SemKer: Syntactic/Semantic Kernels for Recognizing Textual Entailment

Yashar Mehdad[1,2], Alessandro Moschitti[1], and Fabio Massiomo Zanzotto[3]

[1] DISI, University of Trento, POVO (TN) - Italy
[2] FBK-IRST, POVO (TN) - Italy
[3] DISP, University of Rome "Tor Vergata", Roma, Italy
mehdad@fbk.eu, moschitti@disi.unitn.it, zanzotto@info.uniroma2.it

Abstract. In this paper we describe the SemKer system participating to the fifth Recognizing of Textual Entailment (RTE5) challenge. The major novelty with respect to the systems with which we participated to the previous challenges is the use of semantic knowledge based on Wikipedia. More specifically, we used it to enrich the similarity measure between pairs of text and hypothesis (i.e. the tree kernel for text and hypothesis pairs), with a lexical similarity (i.e. the similarity between the leaves of the trees. The results show the benefit of this added semantic information.

1 Introduction

Our previous model [1] used syntactic tree kernels to define similarity between pairs of text trees and pairs of hypothesis trees. In our RTE5 system we extend such model by means of a similarity measure based on Wikipedia. We applied such similarity between terms, using the Syntactic Semantic Tree Kernel (SSTK) [2], which encodes lexical similarity in the fragment (subtree) matching, which is typically carried out by tree kernel functions.

We show that this approach can have a significant positive impact in accuracy, as well as, a very good coverage. Moreover, our approach for the computation of the similarity function based on Wikipedia is faster than previous tools based on WordNet or other resources [3].

Furthermore, we also explored the relationship between previous kernels for RTE [4, 5] and the new approach [6].

This paper is structured as below. Section 2 introduces the Wikipedia semantic model and our approach in building the similarity matrix over Wikipedia. Section 3 describes our kernel methods in recognizing textual entailment based on the previous work and the new approach. Finally, Section 4 illustrates our performance in RTE5 and the results of ablation tests following the discussion about the results.

2 Wikipedia Semantic Tree Kernel

In this section we present the main component of our new kernel, i.e. a lexical similarity derived from Wikipedia. This is used inside the syntactic/semantic tree kernel defined in [2] to enhance the basic tree kernel functions.

2.1 Lexical Semantic Similarity based on Wikipedia

Previous research in computational linguistics has produced many effective lexical similarity measures based on many different resources or corpora. For example, WordNet [7] similarities, such as Resnik, Lin, Path, Lesk and hso, are widely used in many applications and for the definition of kernel functions, e.g. [8–10]. However, such resources are limited in coverage thus we decided to focus on a larger source of knowledge such as Wikipedia.

The English version of Wikipedia, as of 17 October 2009, contains 3,064,846 articles and 18,272,763 pages, for a total of approximately 4.4 GB across 609 million words. This provides the largest coverage knowledge resource developed by a community. Another attractive property of Wikipedia is the noticeable coverage of named entities, which further motivates the design of a similarity measure based on it.

In our approach for defining a proximity matrix, we consider semantically related those words which frequently co-occur in the same text. The core of our approach lies on LSI (Latent Semantic Indexing) over a large corpus. We used singular valued decomposition (SVD) to build the proximity matrix P from Wikipedia, represented by its term-by-document matrix D.

SVD decomposes D into three matrices $U\Sigma V'$, where U and V are orthogonal matrices whose columns are the eigenvectors of DD' and $D'D$ respectively, and Σ is the diagonal matrix containing the singular value of D [11].

Given such decomposition, P can be obtained as $U_k\Sigma_k$, where U_k is the matrix containing the first k columns of U and k is the dimensionality of the latent semantic space[4]. Finally we computed the term similarity using the cosine measure in the VSM.

For this goal we used the jLSI (java Latent Semantic Indexing) tool [12] to create a term-by-document matrix from the Wikipedia, after cleaning the unnecessary tags. Then, we decomposed the term-by-document matrix, truncated to 400 dimensions. Lastly, we estimated the cosine similarity between pairs extracted from the RTE5 development and test set.

It is worth mentioning that the similarity measure over Wikipedia, not only covers all the existing pairs, but its computation is also about 100 times faster than the construction of the similarity matrix based on the WordNet Similarity package [7].

[4] This is used efficiently to reduce the memory requirements while retains the information

2.2 Syntactic/Semantic Tree Kernel (SSTK)

Standard tree kernel functions, e.g. [13], measure the similarity of two trees in terms of the number of tree fragments (subtrees or substructures) that they have in common. One drawback of such functions is that two identical subtrees with different leaves do not match even if the leaves, i.e. the words, are synonyms. To overcome this problem the Syntactic Semantic Tree Kernel (SSTK) has been defined in [2, 14]. Hereafter, we report is definition.

