haLF: Comparing a Pure CDSM Approach with a Standard Machine Learning System for RTE

Lorenzo Ferrone
University of Rome "Tor Vergata"
Via del Politecnico 1
00133 Roma, Italy
lorenzo.ferrone@gmail.com

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

Abstract

In this paper, we describe our submission to the Shared Task #1. We tried to follow the underlying idea of the task, that is, evaluating the gap of full-fledged recognizing textual entailment systems with respect to compositional distributional semantic models (CDSMs) applied to this task. We thus submitted two runs: 1) a system obtained with a machine learning approach based on the feature spaces of rules with variables and 2) a system completely based on a CDSM that mixes structural and syntactic information by using distributed tree kernels. Our analysis shows that, under the same conditions, the fully CDSM system is still far from being competitive with more complex methods.

1 Introduction

Recognizing Textual Entailment is a largely explored problem (Dagan et al., 2013). Past challenges (Dagan et al., 2006; Bar-Haim et al., 2006; Giampiccolo et al., 2007) explored methods and models applied in complex and natural texts. In this context, machine learning solutions show interesting results. The Shared Task #1 of SemEval instead wants to explore systems in a more controlled textual environment where the phenomena to model are clearer. The aim of the Shared Task is to study how RTE systems built upon compositional distributional semantic models behave with respect to the above tradition. We tried to capture this underlying idea of the task.

In this paper, we describe our submission to the Shared Task #1. We tried to follow the underlying idea of the task, that is, evaluating the gap of full-fledged recognizing textual entailment systems with respect to compositional distributional semantic models (CDSMs) applied to this task. We thus submitted two runs: 1) a system obtained with a machine learning approach based on the feature spaces of rules with variables (Zanzotto et al., 2009) and 2) a system completely based on a CDSM that mixes structural and syntactic information by using distributed tree kernels (Zanzotto and Dell'Arciprete, 2012). Our analysis shows that, under the same conditions, the fully CDSM system is still far from being competitive with more complete methods.

The rest of the paper is organized as follows. Section 2 describes the full-fledged recognizing textual entailment system that is used for comparison. Section 3 introduces a novel compositional distributional semantic model, namely, the distributed smoothed tree kernels, and the way this model is applied to the task of RTE. Section 4 describes the results in the challenge and it draws some preliminary conclusions.

2 A Standard full-fledged Machine Learning Approach for RTE

For now on, the task of recognizing textual entailment (RTE) is defined as the task to decide if a pair $p = (a, b)$ like:

("Two children are lying in the snow and are making snow angels", "Two angels are making snow on the lying children")

is in entailment, in contradiction, or neutral. As in the tradition of applied machine learn-

Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), pages 300–304,
Dublin, Ireland, August 23-24, 2014.

ing models, the task is framed as a multi-classification problem. The difficulty is to determine the best feature space on which to train the classifier.

A full-fledged RTE systems based on machine learning that has to deal with natural occurring text is generally based on:

- some within-pair features that model the similarity between the sentence a and the sentence b
- some features representing more complex information of the pair (a, b) such as rules with variables that fire (Zanzotto and Moschitti, 2006)

In the following, we describe the within-pair feature and the syntactic rules with variable features used in the full-fledged RTE system.

As the second space of features is generally huge, the full feature space is generally used in kernel machines where the final kernel between two instances $p_1 = (a_1, b_1)$ and $p_2 = (a_2, b_2)$ is:

$$\begin{aligned} K(p_1, p_2) = & \quad FR(p_1, p_2) + \\ + & \quad (WTS(a_1, b_1) \cdot WTS(a_2, b_2) + 1)^2 \end{aligned}$$

where FR counts how many rules are in common between p_1 and p_2 and WTS computes a lexical similarity between a and b. In the following sections we describe the nature of WTS and of FR

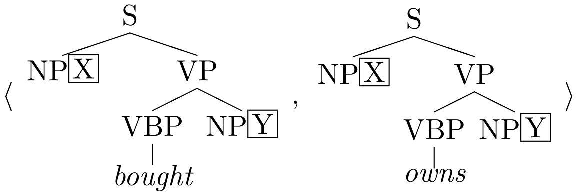
2.1 Weighted Token Similarity (WTS)

This similarity model was first defined bt Corley and Mihalcea (2005) and since then has been used by many RTE systems. The model extends a classical bag-of-word model to a Weighted-Bag-of-Word (WBOW) by measuring similarity between the two sentences of the pair at the semantic level, instead of the lexical level.

