A Preliminary Evaluation of the Impact of Syntactic Structure in Semantic Textual Similarity and Semantic Relatedness Tasks

Ngoc Phuoc An Vo
Fondazione Bruno Kessler,
University of Trento
Trento, Italy
ngoc@fbk.eu

Octavian Popescu
IBM Research, T.J. Watson
Yorktown, US
o.popescu@us.ibm.com

Abstract

The well related tasks of evaluating the Semantic Textual Similarity and Semantic Relatedness have been under a special attention in NLP community. Many different approaches have been proposed, implemented and evaluated at different levels, such as lexical similarity, word/string/POS tags overlapping, semantic modeling (LSA, LDA), etc. However, at the level of syntactic structure, it is not clear how significant it contributes to the overall accuracy. In this paper, we make a preliminary evaluation of the impact of the syntactic structure in the tasks by running and analyzing the results from several experiments regarding to how syntactic structure contributes to solving these tasks.

1 Introduction

Since the introduction of Semantic Textual Similarity (STS) task at SemEval 2012 and the Semantic Relatedness (SR) task at SemEval 2014, a large number of participating systems have been developed to resolve the tasks.[1,2] The systems must quantifiably identify the degree of similarity, relatedness, respectively, for pair of short pieces of text, like sentences, where the similarity or relatedness is a broad concept and its value is normally obtained by averaging the opinion of several annotators. A semantic similarity/relatedness score is usually a real number in a semantic scale, [0-5] in STS, or [1-5] in SR, in the direction from *no relevance* to *semantic equivalence*. Some examples from the dataset *MSRpar* of STS 2012 with associated similarity scores (by human judgment) are as below:

- *The bird is bathing in the sink.* vs. *Birdie is washing itself in the water basin.* (score = 5.0)
- *Shares in EDS closed on Thursday at $18.51, a gain of 6 cents.* vs. *Shares of EDS closed Thursday at $18.51, up 6 cents on the New York Stock Exchange.* (score = 3.667)
- *Vivendi shares closed 3.8 percent up in Paris at 15.78 euros.* vs. *Vivendi shares were 0.3 percent up at 15.62 euros in Paris at 0841 GMT.* (score = 2.6)
- *John went horse back riding at dawn with a whole group of friends.* vs. *Sunrise at dawn is a magnificent view to take in if you wake up early enough for it.* (score = 0)

From our reading of the literature (Marelli et al., 2014b; Agirre et al., 2012; Agirre et al., 2013; Agirrea et al., 2014), most of STS/SR systems rely on pairwise similarity, such as lexical similarity using taxonomies (WordNet (Fellbaum, 1998)) or distributional semantic models (LDA (Blei et al., 2003), LSA (Landauer et al., 1998), ESA (Gabrilovich and Markovitch, 2007), etc), and word/n-grams overlap as main features to train a support vector machines (Joachims, 1998) regression model (supervised), or use a word-alignment metric (unsupervised) aligning the two given texts to compute their semantic similarity.

Intuitively, the syntactic structure plays an important role for human being to understand the mean-

[1] http://www.cs.york.ac.uk/semeval-2012/task6
[2] http://alt.qcri.org/semeval2014/task1

Proceedings of NAACL-HLT 2015 Student Research Workshop (SRW), pages 64–70,
Denver, Colorado, June 1, 2015.

ing of a given text. Thus, it also may help to identify the semantic equivalence/relatedness between two given texts. However, in the STS/SR tasks, very few systems provide evidence of the contribution of syntactic structure in its overall performance. Some systems report partially on this issue, for example, iKernels (Severyn et al., 2013) carried out an analysis on the STS 2012, but not on STS 2013 datasets. They found that syntactic structure contributes 0.0271 and 0.0281 points more to the overall performance, from 0.8187 to 0.8458 and 0.8468, for adopting constituency and dependency trees, respectively.

In this paper, we analyze the impact of syntactic structure on the STS 2014 and SICK datasets of STS/SR tasks. We consider three systems which are reported to perform efficiently and effectively on processing syntactic trees using three proposed approaches Syntactic Tree Kernel (Moschitti, 2006), Syntactic Generalization (Galitsky, 2013) and Distributed Tree Kernel (Zanzotto and Dell'Arciprete, 2012).

