Learning the Impact and Behavior of Syntactic Structure: A Case Study in Semantic Textual Similarity

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

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

Abstract

We present a case study on the role of syntactic structures towards resolving the Semantic Textual Similarity (STS) task. Although various approaches have been proposed, the research of using syntactic information to determine the semantic similarity is a relatively under-researched area. At the level of syntactic structure, it is not clear how significant the syntactic structure contributes to the overall accuracy of the task. In this paper, we analyze the impact of syntactic structure towards the overall performance and its behavior in different score ranges of the STS semantic scale.

1 Introduction

The task Semantic Textual Similarity (STS) has become a noticed trend in the Natural Language Processing (NLP) community since the SemEval 2012 with a large number of participating systems.[1] The participating systems should be able to determine the degree of similarity for pair of short pieces of text, like sentences, where the similarity score is normally obtained by averaging the opinion of several annotators. A semantic similarity score is usually a real number in a semantic scale [0-5], from *no relevance* to *semantic equivalence*. Some examples from the STS 2012 dataset with associated similarity scores (by human judgment) are as below:

_ *In May 2010, the troops attempted to invade Kabul.* vs. *The US army invaded Kabul on May 7th last year, 2010.* (score = 4.0)

_ *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)

_ *The woman is playing the violin.* vs. *The young lady enjoys listening to the guitar.* (score = 1.0)

The literature (Agirre et al., 2012; Agirre et al., 2013; Agirre et al., 2014) shows that in order to compute the semantic similarity, most STS systems rely on pairwise similarity, either using taxonomies (WordNet (Fellbaum, 1998)) or distributional semantic models LDA (Blei et al., 2003), LSA (Landauer et al., 1998), ESA (Gabrilovich and Markovitch, 2007), and word/n-grams overlap as main features to train supervised models, or deploy unsupervised word-alignment metrics to align two given texts.

In common sense, syntactic structure may keep a crucial part for human being to understand the meaning of a given text. Thus, it also may help to identify the semantic equivalence between two given texts. However, in the STS task, very few systems provide evidence of the contribution of syntactic structure in its overall performance. Following the work in the literature (Vo and Popescu, 2015), we would like to make a deeper study and analysis whose contribution consists of two folds, on the STS 2012, 2013, and 2014 datasets (1) we assess the impact of syntactic structure towards the overall performance, and (2) analyze the behavior of syntactic structure in each score range of STS semantic scale. We consider three methods 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 reason for this selection consists of two folds: (1) all these approaches use the syntactic parsing as a source for learning syntactic struc-

[1] http://www.cs.york.ac.uk/semeval-2012/task6

ture and information, (2) we compare two well-known groups of method for learning syntactic structure: tree kernel and generalization.

The remainder of the paper is as follows: Section 2 introduces three approaches to exploit the syntactic structure in STS task, Section 3 describes Experimental Settings, Section 4 discusses about the Evaluations and Discussion, Section 5 is the Related Work, and Section 6 is the Conclusions and Future Work.

2 Three Approaches for Learning the Syntactic Structure

In this section, we describe three approaches exploiting the syntactic structure to be used in the STS task: **Syntactic Tree Kernel** (Moschitti, 2006), **Syntactic Generalization** (Galitsky, 2013), and **Distributed Tree Kernel** (Zanzotto and Dell'Arciprete, 2012). They learn the syntactic information either from the dependency or constituency parse trees. Table 1 shows a side-by-side comparison between three approaches for learning syntactic structures.

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 (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, the STS corpus does not seem suitable. As the STS corpus contains several different datasets derived from different sources (see Table 2) which carry a large variety of syntactic structure representations, but lack of learning examples from sentence pairs due to different sentence structures. 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.[2] This corpus is split into Training set (4,076 pairs) and Testing set (1,725 pairs).