Given two trees T_1 and T_2 and an indicator function $I_i(n)$, which determines whether the fragment f_i is rooted in node n, a SSTK is defined as:

$$\kappa_T(T_1, T_2) = \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \sum_{i,j=1}^{|\mathcal{F}|} I_i(n_1) I_j(n_2) \kappa(f_i, f_j),$$

where:

- N_{T_1} and N_{T_2} are the set of nodes of T_1 and T_2;
- $\mathcal{F}$ is the set of the space fragments;
- $\kappa(f_1, f_2) = comp(f_1, f_2) \prod_{t=1}^{nt(f_1)} \kappa_S(f_1(t), f_2(t))$;
- $comp(f_1, f_2)$ (compatible) is 1 if f_1 differs from f_2 only in the terminal nodes and is 0 otherwise, $nt(f_i)$ is the number of terminal nodes and $f_i(t)$ is the t-th terminal symbol of f_i (numbered from left to right).

Finally, κ_S is a similarity between the leaves, i.e. lexicals, which in our case is given by the proximity matrix P derived from Wikipedia.

3 Kernels for Textual Entailment Recognition

In this section, we describe the kernels we used for RTE5. In Sec. 3.1 we describe the Max Similarity kernel (MSK), presented in [1]. In Sec. 3.2 we describe an enhancement of such kernel, namely the placeholder kernel (PK), presented in [6]. Finally, in Sec. 3.3 we describe the combination of the Syntactic/Semantic Tree Kernel (SSTK) with the MSK.

3.1 Max Similarity Kernel (MSK)

In [1, 15, 5], we proposed a kernel that can compute the similarity in a feature space modeling first-order rules. The idea was to model two separated features the left-hand sides (LHS) and the right-hand sides (RHS) of rules. Yet, these include variables in it. The model is fully presented in [5]. The resulting kernel is:

$$K_p(\langle T, H\rangle, \langle T', H'\rangle) = max_{c \in C}(TK(t(T, c), t(T', i)) + TK(t(H, c), t(H', i)), \quad (1)$$

where

– C is the set of all bijective mappings from the placeholders (i.e., possible variables) of the pair $\langle T, H\rangle$ and the ones of the pair $\langle T', H'\rangle$ (an element $c \in C$ is a substitution function)
– $t(\cdot, c)$ returns the syntactic tree enriched with placeholders replaced by means of the substitution c
– $TK(\tau_1, \tau_2)$ is a tree kernel function (as described in [16])

3.2 Placeholder Kernel (PK)

In [4], we studied the relation of the previous kernel with a kernel that computes the similarity in a feature space where each feature is a first-order rule. The resulting kernel may be seen as:

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

where the sum has been replaced by the product. In [6], we give another and more efficient formulation of a kernel for the same feature space which does not requires the maximization process. The equation has the following aspect. If we can define C^* as the set of all intersections of constraints in C, i.e. $C^* = \{c(J)|J \in 2^{\{1,\ldots,|C|\}}\}$, we can rewrite the kernel as:

$$K(G_1, G_2) = \sum_{c\in C^*} K_S(\tau_1, \tau_2, c)K_S(\gamma_1, \gamma_2, c)N(c) \tag{2}$$

where

$$N(c) = \sum_{\substack{J\in 2^{\{1,\ldots,|C|\}} \\ c=c(J)}} (-1)^{|J|-1} \tag{3}$$

This is a valid kernel.

3.3 Semantic Boosting via SSTK

Both MSK and PK can be boosted by using SSTK in place of a standard tree kernel, i.e. in place of TK in Eq. 1 or K_S in Eq. 2, respectively. For RTE5 we could only experiment with MSK-SSTK combination.

4 Results

In this section, the settings of our main three runs and the obtained results are discussed. Moreover, a set of ablation tests is presented which describes the efficiency of different resources, in our approach, on the textual entailment task.

4.1 Experimental Setup

We submitted three runs in RTE5 challenge. In each run, the RTE5 development set was used for training. We used the Charniak Parser [17] for parsing sentences, and SVM-light-TK[5] [16, 18] extended with the syntactic first-order rule kernels described in [5]. Additionally, we used the lexical overlap similarity (lex model) score described in [19]. Moreover, for obtaining the similarity matrix, we estimated the cosine similarity between all possible unique term pairs extracted from the RTE5 development and test set using the jLSI (java Latent Semantic Indexing) tool [12].

4.2 Experimental Results

Based on the configuration of each run, the results are illustrated in Table 1. The best run was performed using the same kernel method as discussed in [1], boosted by the Syntactic Semantic Tree Kernel (SSTK) described in section 3, exploiting the similarity matrix from Wikipedia to augment the cross-pair similarity. With our novel method, we outperform our previous best result by about 2% in accuracy. Moreover, the results show that our kernel approach MSK [1] is better than the PK (presented in section 3.2).