The remainder of the paper is as follows: Section 2 introduces three approaches to exploit the syntactic structure in STS/SR tasks, Section 3 describes Experimental Settings, Section 4 discusses about the Evaluations and Section 5 is the Conclusions and Future Work.

2 Three Approaches for Exploiting the Syntactic Structure

In this section, we describe three different approaches exploiting the syntactic structure to be used in the STS/SR tasks, which are **Syntactic Tree Kernel** (Moschitti, 2006), **Syntactic Generalization** (Galitsky, 2013), and **Distributed Tree Kernel** (Zanzotto and Dell'Arciprete, 2012). All these three approaches learn the syntactic information either from the dependency parse trees produced by the Stanford Parser (standard PCFG Parser) (Klein and Manning, 2003) or constituency parse trees obtained by OpenNLP.[3] The output of each approach is normalized to the standard semantic scale of STS [0-5] or SR [1-5] tasks to evaluate its standalone performance, or combined with other features in our baseline system for assessing its contribution to the overall accuracy by using the same WEKA machine learning tool (Hall et al., 2009) with as same configurations and parameters as our baseline systems.

[3] https://opennlp.apache.org

2.1 Syntactic Tree Kernel (STK)

Given two trees T1 and T2, the functionality of tree kernels is to compare two tree structures by computing the number of common substructures between T1 and T2 without explicitly considering the whole fragment space. According to the literature (Moschitti, 2006), there are three types of fragments described as the subtrees (STs), the subset trees (SSTs) and the partial trees (PTs). A subtree (ST) is a node and all its children, but terminals are not STs. A subset tree (SST) is a more general structure since its leaves need not be terminals. The SSTs satisfy the constraint that grammatical rules cannot be broken. When this constraint is relaxed, a more general form of substructures is obtained and defined as partial trees (PTs).

Syntactic Tree Kernel (STK) (Moschitti, 2006) is a tree kernels approach to learn the syntactic structure from syntactic parsing information, particularly, the Partial Tree (PT) kernel is proposed as a new convolution kernel to fully exploit dependency trees. The evaluation of the common PTs rooted in nodes n1 and n2 requires the selection of the shared child subsets of the two nodes, e.g. [S [DT JJ N]] and [S [DT N N]] have [S [N]] (2 times) and [S [DT N]] in common.

In order to learn the similarity of syntactic structure, we seek for a corpus which should fulfill the two requirements, (1) sentence-pairs contain similar syntactic structure, and with (2) a variety of their syntactic structure representations (in their parsing trees). However, neither SICK nor STS corpus seems to be suitable. As the SICK corpus is designed for evaluating compositional distributional semantic models through semantic relatedness and textual entailment, the syntactic structure of sentence pairs are quite simple and straightforward. In contrast, the STS corpus contains several different datasets derived from different sources (see Table 1) which carry a large variety of syntactic structure representations, but lack of learning examples due to no human annotation given for syntactic structure similarity (only annotation for semantic similarity exists); and it is difficult to infer the syntactic structure

similarity from general semantic similarity scores in STS datasets. Hence, having assumed that paraphrased pairs would share the same content and similar syntactic structures, we decide to choose the Microsoft Research Paraphrasing Corpus (Dolan et al., 2005) which contains 5,800 sentence pairs extracted from news sources on the web, along with human annotations indicating whether each pair captures a paraphrase/semantic equivalence relationship.[4] This corpus is split into Training set (4,076 pairs) and Testing set (1,725 pairs).

We use Stanford Parser (PCFG Parser) trained on Penn TreeBank (Klein and Manning, 2003) to obtain the dependency parsing from sentence pairs. Then we use the machine learning tool svm-light-tk 1.2 which uses Tree Kernel approach to learn the similarity of syntactic structure to build a binary classifying model on the Train dataset.[5] According to the assumption above, we label paraphrased pairs as 1, -1 otherwise. We test this model on the Test dataset and obtain the Accuracy of 69.16%, with Precision/Recall is: 69.04%/97.21%.