We use Stanford Parser[3] to obtain the dependency parsing from sentence pairs. Then we use the machine learning tool svm-light-tk 1.5 which uses Tree Kernel approach to learn the similarity of syntactic structure to build a binary classifying model on the Train dataset.[4] 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 evaluate this model on the STS data to predict the semantic similarity between sentence pairs. The output predictions are probability confidence scores in [-1,1], corresponds to the probability of the label to be positive.

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

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

[3] http://nlp.stanford.edu/software/lex-parser.shtml

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

Properties	STK	SG	DTK
Method	- tree kernel	- least general generalization	- distributed tree kernel
Parsing	- dependency parse	- constituency parse	- dependency parse
Function	- computes the number of common partial trees between trees T1 & T2	- computes the most specific generalization between two expressions	- uses a linear complexity algorithm to compute vectors for trees

Table 1: Methods Comparison.

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.

SG uses least general generalization (also called anti-unification) (Plotkin, 1970) to anti-unify texts. Given two terms E_1 and E_2, it produces a more general one E that covers both rather than a more specific one as in unification. Term E is a generalization of E_1 and E_2 if there exist two substitutions σ_1 and σ_2 such that $\sigma_1(E) = E_1$ and $\sigma_2(E) = E_2$. The most specific generalization of E_1 and E_2 is called anti-unifier. Technically, two words of the same Part-of-Speech (POS) may have their generalization which is the same word with POS. If lemmas are different but POS is the same, POS stays in the result. If lemmas are the same but POS is different, lemma stays in the result. The example for finding a commonality between two expressions as below:

- camera with digital zoom.
- camera with zoom for beginners.

Then, we can use logic predicates to express the meanings as:

- *camera(zoom(digital), AnyUser)*
- *camera(zoom(AnyZoom), beginner)*

where variables (empty values, not specified in the expressions) are capitalized. Given the above pair of formulas, the unification computes their most general specialization *camera(zoom(digital), beginner)*, while the anti-unification computes their most specific generalization, *camera(zoom(AnyZoom), AnyUser)*.

At syntactic level, we have generalization of two noun phrases as: *{NN-camera, PRP-with, [digital], NN-zoom [for beginners]}*. Then, the expressions in square brackets are eliminated since they occur in one expression and do not occur in another. As a result, we obtain *{NN-camera, PRP-with, NN-zoom]}*, which is a syntactic analog as the semantic generalization above.

We use the toolkit "relevance-based-on-parse-trees" to measure the similarity between two sentences by finding a set of maximal common subtrees, using representation of constituency parse trees via chunking.[5]

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. We extract the parsing by the Stanford Parser and use the software "distributed-tree-kernels" to produce the distributed trees.[6] Then, we compute the Cosine similarity between the vectors of distributed trees of each sentence pair.

3 Experiments

In this section, we describe the STS datasets that we experiment with several different settings in order to evaluate the impact of each syntactic structure approach and in combination with other features in our baseline system.

3.1 Datasets

The STS dataset (English STS) consists of several datasets in STS 2012, 2013 and 2014 (Agirre et al., 2012; Agirre et al., 2013; Agirre et al., 2014). Each sentence pair is annotated the semantic similarity score in the scale [0-5]. Table 2 shows the

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

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

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 2: Summary of STS datasets in 2012, 2013, and 2014.

summary of STS datasets and sources over the years. We use four settings for training and evaluation as below:

- Setting 1: train on STS 2012 Train datasets, and evaluate on STS 2012 Test datasets.
- Setting 2: train on all STS 2012 datasets, and evaluate on STS 2013 datasets.
- Setting 3: train on all STS 2012 and 2013 datasets, and evaluate on STS 2014 datasets.
- Setting 4: to avoid the fact that STS provides train and test data derived from different sources, which may requires domain adaptation technique, we performs 10-fold cross validation on each year datasets in 2012, 2013 and 2014; and on all STS datasets together, to speculate the behavior of syntactic structure on each score range of STS, i.e [0-1], [1-2], [2-3], [3-4], and [4-5].