Table 1. Main task results (two-way submission)

	Settings	**Main**		**IR**		**QA**		**IE**	
		Acc.	Prc.	Acc.	Prc.	Acc.	Prc.	Acc.	Prc.
Run1	MSK	0.642	0.643	0.805	0.875	0.605	0.581	0.515	0.537
Run2	MSK+SSTK	**0.662**	0.66	0.815	0.888	0.62	0.566	0.55	0.587
Run3	PK	0.618	0.624	0.765	0.86	0.605	0.597	0.485	.487

4.3 Ablation Tests

To measure the effectiveness of our different modules and resources, we tried to ablate some of them (with respect to our first two submissions) and run our system with the same settings. The results of are illustrated in table 2 whereas the ablation modules are summarized in below:

1. Derivational Morphology from WordNet
2. The idf score
3. Proper Noun Levenshtein Distance
4. J&C similarity on nouns and adjectives
5. Verb Entailment from WordNet

[5] http://dit.unitn.it/∼moschitti/Tree-Kernel.htm

Table 2. Ablation tests result

	Main	**Abl. 1**		**Abl. 2**		**Abl. 3**		**Abl. 4**		**Abl. 5**	
	Acc.	Acc.	rel.	Acc.	rel.	Acc.	rel.	Acc.	rel.	Acc.	rel.
Run1	64	65	+1	67	+3	65	+1	64	0	66	+2
Run2	66	65	-1	65	-1	66	0	66	0	67	+1

We note that: removing J&C similarity on nouns and adjectives do not affect the results in both runs. However, ablating other modules and resources produces some effects. For example, deducting the idf score increases the accuracy in the first run and decreases it in the second run. Another interesting observation is that, in contrast with our intuition, removing the verb entailment rules extracted from WordNet increases the accuracy in both runs. However, to generalize such claim further tests and investigations are required.

References

1. Zanzotto, F.M., Moschitti, A.: Automatic learning of textual entailments with cross-pair similarities. In: ACL-44: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics, Morristown, NJ, USA, Association for Computational Linguistics (2006) 401–408
2. Bloehdorn, S., Moschitti, A.: Combined syntactic and semantic kernels for text classification. In: ECIR. (2007)
3. Basili, R., Cammisa, M., Moschitti, A.: Effective use of wordnet semantics via kernel-based learning. In: CoNLL. (2005)
4. Moschitti, A., Zanzotto, F.M.: Encoding tree pair-based graphs in learning algorithms: the textual entailment recognition case. In: Proceedings of the CoLing Workshop on Textgraph-3. (2008)
5. Zanzotto, F.M., Pennacchiotti, M., Moschitti, A.: A machine learning approach to textual entailment recognition. NATURAL LANGUAGE ENGINEERING **15-04** (2009) 551–582 Accepted for pubblication.
6. Zanzotto, F.M., Dell'arciprete, L.: Efficient kernels for sentence pair classification. In: Conference on Empirical Methods on Natural Language Processing. (6-7 August 2009) 91–100
7. Pedersen, T., Patwardhan, S., Michelizzi, J.: Wordnet::similarity - measuring the relatedness of concepts. (2004) 1024–1025
8. Basili, R., Cammisa, M., Moschitti, A.: A semantic kernel to classify texts with very few training examples. In: in Informatica, an international journal of Computing and Informatics. (2006)
9. Basili, R., Cammisa, M., Moschitti, A.: A semantic kernel to classify texts with very few training examples. In: In Proceedings of the Workshop on Learning in Web Search, at the. (2005)
10. Bloehdorn, S., Basili, R., Cammisa, M., Moschitti, A.: Semantic kernels for text classification based on topological measures of feature similarity. In: Proceedings of ICDM 06, Hong Kong, 2006. (2006)
11. Giuliano, C.: Fine-grained classification of named entities exploiting latent semantic kernels. In: CoNLL '09: Proceedings of the Thirteenth Conference on Computational Natural Language Learning, ACL (2009)

12. Giuliano, C.: jLSI a for latent semantic indexing. (2007) Software available at `http://tcc.itc.it/research/textec/tools-resources/jLSI.html`.
13. Moschitti, A.: Efficient convolution kernels for dependency and constituent syntactic trees. In: Proceedings of ECML. (2006)
14. Bloehdorn, S., Moschitti, A.: Structure and semantics for expressive text kernels. In: In proceedings of CIKM '07. (2007)
15. Moschitti, A., Zanzotto, F.M.: Fast and effective kernels for relational learning from texts. In: Proceedings of 24th Annual International Conference on Machine Learning. Volume 227., ACM (June 2007) 649–656
16. Moschitti, A.: Making tree kernels practical for natural language learning. In: Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics. (2006)
17. Charniak, E.: A maximum-entropy-inspired parser. In: Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference. (2000)
18. Joachims, T.: Making large-scale support vector machine learning practical. (1999) 169–184
19. Corley, C., Mihalcea, R.: Measuring the semantic similarity of texts. In: Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment. (2005)