We apply this model on the STS and SICK data to predict the similarity between sentence pairs. The output predictions are probability confidence scores in [-1,1], corresponds to the probability of the label to be positive. Thus, we convert the prediction value into the semantic scale of STS and SR tasks to compare to the human annotation. The example data (including train, test, and predictions) of this tool is available here.[6]

2.2 Syntactic Generalization (SG)

Given a pair of parse trees, the Syntactic Generalization (SG) (Galitsky, 2013) finds a set of maximal common subtrees. Though generalization operation is a formal operation on abstract trees, it yields semantics information from commonalities between sentences. Instead of only extracting common keywords from two sentences, the generalization operation produces a syntactic expression. This expression maybe semantically interpreted as a common meaning held by both sentences. This syntactic parse tree generalization learns the semantic information differently from the kernel methods which compute a kernel function between data instances, whereas a kernel function is considered as a similarity measure. Other than the kernel methods, SG is considered as structure-based and deterministic, in which linguistic features remain their structure, not as value presentations.

The toolkit "relevance-based-on-parse-trees" is an open-source project which evaluates text relevance by using syntactic parse tree-based similarity measure.[7] Given a pair of parse trees, it measures the similarity between two sentences by finding a set of maximal common subtrees, using representation of constituency parse trees via chunking. Each type of phrases (NP, VP, PRP etc.) will be aligned and subject to generalization. It uses the OpenNLP system to derive dependency trees for generalization (chunker and parser).[8] This tool is made to give as a tool for text relevance which can be used as a black box, no understanding of computational linguistics or machine learning is required. We apply the tool on the SICK and STS datasets to compute the similarity of syntactic structure of sentence pairs. The similarity score from this tool is converted into the semantic scale of STS and SR tasks for comparison against the human annotation.

2.3 Distributed Tree Kernel (DTK)

Distributed Tree Kernel (DTK) (Zanzotto and Dell'Arciprete, 2012) is a tree kernels method using a linear complexity algorithm to compute vectors for trees by embedding feature spaces of tree fragments in low-dimensional spaces. Then a recursive algorithm is proposed with linear complexity to compute reduced vectors for trees. The dot product among reduced vectors is used to approximate the original tree kernel when a vector composition function with specific ideal properties is used.

Firstly, we use Stanford Parser (PCFG Parser) trained on Penn TreeBank (Klein and Manning, 2003) to obtain the dependency parsing of sentences, and feed them to the software "distributed-tree-kernels" to produce the distributed trees.[9] Then, we compute the Cosine similarity between the vectors of distributed trees of each sentence pair. This

[4] http://research.microsoft.com/en-us/downloads/607d14d9-20cd-47e3-85bc-a2f65cd28042

[5] http://disi.unitn.it/moschitti/SIGIR-tutorial.htm

[6] http://disi.unitn.it/moschitti/Tree-Kernel.htm

[7] https://code.google.com/p/relevance-based-on-parse-trees

[8] https://opennlp.apache.org

[9] https://code.google.com/p/distributed-tree-kernels

cosine similarity score is converted to the scale of STS and SR for evaluation.

3 Experiments

In this section, we describe the two corpora we use for experiments with several different settings to evaluate the contribution of each syntactic structure approach and in combination with other features in our baseline systems.

3.1 Datasets

We run our experiments on two datasets from two different tasks at SemEval 2014 as follows:

- The SICK dataset (Marelli et al., 2014a) is used in Task# 1 "Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment".[10] It consists of 10,000 English sentence pairs, built from two paraphrase sets: the 8K ImageFlickr dataset and the STS 2012 Video Descriptions dataset.[11,12] Each sentence pair was annotated for relatedness score in scale [1-5] and entailment relation. It is split into three parts: Trial (500 pairs), Training (4,500 pairs) and Testing (4,927 pairs).

- The STS dataset is used in Task #10 "Multilingual Semantic Textual Similarity" (STS English subtask) which consists of several datasets in STS 2012 (Agirre et al., 2012), 2013 (Agirre et al., 2013) and 2014 (Agirrea et al., 2014). Each sentence pair is annotated the semantic similarity score in the scale [0-5]. Table 1 shows the summary of STS datasets and sources over the years. For training, we use all data in STS 2012 and 2013; and for evaluation, we use STS 2014 datasets.