3.2 Baseline

In order to assess the impact of syntactic structure in the STS task, we not only examine the syntactic structure alone, but also combine it with features learned from the most common approach, bag-of-words. Therefore, we use a bag-of-word baseline to evaluate the performance of syntactic approaches. This baseline is the basic one used for evaluation in the STS task, namely **tokencos**. It 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.

3.3 Settings

In this section, we present other eight different settings for experimenting the contribution of syntactic structure individually and in combination with typical similarity features to the overall performance of computing similarity score on STS datasets, as follows:

- The STK (2), SG (3), and DTK (4) assess the individual contribution of Syntactic Tree Kernel, Syntactic Generalization and Distributed Tree Kernel approaches, respectively.
- The (2), (3) & (4), assesses the overall contribution of syntactic structure of three approaches.
- The (1) & (2), (1) & (3), and (1) & (4), examine the contribution of each syntactic approach with feature learned from bag-of-words approach in the baseline tokencos.
- The (1), (2), (3) & (4), is the combination of all three approaches with the baseline tokencos.

The output of each approach is normalized to the standard semantic scale [0-5] of STS task to evaluate its standalone performance, or combined with result from other approaches using a simple Linear Regression model in WEKA machine learning tool (Hall et al., 2009) with default configurations and parameters.

4 Evaluations and Discussion

The results reported here are obtained by Pearson correlation, which is the official measure used in STS task.[7]

4.1 Evaluation on STS 2012

Only STS 2012 datasets consists of several of test datasets which have designated training data. Table 3 shows that each method behaves differently on different dataset and results in both positive and negative correlation to human judgment. Only the STK and SG outperform the baseline on *MSRpar* and *MSRvid* by large margins of 16% and 13%, respectively. All methods perform lower than the baseline on most of the datasets, even negative results.

The combination of three approaches does not improve the overall performance on each dataset

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

Settings	MSRpar	MSRvid	SMTeuroparl	OnWN	SMTnews	Mean
Baseline (1)	0.4334	0.2996	0.4542	0.5864	0.3908	0.4329
STK (2)	**0.5988**	0.0916	*-0.1647*	0.0621	0.0986	0.1373
SG (3)	*-0.08*	**0.5354**	0.2095	0.4738	0.3395	0.2956
DTK (4)	0.0205	0.1139	*-0.3427*	*-0.2466*	*-0.1217*	*-0.1153*
(2), (3) & (4)	**0.5832**	0.2339	*-0.0895*	0.2625	0.1897	0.236
(1) & (2)	**0.6546**	0.285	0.2615	0.4687	0.323	0.3986
(1) & (3)	0.1812	**0.3889**	0.4539	**0.5964**	**0.436**	0.4113
(1) & (4)	0.4326	**0.4421**	0.044	0.4986	0.3074	0.3449
(1), (2), (3) & (4)	**0.6447**	**0.4072**	0.0614	0.4799	0.3159	0.3818

Table 3: Results on STS 2012 datasets (represent Pearson correlation with human judgments).

Settings	FNWN	headlines	OnWN	SMT	Mean
Baseline (1)	0.2146	0.5399	0.2828	0.2861	0.3309
STK (2)	0.0458	0.0286	0.0365	*-0.0329*	0.0195
SG (3)	**0.2154**	0.4434	**0.4558**	0.2675	**0.3455**
DTK (4)	*-0.0516*	*-0.1241*	0.1247	*-0.2577*	*-0.0772*
(2), (3) & (4)	0.0991	0.2981	0.2585	0.2096	0.2163
(1) & (2)	0.1398	0.4937	0.2634	0.2321	0.2823
(1) & (3)	**0.2307**	**0.5676**	**0.3617**	**0.3091**	**0.3673**
(1) & (4)	0.2005	**0.547**	**0.3541**	0.181	0.3207
(1), (2), (3) & (4)	0.1651	0.5355	**0.3585**	0.2145	0.3184

Table 4: Results on STS 2013 datasets (represent Pearson correlation with human judgments).