For example, consider the pair: “Oscars forgot Farrah Fawcett”, “Farrah Fawcett snubbed at Academy Awards”. This pair is redundant, and, hence, should be assigned a very high similarity. Yet, a bag-of-word model would assign a low score, since many words are not shared across the two sentences. WBOW fixes this problem by matching ‘Oscar’-‘Academy Awards’ and ‘forgot’-‘snubbed’ at the semantic level. To provide these matches, WBOW relies on specific word similarity measures over WordNet (Miller, 1995), that allow synonymy and hyperonymy matches: in our experiments we specifically use Jiang&Conrath similarity (Jiang and Conrath, 1997).

2.2 Rules with Variables as Features

The above model alone is not sufficient to capture all interesting entailment features as the relation of entailment is not only related to the notion of similarity between a and b. In the tradition of RTE, an interesting feature space is the one where each feature represents a rule with variables, i.e. a first order rule that is activated by the pairs if the variables are unified. This feature space has been introduced in (Zanzotto and Moschitti, 2006) and shown to improve over the one above. Each feature $\langle fr_1, fr_2 \rangle$ is a pair of syntactic tree fragments augmented with variables. The feature is active for a pair (t_1, t_2) if the syntactic interpretations of t_1 and t_2 can be unified with $< fr_1, fr_2 >$. For example, consider the following feature:



This feature is active for the pair (“*GM bought Opel*”,“*GM owns Opel*”), with the variable unification $\boxed{\text{X}}$ = “*GM*” and $\boxed{\text{Y}}$ = “*Opel*”. On the contrary, this feature is not active for the pair (“*GM bought Opel*”,“*Opel owns GM*”) as there is no possibility of unifying the two variables.

$FR(p_1, p_2)$ is a kernel function that counts the number of common rules with variables between p_1 and p_2. Efficient algorithms for the computation of the related kernel functions can be found in (Moschitti and Zanzotto, 2007; Zanzotto and Dell’Arciprete, 2009; Zanzotto et al., 2011).

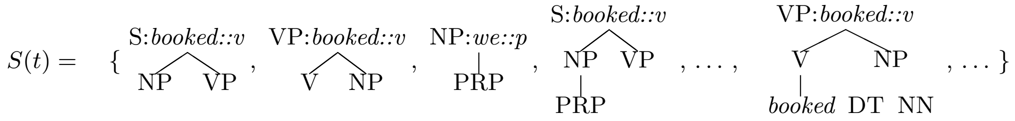


Figure 1: Subtrees of the tree t in Figure 2 (a non-exhaustive list.)

3 Distributed Smoothed Tree Kernel: a Compositional Distributional Semantic Model for RTE

The above full-fledged RTE system, although it may use distributional semantics, is not a model that applies a compositional distributional semantic model as it does not explicitly transform sentences in vectors, matrices, or tensors that represent their meaning.

We here propose a model that can be considered a compositional distributional semantic model as it transforms sentences into matrices that are then used by the learner as feature vectors. Our model is called *Distributed Smoothed Tree Kernel* (Ferrone and Zanzotto, 2014) as it mixes the distributed trees (Zanzotto and Dell'Arciprete, 2012) representing syntactic information with distributional semantic vectors representing semantic information. The computation of the final matrix for each sentence is done compositionally.

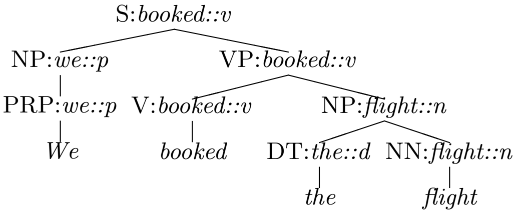


Figure 2: A lexicalized tree.

3.1 Notation

Before describing the *distributed smoothed trees* (DST) we introduce a formal way to denote constituency-based *lexicalized parse trees*, as DSTs exploit this kind of data structures. *Lexicalized trees* are denoted with the letter t and $N(t)$ denotes the set of non terminal nodes of tree t. Each non-terminal node $n \in N(t)$ has a label l_n composed of two parts $l_n = (s_n, w_n)$: s_n is the syntactic label, while w_n is the semantic headword of the tree headed by n, along with its part-of-speech tag. Terminal nodes of trees are treated differently, these nodes represent only words w_n without any additional information, and their labels thus only consist of the word itself (see Fig. 2). The structure of a DST is represented as follows: Given a tree t, $\mathsf{h}(t)$ is its root node and $\mathsf{s}(t)$ is the tree formed from t but considering only the syntactic structure (that is, only the s_n part of the labels), $c_i(n)$ denotes i-th child of a node n. As usual for constituency-based parse trees, pre-terminal nodes are nodes that have a single terminal node as child.