3.2 Baselines

In order to evaluate the significance of syntactic structure in the STS/SR tasks, we not only examine the syntactic structure alone, but also combine it with some features learned from common approaches, such as bag-of-words, pairwise similarity, n-grams overlap, etc. Therefore, we use two baseline systems for evaluations, the weak and the strong ones. The weak baseline is the basic one used for evaluation in all the STS tasks, namely **tokencos**. It uses the bag-of-words approach which represents each sentence as a vector in the multidimensional token space (each dimension has 1 if the token is present in the sentence, 0 otherwise) and computes the cosine similarity between vectors.

Besides the weak baseline, we use **DKPro Similarity** (Bär et al., 2012) as a strong baseline which is an open source software and intended to use as a baseline-system in the share task STS at *SEM 2013.[13] It uses a simple log-linear regression model (about 18 features), to combine multiple text similarity measures of varying complexity ranging from simple character/word n-grams and common subsequences to complex features such as Explicit Semantic Analysis vector comparisons and aggregation of word similarity based on lexical-semantic resources (WordNet and Wiktionary).[14,15]

year	dataset	pairs	source
2012	MSRpar	1500	newswire
2012	MSRvid	1500	video descriptions
2012	OnWN	750	OntoNotes, WordNet glosses
2012	SMTnews	750	Machine Translation evaluation
2012	SMTeuroparl	750	Machine Translation evaluation
2013	headlines	750	newswire headlines
2013	FNWN	189	FrameNet, WordNet glosses
2013	OnWN	561	OntoNotes, WordNet glosses
2013	SMT	750	Machine Translation evaluation
2014	headlines	750	newswire headlines
2014	OnWN	750	OntoNotes, WordNet glosses
2014	Deft-forum	450	forum posts
2014	Deft-news	300	news summary
2014	Images	750	image descriptions
2014	Tweet-news	750	tweet-news pairs

Table 1: Summary of STS datasets in 2012, 2013, 2014.

4 Evaluations and Discussions

In this section, we present twelve different settings for experimenting the contribution of syntactic structure individually and in combination with typi-

[10] http://alt.qcri.org/semeval2014/task1

[11] http://nlp.cs.illinois.edu/HockenmaierGroup/data.html

[12] http://www.cs.york.ac.uk/semeval-2012/task6/index.php?id=data

[13] https://code.google.com/p/dkpro-similarity-asl/wiki/SemEval2013

[14] http://wordnet.princeton.edu

[15] http://en.wiktionary.org/wiki/Wiktionary:Main_Page

Settings	deft-forum	deft-news	headlines	images	OnWN	tweet-news	STS2014 Mean	SICK-test
Tokencos (0)	0.353	0.596	0.510	0.513	0.406	0.654	0.5054	0.501
DKPro (1)	0.4314	0.7089	0.6887	0.7671	0.8125	0.6932	0.6836	0.6931
STK (2)	0.1163	0.2369	0.0374	*-0.1125*	0.0865	*-0.0296*	0.0558	0.0757
SG (3)	0.2816	0.3808	0.4078	0.4449	0.4934	0.5487	0.4262	0.4498
DTK (4)	0.0171	0.1	*-0.0336*	*-0.109*	0.0359	*-0.0986*	*-0.0147*	0.2657
STK & SG & DTK	0.2402	0.3886	0.3233	0.2419	0.4066	0.4489	0.3416	0.4822
(0) & (2)	0.3408	0.5738	0.4817	0.4184	0.4029	0.6016	0.4699	**0.5074**
(0) & (3)	**0.3735**	0.5608	**0.5367**	**0.5432**	**0.4813**	**0.6736**	**0.5282**	**0.522**
(0) & (4)	**0.3795**	**0.6343**	**0.5399**	0.5096	**0.4504**	0.6539	**0.5279**	**0.5018**
(0), (2), (3) & (4)	**0.3662**	0.5867	**0.5265**	0.464	**0.4758**	0.6407	**0.51**	**0.5252**
(1) & (2)	**0.4423**	0.7019	**0.6919**	0.7653	0.8122	**0.7105**	**0.6874**	**0.7239**
(1) & (3)	**0.4417**	0.7067	0.6844	0.7636	0.812	0.6777	0.6810	**0.6948**
(1) & (4)	0.4314	0.7089	0.6887	0.7671	0.8125	0.6932	0.6836	**0.6953**
(1), (2), (3) & (4)	**0.4495**	0.7032	**0.6902**	0.7627	0.8115	**0.6974**	**0.6857**	**0.7015**

