Samsung: Align-and-Differentiate Approach to Semantic Textual Similarity

Lushan Han, Justin Martineau, Doreen Cheng and Christopher Thomas
Samsung Research America
665 Clyde Avenue
Mountain View, CA 94043, USA
{lushan.han, justin.m, doreen.c, c2.thomas}@samsung.com

Abstract

This paper describes our Align-and-Differentiate approach to the SemEval 2015 Task 2 competition for English Semantic Textual Similarity (STS) systems. Our submission achieved the top place on two of the five evaluation datasets. Our team placed 3rd among 28 participating teams, and our three runs ranked 4th, 6th and 7th among the 73 runs submitted by the 28 teams. Our approach improves upon the UMBC *PairingWords* system by semantically differentiating distributionally similar terms. This novel addition improves results by 2.5 points on the Pearson correlation measure.

1 Introduction

Since its inception in 2012, the annual Semantic Textual Similarity (STS) task has attracted and increasing amount of interest in the NLP community. The task is to measure the semantic similarity between two sentences using a scale ranging from 0 to 5 (Agirre et al., 2012; Agirre et al., 2013; Agirrea et al., 2014). In this task, 0 means *unrelated* and 5 means *complete semantic equivalence*. For example, the sentence "China's new PM rejects US hacking claims" is semantically equivalent to the sentence "China Premier Li rejects 'groundless' US hacking accusations" even though there are many word level differences between the two sentences.

Improvements in the STS task can advance or benefit many research areas, such as paraphrase recognition (Dolan et al., 2004), automatic machine translation evaluation (Kauchak and Barzilay, 2006), ontology mapping and schema matching (Han, 2014), Twitter search (Sriram et al., 2010), image retrieval by captions (Coelho et al., 2004) and information retrieval in general.

Measuring semantic similarity is difficult because it is relatively easy to express the same idea in very different ways. Both word choice and word order can have a great impact on the semantics of a sentence, or not at all. For example, the sentences "A woman is playing piano on the street" and "A lady is playing violin on the street" have a semantic similarity score of only 2, because pianos are not violins so the two events in the sentences must be different. This is problematic because common solutions, such as bag-of-words representations, parse trees, and word alignments measure word choice and word order. We improve upon existing word choice approaches with better measures to semantically differentiate distributionally similar terms, and by using these measures to also improve the word alignment.

Our solution is an *Align-and-Differentiate* approach, in which we greedily align words between sentences, before penalizing non-matching words in the differentiate-phase. Our system improves upon the successful UMBC *PairingWords* system by about 2 points of Pearson's Correlation measure. The success of the *PairingWords* system is largely due to their high-quality distributional word similarity model[1] described in (Han et al., 2013). The distributional similarity model can tell that "woman" and "lady" in the above example are highly similar, which is usually correct, but it also says that "pi-

[1]See `http://swoogle.umbc.edu/SimService/` for a demo.

Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), pages 172–177,
Denver, Colorado, June 4-5, 2015.

ano" and "violin" are very similar, which in many contexts is incorrect. While distributional similarity measures can be criticized for producing high similarity scores for antonyms and contrasting words, we find that this property is actually advantageous when performing word alignment between two sentences. We take advantage of this property by first aligning with distributional similarity, and then differentiate by penalizing alignments of words that are semantically disjoint (Ex: antonyms). This technique to first align and then differentiate is our key improvement.

The remainder of the paper proceeds as follows. Section 2 briefly revisits the UMBC *PairingWords* system. Section 3 presents our new *Align-and-Differentiate* approach. Section 4 presents and discusses our results.

2 UMBC PairingWords System

The *PairingWords* system (Han et al., 2013) uses a state-of-the-art word similarity measure to align words in the sentence pair and computes the STS score using a simple metric that combines individual term alignment scores.

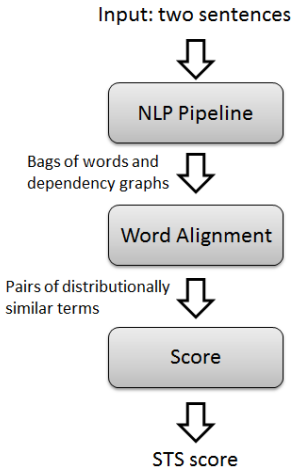

Figure 1: Overview of UMBC PairingWords system.

2.1 Precompute Word Similarities

First, a distributional model was built on an English corpus[2] of three-billion words and separated into paragraphs. Words are POS tagged and lemmatized. A small context window of ± 4 words is used to count word co-occurrences. The vocabulary has a size of 29,000 terms, which includes primarily open-class words (i.e. nouns, verbs, adjectives and adverbs). Singular Value Decomposition (SVD) (Landauer and Dumais, 1997; Burgess et al., 1998) has been used to reduce the 29K word vectors to 300 dimensions. The distributional similarity between two words is measured by the cosine similarity of their corresponding reduced word vectors. The distributional similarity is then enhanced with WordNet (Fellbaum, 1998) relations in eight categories (See (Han et al., 2013)). Finally it is wrapped with surface similarity modules to handle the matching of out-of-vocabulary words.

[2]The UMBC WebBase corpus is available for download at `http://ebiq.org/r/351`

2.2 NLP Pipeline

The Stanford POS tagger is applied to tag and lemmatize the input sentences. A predefined vocabulary, POS tags, and regular expressions are used to recognize multi-word terms including noun and verb phrases, proper nouns, numbers and time. Stop words are ignored. The stop word list was augmented with adverbs that occurred more than $500,000$ times in the corpus.