Finally, we use $\vec{w_n} \in \mathbb{R}^k$ to denote the *distributional* vector for word w_n, whereas $\mathbf{T}$ represents the matrix of a tree t encoding structure and distributional meaning.

3.2 The Method in a Glance

We describe here the approach in a few sentences. In line with tree kernels over structures (Collins and Duffy, 2002), we introduce the set $S(t)$ of the subtrees t_i of a given lexicalized tree t. A subtree t_i is in the set $S(t)$ if $\mathsf{s}(t_i)$ is a subtree of $\mathsf{s}(t)$ and, if n is a node in t_i, all the siblings of n in t are in t_i. For each node of t_i we only consider its syntactic label s_n, except for the head $\mathsf{h}(t_i)$ for which we also consider its semantic component w_n (see Fig. 1). The functions DSTs we define compute the following:

$$DST(t) = \mathbf{T} = \sum_{t_i \in S(t)} \mathbf{T}_i$$

where $\mathbf{T}_i$ is the matrix associated to each subtree t_i. The similarity between two text fragments a and b represented as lexicalized trees t^a and t^b can be computed using the Frobenius product between the two matrices $\mathbf{T}^a$ and $\mathbf{T}^b$, that is:

$$\langle \mathbf{T}^a, \mathbf{T}^b \rangle_F = \sum_{\substack{t_i^a \in S(t^a) \\ t_j^b \in S(t^b)}} \langle \mathbf{T}_i^a, \mathbf{T}_j^b \rangle_F \qquad (1)$$

We want to obtain that the product $\langle \mathbf{T}_i^a, \mathbf{T}_j^b \rangle_F$ approximates the dot product between the distributional vectors of the head words ($\langle \mathbf{T}_i^a, \mathbf{T}_j^b \rangle_F \approx \langle \overrightarrow{\mathsf{h}(t_i^a)}, \overrightarrow{\mathsf{h}(t_j^b)} \rangle$) whenever the syntactic structure of the subtrees is the same (that is $\mathsf{s}(t_i^a) = \mathsf{s}(t_j^b)$), and $\langle \mathbf{T}_i^a, \mathbf{T}_j^b \rangle_F \approx 0$ otherwise. This property is expressed as:

$$\langle \mathbf{T}_i^a, \mathbf{T}_j^b \rangle_F \approx \delta(\mathsf{s}(t_i^a), \mathsf{s}(t_j^b)) \cdot \langle \overrightarrow{\mathsf{h}(t_i^a)}, \overrightarrow{\mathsf{h}(t_j^b)} \rangle \quad (2)$$

To obtain the above property, we define

$$\mathbf{T}_i = \overrightarrow{\mathsf{s}(t_i)} \overrightarrow{w_{\mathsf{h}(t_i)}}^{\top}$$

where $\overrightarrow{\mathsf{s}(t_i)}$ are distributed tree fragment (Zanzotto and Dell'Arciprete, 2012) for the subtree t and $\overrightarrow{w_{\mathsf{h}(t_i)}}$ is the distributional vector of the head of the subtree t. Distributed tree fragments have the property that $\overrightarrow{\mathsf{s}(t_i)} \overrightarrow{\mathsf{s}(t_j)} \approx \delta(t_i, t_j)$. Thus, given the important property of the outer product that applies in the Frobenius product: $\langle \vec{a}\vec{w}^{\top}, \vec{b}\vec{v}^{\top} \rangle_F = \langle \vec{a}, \vec{b} \rangle \cdot \langle \vec{w}, \vec{v} \rangle$. we have that Equation 2 is satisfied as:

$$\begin{aligned} \langle \mathbf{T}_i, \mathbf{T}_j \rangle_F &= \langle \overrightarrow{\mathsf{s}(t_i)}, \overrightarrow{\mathsf{s}(t_j)} \rangle \cdot \langle \overrightarrow{w_{\mathsf{h}(t_i)}}, \overrightarrow{w_{\mathsf{h}(t_j)}} \rangle \\ &\approx \delta(\mathsf{s}(t_i), \mathsf{s}(t_j)) \cdot \langle \overrightarrow{w_{\mathsf{h}(t_i)}}, \overrightarrow{w_{\mathsf{h}(t_j)}} \rangle \end{aligned}$$

It is possible to show that the overall compositional distributional model $DST(t)$ can be obtained with a recursive algorithm that exploit vectors of the nodes of the tree.