Table 2: Experiment Results on STS 2014 and SICK datasets.

cal similarity features to the overall performance of computing similarity/relatedness score on SICK and STS datasets. The results reported here are obtained with Pearson correlation, which is the official measure used in both tasks.[16] We have some discussions from the results in Table 2 as below:

Baseline comparison. The strong baseline DKPro is superior than the bag-of-word baseline on most of datasets (both STS and SICK), except the *tweet-news* where their performances are close as the *tweet-news* dataset contains little or no syntactic information compared to others.

Individual approach evaluation. Each syntactic approach is weaker than both baselines. Though the STK and DTK both use the tree kernel approach, just different representations, the performance is similar only on the dataset *images*. The STK still performs better than DTK on most of STS datasets, but much lower on SICK dataset. This is reasonable as the SICK dataset is created for evaluating distributional semantics which suits the DTK approach. Both approaches have some negative results on STS datasets; especially, both methods obtain negative correlation on two datasets *"images"* and *"tweet-news"*. It seems that both methods struggle to learn the semantic information (in parsing) extracted from these two datasets. Moreover, due to the fact that Twitter data is informal text which carries lot of noise created by users, and very different from formal text from other STS datasets, the syntactic approach does not seem to capture correct meaning, thus, the result confirms that syntactic approach is not suitable and beneficial for social media text.

In contrast, the SG performs better than other two approaches to obtain better correlation with human judgment; yet it is still below the bag-of-word baseline (only better on *OnWN* dataset). Hence, using any of these syntactic approaches is not sufficient to solve the STS/SR task as its performance is still lower than the weak baseline. Some examples with gold-standard and system scores as below:

- *Blue and red plane in mid-air flight.* vs. *A blue and red airplane while in flight.* (gold=4.8; STK=3.418; DTK=3.177; SG=3.587)
- *Global online #education is a key to democratizing access to learning and overcoming societal ills such as poverty* vs. *Op-Ed Columnist: Revolution Hits the Universities* (gold=0.6; STK=3.054; DTK=3.431; SG=2.074)
- *you are an #inspiration! #Keepfighting* vs. *The front lines in fight for women* (gold=0.4; STK=3.372; DTK=3.479; SG=2.072)
- *CGG - 30 die when bus plunges off cliff in Nepal* vs. *30 killed as bus plunges off cliff*

[16] http://en.wikipedia.org/wiki/Pearson_product-moment_correlation_coefficient

in Nepal (gold=5; STK=3.155; DTK=3.431; SG=3.402)

The combination of three approaches. These three methods do not collaborate well on STS datasets, it even decreases the overall performance of the best method SG by a large margin of 8%. However, it improves the result on SICK dataset by a medium margin around 4%. Finally, the combination of three methods still returns a lower result than the weak baseline. Thus, this combination of syntactic approaches alone cannot solve the STS/SR tasks.

Combination with bag-of-word approach. The combination of syntactic information and bag-of-word approach more or less improves the performance over the weak baseline.

- The STK does not improve but has negative impact to the overall performance on STS with a decrease of 4%. However, it gains a small improvement on SICK of 1%.
- Though the DTK returns 3.5% better result than STK on STS and slightly improves the performance on SICK for less than 1%, it is 0.5% lower than the weak baseline.
- The SG improves the performance 2-12% on most of STS and SICK datasets. It performs 4-8% better than the weak baseline, but still dramatically 11-14% lower than the DKPro baseline.
- The combination of three methods with the bag-of-word results 3-8% better performance than the weak baseline on STS/SICK datasets. However, this combination brings negative effect of 0.5% to the overall result on STS in comparison to the performance of SG.

Combination with DKPro. Perhaps DKPro baseline consists of several strong features which make syntactic features insignificant in the combination. Hence, using a strong baseline like DKPro is not a good way to evaluate the significance of syntactic information.