Settings	deft-forum	deft-news	headlines	images	OnWN	tweet-news	Mean
Baseline (1)	0.3531	0.5957	0.5104	0.5134	0.4058	0.6539	0.5054
STK (2)	0.1163	0.2369	0.0374	*-0.1125*	0.0865	*-0.0296*	0.0558
SG (3)	0.2816	0.3808	0.4078	0.4449	**0.4934**	0.5487	0.4262
DTK (4)	0.0171	0.1	*-0.0336*	*-0.109*	0.0359	*-0.0986*	*-0.0147*
(2), (3) & (4)	0.2402	0.3886	0.3233	0.2419	**0.4066**	0.4489	0.3416
(1) & (2)	0.3408	0.5738	0.4817	0.4184	0.4029	0.6016	0.4699
(1) & (3)	**0.3735**	0.5608	**0.5367**	**0.5432**	**0.4813**	**0.6736**	**0.5282**
(1) & (4)	**0.3795**	**0.6343**	**0.5399**	0.5096	**0.4504**	**0.6539**	**0.5279**
(1), (2), (3) & (4)	**0.3662**	0.5867	**0.5265**	0.464	**0.4758**	0.6407	**0.51**

Table 5: Results on STS 2014 datasets (represent Pearson correlation with human judgments).

or overall result. However, it partially covers the weakness of each method on each dataset.

The combination of each method with the bag-of-word approach returns both increase and decrease results. However, this combination obtains the best performance on the dataset *MSRvid* whereas two settings outperform the baseline and another is very close to the baseline. Among the three methods, SG seems to integrate well with the bag-of-word approach in which its combinations outperform the baseline on three datasets *MSRvid, OnWN*, and *SMTnews*. However, none of these settings equals to the baseline in overall result.

4.2 Evaluation on STS 2013

Table 4 shows that none of the approach individually equals to the baseline on any dataset, except the SG is slightly better than the baseline

Settings	STS 2012	STS 2013	STS 2014	STS 2012-2013-2014
Baseline (1)	0.3147	0.3541	0.4353	0.3826
STK (2)	**0.3267**	0.2652	0.0019	0.2335
SG (3)	0.2613	**0.429**	0.4268	0.3583
DTK (4)	0.0842	*-0.0543*	*-0.0428*	0.0184
(2), (3) & (4)	**0.3954**	**0.4662**	0.4271	**0.4041**
(1) & (2)	**0.4316**	**0.452**	0.4346	**0.4361**
(1) & (3)	**0.3544**	**0.4498**	**0.4921**	**0.4353**
(1) & (4)	**0.3905**	**0.3754**	**0.4617**	**0.4223**
(1), (2), (3) & (4)	**0.4634**	**0.5**	**0.5082**	**0.4796**

Table 6: Cross Validation Results on STS datasets (represent Pearson correlation with human judgments).

on *FNWN*. The DTK the returns the worst performance (negative results) on three datasets *FNWN, headlines* and *SMT*.

The combination of three approaches brings no improvement over the baseline, but it covers the weakness from DTK on all datasets.

The combination between each method with the bag-of-word approach covers the weakness of each method itself (no more negative result appears). This combination especially works very well on the datasets *headlines* and *OnWN* with two times outperform the baseline. SG proves to be the best method integrate with the bag-of-word approach by obtaining 3% better than the baseline.

4.3 Evaluation on STS 2014

Table 5 shows that none of these three methods equals to the baseline. Though the STK and DTK both use the tree kernel approach, just different representations, in overall, the STK performs better than DTK on most of datasets. STK and DTK return negative results on the datasets images and tweet-news whereas the SG obtains quite good result.

The combination of three approaches does not collaborate well on STS datasets, it even decreases the overall performance of the best method SG by a large margin of 8%. Finally, this combination does not make any improvement over the baseline. Thus, this combination of syntactic approaches cannot solve the STS task.