The compositional distributional model is then used in the same learning machine used for the traditional RTE system with the following kernel function:

$$\begin{gathered} K(p_1, p_2) = \\ \langle DST(a_1), DST(a_2) \rangle + \langle DST(b_1), DST(b_2) \rangle + \\ +(WTS(a_1, b_1) \cdot WTS(a_2, b_2) + 1)^2 \end{gathered}$$

4 Results and Conclusions

For the submission we used the java version of LIBSVM (Chang and Lin, 2011). Distributional vectors are derived with DISSECT (Dinu et al., 2013) from a corpus obtained by the concatenation of ukWaC (wacky.sslmit.unibo.it), a mid-2009 dump of the English Wikipedia (en.wikipedia.org) and the British National Corpus (www.natcorp.ox.ac.uk), for a total of about 2.8 billion words. The raw co-occurrences count vectors were transformed into positive Pointwise Mutual Information scores and reduced to 300 dimensions by Singular Value Decomposition. This setup was picked without tuning, as we found it effective in previous, unrelated experiments.

Model	Accuracy (3-ways)
DST	69.42
full-fledged RTE System	75.66
Max	84.57
Min	48.73
Average	75.35

Table 1: Accuracies of the two systems on the test set, together with the maximum, minimum and average score for the challenge.

We parsed the sentence with the Stanford Parser (Klein and Manning, 2003) and extracted the heads for use in the lexicalized trees with Collins' rules (Collins, 2003).

Table 1 reports our results on the textual entailment classification task, together with the maximum, minimum and average score for the challenge. The first observation is that the full-fledged RTE system is still definitely better than our CDSM system. We believe that the main reason is that the DST cannot encode variables which is an important aspect to capture when dealing with textual entailment recognition. This is particularly true for this dataset as it focuses on word ordering and on specific and recurrent entailment rules. Our full-fledged system scored among the first 10 systems, slightly above the overall average score, but our pure CDSM system is instead ranked within the last 3. We think that a more in-depth comparison with other fully CDSM systems will give us a better insight on our model and will also assess more realistically the quality of our system.

References

Roy Bar-Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The second pascal recognising textual entailment challenge. In *Proceedings of the Second PASCAL Challenges Work-*

shop on Recognising Textual Entailment. Venice, Italy.

Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

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.*

Michael Collins. 2003. Head-driven statistical models for natural language parsing. *Comput. Linguist.*, 29(4):589–637.

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*, pages 13–18. Association for Computational Linguistics, Ann Arbor, Michigan, June.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The pascal recognising textual entailment challenge. In Quionero-Candela et al., editor, *LNAI 3944: MLCW 2005*, pages 177–190. Springer-Verlag, Milan, Italy.

Ido Dagan, Dan Roth, Mark Sammons, and Fabio Massimo Zanzotto. 2013. *Recognizing Textual Entailment: Models and Applications.* Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Georgiana Dinu, Nghia The Pham, and Marco Baroni. 2013. DISSECT: DIStributional SEmantics Composition Toolkit. In *Proceedings of ACL (System Demonstrations)*, pages 31–36, Sofia, Bulgaria.

Lorenzo Ferrone and Fabio Massimo Zanzotto. 2014. Towards syntax-aware compositional distributional semantic models. In *Proceedings of Coling 2014.* COLING, Dublin, Ireland, Aug 23–Aug 29.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 1–9. Association for Computational Linguistics, Prague, June.

Jay J. Jiang and David W. Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proc. of the 10th ROCLING*, pages 132–139. Tapei, Taiwan.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 423–430, Stroudsburg, PA, USA. Association for Computational Linguistics.

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

Alessandro Moschitti and Fabio Massimo Zanzotto. 2007. Fast and effective kernels for relational learning from texts. In *Proceedings of the International Conference of Machine Learning (ICML).* Corvallis, Oregon.

Fabio Massimo Zanzotto and Lorenzo Dell'Arciprete. 2009. Efficient kernels for sentence pair classification. In *Conference on Empirical Methods on Natural Language Processing*, pages 91–100, 6-7 August.

F.M. Zanzotto and L. Dell'Arciprete. 2012. Distributed tree kernels. In *Proceedings of International Conference on Machine Learning*, pages 193–200.

Fabio Massimo Zanzotto 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.

Fabio Massimo Zanzotto, Marco Pennacchiotti, and Alessandro Moschitti. 2009. A machine learning approach to textual entailment recognition. *NATURAL LANGUAGE ENGINEERING*, 15-04:551–582.

Fabio Massimo Zanzotto, Lorenzo Dell'Arciprete, and Alessandro Moschitti. 2011. Efficient graph kernels for textual entailment recognition. *Fundamenta Informaticae*, 107(2-3):199 – 222.