- The STK gains small improvement on SICK (3%) and some STS datasets (1%), whereas other datasets remain unchanged.
- The DTK does not have any effect to the result of DKPro standalone. This shows that DTK has no integration with DKPro features.
- The SG only makes slight improvement on SICK (0.2%) and *deft-forum* (1%), whereas little decrease on other datasets. This shows that SG does not collaborate well with DKPro either.
- On STS, this total combination returns few small improvements around 1% on some datasets *deft-forum, headlines, tweet-news* and mean value, whereas 1-3% better on SICK dataset.

In conclusion, despite the fact that we experiment different methods to exploit syntactic information on different datasets derived from various data sources, the results in Table 2 confirms the positive impact of syntactic structure in the overall performance on STS/SR tasks. However, syntactic structure does not always work well and effectively on any dataset, it requires a certain level of syntactic presentation in the corpus to exploit. In some cases, applying syntactic structure on poor-structured data may cause negative effect to the overall performance.

5 Conclusions and Future Work

In this paper, we deploy three different approaches to exploit and evaluate the impact of syntactic structure in the STS/SR tasks. We use a freely available STS system, DKPro, which is using similarity features for computing the semantic similarity/relatedness scores as a strong baseline. We also evaluate the contribution of each syntactic structure approach and different combinations between them and the typical similarity approach in the baseline. From our observation, in the mean time with recent proposed approaches, the results in Table 2 shows that the syntactic structure does contribute individually and together with typical similarity approaches for computing the semantic similarity/relatedness scores between given sentence pairs. However, compared to the baselines, the contribution of syntactic structure is not significant to the overall performance. For future work, we may expect to see more effective ways for exploiting and learning syntactic structure to have better contribution into the overall performance in the STS/SR tasks.

References

Eneko Agirre, Mona Diab, Daniel Cer, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 task 6: A pilot on semantic textual similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, pages 385–393. Association for Computational Linguistics.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. sem 2013 shared task: Semantic textual similarity, including a pilot on typed-similarity. In *In* SEM 2013: The Second Joint Conference on Lexical and Computational Semantics. Association for Computational Linguistics*. Citeseer.

Eneko Agirrea, Carmen Baneab, Claire Cardiec, Daniel Cerd, Mona Diabe, Aitor Gonzalez-Agirrea, Weiwei Guof, Rada Mihalceab, German Rigaua, and Janyce Wiebeg. 2014. Semeval-2014 task 10: Multilingual semantic textual similarity. *SemEval 2014*, page 81.

Daniel Bär, Chris Biemann, Iryna Gurevych, and Torsten Zesch. 2012. Ukp: Computing semantic textual similarity by combining multiple content similarity measures. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, pages 435–440. Association for Computational Linguistics.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.

Bill Dolan, Chris Brockett, and Chris Quirk. 2005. Microsoft research paraphrase corpus. *Retrieved March*, 29:2008.

Christiane Fellbaum. 1998. *WordNet*. Wiley Online Library.

Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *IJCAI*, volume 7, pages 1606–1611.

Boris Galitsky. 2013. Machine learning of syntactic parse trees for search and classification of text. *Engineering Applications of Artificial Intelligence*, 26(3):1072–1091.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. 2009. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18.

Thorsten Joachims. 1998. *Text categorization with support vector machines: Learning with many relevant features*. Springer.

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

Thomas K Landauer, Peter W Foltz, and Darrell Laham. 1998. An introduction to latent semantic analysis. *Discourse processes*, 25(2-3):259–284.

M Marelli, S Menini, M Baroni, L Bentivogli, R Bernardi, and R Zamparelli. 2014a. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of LREC 2014, Reykjavik (Iceland): ELRA*.

Marco Marelli, Luisa Bentivogli, Marco Baroni, Raffaella Bernardi, Stefano Menini, and Roberto Zamparelli. 2014b. Semeval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. *SemEval-2014*.

Alessandro Moschitti. 2006. Efficient convolution kernels for dependency and constituent syntactic trees. In *Machine Learning: ECML 2006*, pages 318–329. Springer.

Aliaksei Severyn, Massimo Nicosia, and Alessandro Moschitti. 2013. ikernels-core: Tree kernel learning for textual similarity. *Atlanta, Georgia, USA*, page 53.

Fabio Massimo Zanzotto and Lorenzo Dell'Arciprete. 2012. Distributed tree kernels. *arXiv preprint arXiv:1206.4607*.