paper. In particular, we compare our **ta** and **tap** approaches with the strategies for RTE: *lexical overlap*, *syntactic matching* and *entailment triggering*. To perform the comparison we implemented these strategies in our machine learning platform for RTE, which also allows to combine them in more complex configurations.

4.1 Experimental Setup

For our evaluation we use the same methodology adopted at the RTE challenges [1]. The RTE task is to classify a test set of entailment pairs as true or false entailment, by relying on an annotated development set. Systems are evaluated on their prediction accuracy. We here adopt 4-fold cross validation, to obtain more reliable evidences. We use SVM-light-TK [12] as learning algorithm, which encodes the needed tree kernel functions in SVM-light [8].

We perform our experiments using the RTE-2 dataset, composed of 1600 entailment pairs from the RTE-2 challenge (800 true and 800 false entailment).

We evaluate *ta* and *tap* by comparing the performance of SVM with feature sets representing different *basic approaches*. We also experiment more complex feature spaces, representing *combined approaches*:

tree : the standard cross-similarity model described in Sec.2. Its comparison with *ta* and *tap* indicates the effectiveness of our approaches;

lex : a standard approach based on *lexical overlap*. The classifier uses as only feature the lexical overlap similarity score described in [4];

synt : a standard approach based on *syntactic matching*. The classifier uses as only feature a syntactic similarity score. A syntactic similarity measure $synt(T, H)$ is used to compute the score, by comparing all the substructures of the dependency trees of T and H, in line with approaches like [14, 11, 9]. This syntactic similarity is derived using the tree kernel similarity K_T [3] as follows: $synt(T, H) = K_T(T, H)/|H|$ where $|H|$ is the number of subtrees in H;

lex+ta , lex+tap : these configurations mix lexical overlap and our typed anchor approaches;

lex+tree : the comparison of this configuration with *lex+ta* and *lex+tap* should further support the validity of our intuition on typed anchors;

lex+synt : by comparing this configuration with *lex* and *synt* we aim at verifying if lexical and syntactic methods are complementary, as reported in [2];

lex+trig : this configuration mixes lexical overlap with basic entailment triggering features like in [15, 5, 6]. We use the following features: 1) *SVO* that tests if T and H share a similar subj-verb-obj construct; 2) *Apposition* that tests if H is a sentence headed by the verb *to be* and in T there is an apposition that states *H*; 3) *Anaphora* that tests if the SVO sentence in H has a similar wh-sentence in T and the wh-pronoun may be resolved in T with a word similar to the object or the subject of H.

4.2 Results Analysis

Table 2 reports the 4-folds and overall accuracy of the different feature spaces. The left part of the table shows the performance of the *basic approaches*, while the right those of the *combined approaches*.

	ta	*tap*	*tree*	*lex*	*synt*	$lex + ta$	$lex + tap$	$lex + tree$	$lex + synt$	$lex + trig$
Mean	61.29	62.47	61.35	61.81	58.28	63.94	63.81	63.68	61.94	61.56
Std dev	± 2.54	± 2.68	± 2.32	± 1.74	± 2.48	± 1.59	± 1.24	± 1.59	± 1.65	± 2.03

Table 2: *4-folds accuracy using different feature sets over the RTE-2 dataset.*

Results for the **basic approaches** show that *tap* outperforms all the other feature sets[1]. In particular, it guarantees an improvement of +1.12% accuracy over *tree*, suggesting that the addition of typed anchors to the basic cross-pair similarity model is indeed successful. This demonstrates that syntax is not enough, and that lexical-semantic knowledge, and in particular the explicit representation of word level relations, plays a key role in RTE. This is even more evident by comparing results to the pure syntactic approach *synt*, that achieves only 58.28% accuracy.

Also, *tap* outperforms *lex*, supporting a complementary conclusion: lexical-semantic knowledge does not cover alone the entailment phenomenon, but needs some syntactic evidence.

An overall analysis of basic systems further substantiate our intuition: approaches mixing syntax and lexical knowledge (*tap*) outperform method based on lexical knowledge (*lex*), which in turn outperform syntactic methods with weak lexical knowledge (*tree*) and pure syntactic methods (*synt*).