The combination of syntactic information and bag-of-word approach improves the performance on many datasets over the baseline. On STS, SG and DTK are benefited from the combination by outperforming the baseline around 2%. SG is the best method to integrate with the bag-of-word on all STS datasets. The combination of three methods with the bag-of-word returns 0.5% and 2% better results than the baseline.

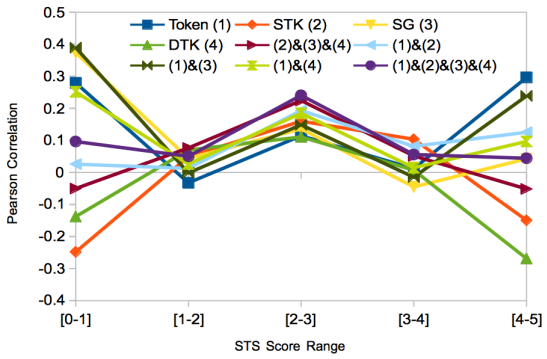

Figure 1: STS2012 Cross-Validation Analysis.

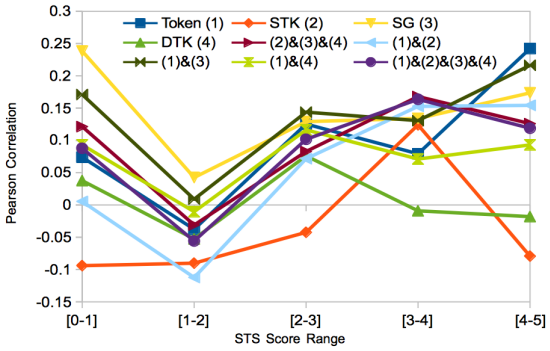

Figure 2: STS2013 Cross-Validation Analysis.

4.4 Evaluation by Cross-Validation

Table 6 shows that each approach usually performs lower than the baseline, but its combinations with baseline outperform the baseline itself in most of cases. In the semantic scale from 0 (dissimilar) to 5 (completely equivalent), we speculate the behavior of syntactic structure and its impact to predicting correct semantic similarity scores in STS.

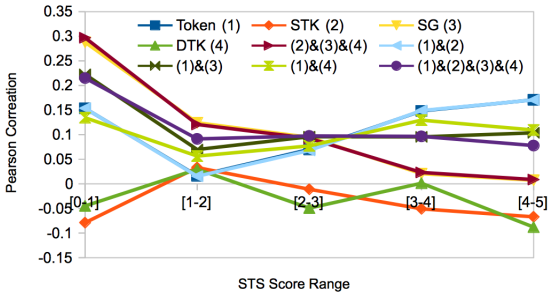

Figure 3: STS2014 Cross-Validation Analysis.

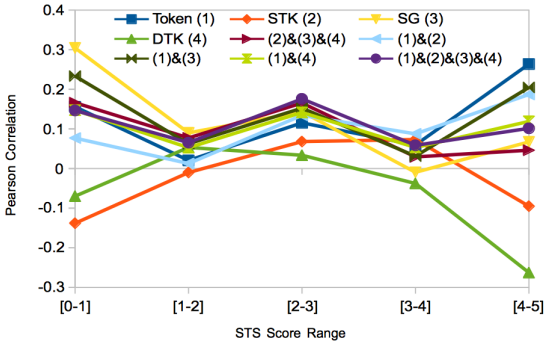

Figure 4: STS2012, 2013, and 2014 Cross-Validation Analysis.

Cross-validation on STS 2012. Figure 1 shows that syntactic structure in different settings results high correlation (either positive or negative) mostly in three score ranges [0-1] (dissimilar, or not equivalent but same topic), [2-3] (not equivalent but share some detailes, or roughly equivalent but some important information missing), and [4-5] (mostly equivalent, or completely equivalent).