The surprisingly low performance of *ta* reveal that encoding typed anchors only at the pre-terminal level is not a sufficiently strong information for the learning algorithm. This further suggests the intuition the the semantics of word relations is indeed central.

Results for **combined approaches** reveals the difficulty of integrating lexical and syntactic information. The $lex + synt$ model does not substantially improves over *lex*. This suggests that a trivial integration of lexical overlap and syntactic matching between T and H is not effective. On the contrary, the use of cross-pair similarity together with lexical overlap ($lex + tree$) is successful, as accuracy improves +1.87% and +2.33% over the related basic methods (respectively *lex* and *tree*). The conclusion is then that cross-pair information across different pairs (in form of *rewrite rules*) and lexical information inside each pair are indeed both relevant. Again, our method mixed with *lex* achieve the best performance, further supporting the usefulness of typed anchors.

In general, our results also empirically confirm the manual analysis on the RTE-2 dataset performed in [2], suggesting that lexical and syntactic level are complementary for RTE, i.e. they recognize different set of entailment pairs.

5 Conclusions

Effectively integrating semantic knowledge in textual entailment recognition systems is one of the major problem in the area. In this paper we presented a simple but effetive model to integrate lexical semantic knowledge in a learner of rewrite rules for detecting textual entailment. Experimental results show that this is a promising model that may be used to integrate more complex semantic information.

[1] According to the sign-test, *tap* outperforms with more than 90% of statistical significance all the other basic approaches except the *lex*. In this case, the statistical significance is lower.

References

[1] R. Bar-Haim, I. Dagan, B. Dolan, L. Ferro, D. Giampiccolo, and I. Magnini, Bernardo Szpektor. The second pascal recognising textual entailment challenge. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, Venice, Italy, 2006.

[2] R. Bar-Haim, I. Szpecktor, and O. Glickman. Definition and analysis of intermediate entailment levels. In *Proc. of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, 2005.

[3] M. Collins and N. Duffy. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of ACL02*, 2002.

[4] C. Corley and R. Mihalcea. Measuring the semantic similarity of texts. In *Proc. of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, 2005.

[5] A. Hickl, J. Williams, J. Bensley, K. Roberts, B. Rink, and Y. Shi. Recognizing textual entailment with lcc's groundhog system. In B. Magnini and I. Dagan, editors, *Proc. of the Second PASCAL RTE Challenge*, Venice, Italy, 2006.

[6] D. Inkpen, D. Kipp, and V. Nastase. Machine learning experiments for textual entailment. In B. Magnini and I. Dagan, editors, *Proc. of the Second PASCAL RTE Challenge*, Venice, Italy, 2006.

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

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

[9] S. Katrenko and P. Adriaans. Using maximal embedded syntactic subtrees for textual entailment recognition. In B. Magnini and I. Dagan, editors, *Proc. of the Second PASCAL RTE Challenge*, Venice, Italy, 2006.

[10] B. MacCartney, T. Grenager, M.-C. de Marneffe, D. Cer, and C. D. Manning. Learning to recognize features of valid textual entailments. In *Proc. of the HLT/NAACL*, 2006.

[11] E. Marsi, E. Krahmer, W. Bosma, and M. Theune. Normalized alignment of dependency trees for detecting textual entailment. In B. Magnini and I. Dagan, editors, *Proceedings of the Second PASCAL RTE Challenge*, Venice, Italy, 2006.

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

[13] A. Moschitti and F. M. Zanzotto. Fast and effective kernels for relational learning from texts. In *Proceedings of the International Conference of Machine Learning (ICML)*, Corvallis, Oregon, 2007.

[14] V. Rus. Dependency-based textual entailment. In *FLAIRS Conference*, pages 110–109, 2006.

[15] R. Snow, L. Vanderwende, and A. Menezes. Effectively using syntax for recognizing false entailment. In *Proc. of HLT/NAACL 2006*, New York, 2006.

[16] F. M. Zanzotto and A. Moschitti. Automatic learning of textual entailments with cross-pair similarities. In *Proceedings of the 21st Coling and 44th ACL*, pages 401–408, Sydney, Australia, July 2006.