Cross-validation on STS 2013. Similar to STS 2012, Figure 2 shows that syntactic structure obtains high correlation (both positive and negative) mostly in three score ranges [0-1], [2-3], [4-5], and also [3-4] (roughly equivalent, or mostly equivalent).

Cross-validation on STS 2014. Figure 3 shows that the impact of syntactic structure presents most significantly in the range [0-1], and almost equivalently in other ranges [1-2], [2-3], [3-4], and [4-5].

Cross-validation on the combination of STS 2012, 2013 and 2014. In overall, Figure 4 confirms the significance of syntactic structure mostly in three score ranges [0-1], [2-3], and [4-5].

All the cross-validation results reveal some interesting behaviors of syntactic structure on STS datasets:

- The bag-of-word approach mostly has positive correlation in all ranges, but highest in [0-1] and [4-5].
- STK always obtains highly negative correlation on STS datasets in the ranges [0-1] and [4-5], but it results unpredictable correlation (both positive and negative) in other ranges.
- DTK seems to have similar behavior to STK but more fluctuate. This confirms that since these two approaches use the same method (tree kernel), they tend to have similar behaviors.
- In contrast, SG always returns positive correlation in all ranges (except the a very slightly negative correlation in range [3-4] on STS 2012), but highest in [0-1] and [4-5]. The trends confirm that SG usually has highest correlation in [0-1], [1-2], and [2-3].
- The combination of three approaches tends to correlate closely to the trend of SG.
- The combination of three approaches with bag-of-word behaves similarly to the bag-of-word itself, but sometimes slightly turns down in ranges [0-1] and [4-5]. This setting usually helps to improve the overall performance in ranges [1-2], [2-3], and [3-4].
- The combination of each approach with the bag-of-word returns similar behavior to the bag-of-word itself. Sometimes, this setting slightly improves the performance of bag-of-word in different ranges.

In conclusion, despite the fact that we experiment different methods to exploit syntactic information on different datasets derived from various data sources, the results confirm the positive impact of syntactic structure in the overall performance on STS task. 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. Among these three methods, the SG shows to be the most effective one to exploit syntactic and semantic information individually or collaboratively with the bag-of-word approach. Moreover, the experiment results show that the bag-of-word approach is still a very strong and effective method to learn the semantic information in the STS task; and its combination with syntactic approaches returns improvement in the overall performance.

5 Related Work

Complex logical representations are usually used for semantic inference tasks. Nevertheless, due to the high cost of constructing complex logical representations, practical applications usually support shallower level of lexical or lexical-syntactic representations. The literature (Bar-Haim et al., 2007) proposed an approach operating on syntactic trees directly. Basically, entailment rules are used to infer new trees and provide unified representation for various inference types. Manual and automatic methods are used to generate rules and cover generic linguistic structures as well as specific lexical-based inferences. However, current works focus on syntactic tree transformation in graph learning framework (Chakrabarti and Faloutsos, 2006), (Kapoor and Ramesh, 1995), treating various phrasings for the same meaning in a more unified and automated manner.

In the STS task, several attempts are made to exploit the syntactic structure to solve the task. In the literature (Islam and Inkpen, 2008), a simple method is deployed to examine the shallow syntactic relation between two given sentences towards computing their semantic similarity, namely, Common Word Order Similarity between Sentences. The basic idea is that if the two texts have some words in common, we can measure how similar the order of the common-words is in the two texts (if these words appear in the same order, or almost the same order, or very different order). This similarity is determined by the normalized difference of common-word order.

The Takelab system (Šarić et al., 2012) which is ranked 2nd at STS 2012 used two methods to learn the syntactic structure for computing the semantic similarity between given sentences. (1) Syntactic Roles Similarity uses dependency parsing to identify the lemmas with the corresponding syntactic roles in the two sentences. Given two sentences, the similarity of words or phrases that have the same syntactic roles may indicate their overall semantic similarity (Oliva et al., 2011). (2) Syntactic Dependencies Overlap computes the overlap of the dependency relations between two given sentences. A similar measure has been proposed in (Wan et al., 2006) in which if two syntactic dependencies share the same dependency type, governing lemma and dependent lemma, they are considered equal.

At STS 2013, the iKernels system (Severyn et al., 2013) proposed the idea of using relational structures to jointly model text pairs. They defined two new relational structures based on constituency and dependency trees. In constituency tree, each sentence is represented by its constituency parse tree. Then a special REL tag is used to link the related structures and encode the structural relationships between two sentences. In contrast, the dependency relations between words are used to derive an alternative structural representation in which words are linked in a way that words are always at the leaf level. The part-of-speech tags between the word nodes and nodes carrying their grammatical role are also plugged in. Then the REL tag is used to establish relations between tree fragments. Finally, the Partial Tree Kernel is used to compute the number of common substructures.

6 Conclusions and Future Work

In this paper, we deploy three different approaches to exploit and evaluate the impact of syntactic structure in the STS task. We use a bag-of-word baseline which is the official baseline of STS task for the evaluation. We also evaluate the contribution of each syntactic structure approach integrated with the bag-of-word approach in the baseline. From our observation, for the time being with recent proposed approaches, the results in Tables 3, 4, and 5 shows that the syntactic structure does contribute and play a part individually and together with typical similarity approaches for computing the semantic similarity scores between given sentence pairs. However, compared to the baseline, the contribution of syntactic structure is not significant to the overall performance. For future works, 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 task.

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 *Proceedings of the *SEM 2013: The Second Joint Conference on Lexical and Computational Semantics. Association for Computational Linguistics*. Citeseer.

Eneko Agirre, 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.

Roy Bar-Haim, Ido Dagan, Iddo Greental, and Eyal Shnarch. 2007. Semantic inference at the lexical-syntactic level. In *Proceedings of the National Conference on Artificial Intelligence*, volume 22, page 871. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.

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

Deepayan Chakrabarti and Christos Faloutsos. 2006. Graph mining: Laws, generators, and algorithms. *ACM Computing Surveys (CSUR)*, 38(1):2.

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.

Aminul Islam and Diana Inkpen. 2008. Semantic text similarity using corpus-based word similarity and string similarity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2(2):10.

Sanjiv Kapoor and Hariharan Ramesh. 1995. Algorithms for enumerating all spanning trees of undirected and weighted graphs. *SIAM Journal on Computing*, 24(2):247–265.

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

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

Jesús Oliva, José Ignacio Serrano, María Dolores del Castillo, and Ángel Iglesias. 2011. Symss: A syntax-based measure for short-text semantic similarity. *Data & Knowledge Engineering*, 70(4):390–405.

Gordon D Plotkin. 1970. A note on inductive generalization. *Machine intelligence*, 5(1):153–163.

Frane Šarić, Goran Glavaš, Mladen Karan, Jan Šnajder, and Bojana Dalbelo Bašić. 2012. Takelab: Systems for measuring semantic text 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 441–448. Association for Computational Linguistics.

Aliaksei Severyn, Massimo Nicosia, and Alessandro Moschitti. 2013. ikernels-core: Tree kernel learning for textual similarity. In *Proceedings of the Second Joint Conference on Lexical and Computational Semantics*, volume 1, pages 53–58. Citeseer.

Ngoc Phuoc An Vo and Octavian Popescu. 2015. A preliminary evaluation of the impact of syntactic structure in semantic textual similarity and semantic relatedness tasks. In *NAACL-HLT 2015 Student Research Workshop (SRW)*, page 64.

Stephen Wan, Mark Dras, Robert Dale, and Cécile Paris. 2006. Using dependency-based features to take the "para-farce" out of paraphrase. In *Proceedings of the Australasian Language Technology Workshop*, volume 2006.

Fabio Massimo Zanzotto and Lorenzo Dell'Arciprete. 2012. Distributed tree kernels. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*.