2.3 Word Alignment Between Two Sentences

The alignment function g for a target word w in one sentence S is simply defined as its most similar word w' in the other sentence S' with respect to the aforementioned word similarity measure. See Equation 1.

$$g(w) = \underset{w' \in S'}{argmax}\ sim(w, w') \qquad (1)$$

2.4 Score

The *PairingWords* systems yield an STS score in the range [0, 1] with a linearly scaled definition corresponding to the standard STS score. This score is computed using the word level semantic similarity of the aligned words. The *PairingWords* system uses a similarity threshold to decide whether a term can be aligned. If a term cannot be aligned then a penalty is imposed. Therefore, the *PairingWords* STS score is the result of subtracting the penalty score P from the overall term alignment score T, which is defined in Equation 2.

$$T = \frac{\sum_{t \in S_1} sim(t, g(t))}{2 \cdot |S_1|} + \frac{\sum_{t \in S_2} sim(t, g(t))}{2 \cdot |S_2|} \quad (2)$$

where S_1 and S_2 are the sets of words/terms in two input sentences.

3 Align-and-Differentiate Approach

Our system extends the UMBC *PairingWords* system by differentiating distributionally similar terms, resulting in a conceptually new framework to tackle the STS challenge. Figure 2 illustrates our system. After preprocessing there are four main algorithms: align, differentiate, score, and rescore.

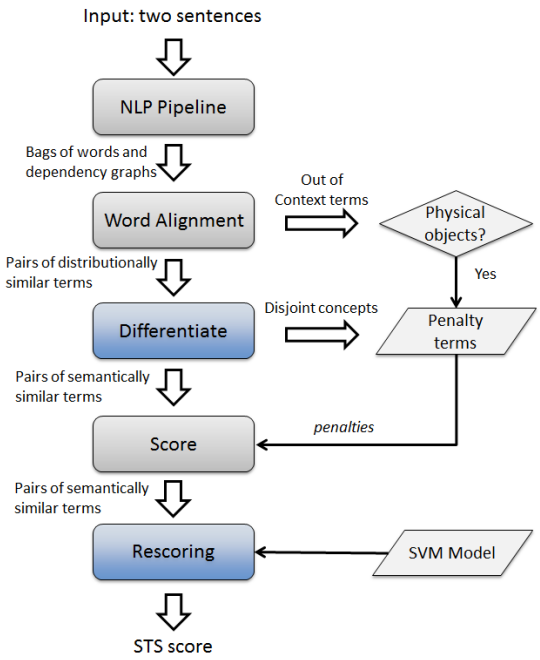

Figure 2: Our system overview. Blue components (Differentiate and Rescore) mark the most novel additions to the PairingWords system.

3.1 Precompute Word Similarities

We reused the distributional model built for the UMBC PairingWords system.

3.2 NLP Pipeline

In addition to the basic NLP techniques used by the *PairingWords* in Section 2.2 we use the Stanford dependency parser to translate the input sentences into their dependency graph representation.

3.3 Word Alignment Between Two Sentences

For alignment we upgraded the *PairingWords* approach (see Equation 1) with candidate disambiguation. If multiple candidates (ambiguity) exist, we use their neighboring words in the sentences and dependency graphs to carry out disambiguation. For two mapping candidates, we found their neighboring words in terms of dependency relations. Then we choose the candidate with the highest neighbor similarity. This alignment method is directional. In domains for which we have high confidence that the dependency parser will correctly parse both sentences, we require mutual agreement in both directions. Mutual alignment is computed by finding g such that $g(w) = w'$ and $g(w') = w$.

The similarity function $sim(w, w')$ is the word similarity function described in Section 2.1.

Following the *PairingWords* system, we use a similarity threshold of .05 to determine whether a vocabulary word[3] has at least some minimum similarity with any of the words in the other sentence. We call a word *Out Of Context (OOC)* if the threshold is not satisfied. The appearance of OOC words could be an indicator of different sentence semantics, as illustrated in the example "A beautiful red car" vs. "A beautiful red rose" where "car" is an OOC word with respect to the other sentence. The impact of OOC words to semantic equivalence is disproportionately high. Therefore, we penalize semantic similarity scores in proportion to the number of OOC words.

However, we observed that if OOC words occur because there are additional details, then these words should not be penalized. For example, in the two sentences "Matt Smith to leave Doctor Who after 4 years" and "Matt Smith quits Doctor Who", the word 'year' is an OOC word that does not significantly reduce the semantic equivalence. We found that many of these extraneous and benign OOC words do not represent physical objects, i.e. something that can be touched. Hence, we chose to only penalize OOC words that are physical objects.

[3] A vocabulary word means a word in our vocabulary of 29k words

WordNet has a synset *physical objects* and we use its descendants to collect the set of physical objects.

3.4 Differentiate

This subsection defines and then describes how we identify *Disjoint Similar Concepts*.

The *semantic similarity* of two words is the degree of semantic equivalence between the two words. We may also say, it is the ability to substitute one term for the other without changing the meaning of a sentence.

Many distributionally similar terms are not semantically similar. Examples include "good" vs "bad", "cat" vs "dog", "Thuesday" vs "Monday", "France" vs "England" and etc. Existing research on distributional models has mainly been focused on studying antonyms or contrasting words (Mohammad et al., 2008; Scheible et al., 2013; Mohammad et al., 2013). However, as shown by the above examples, the scope of distributionally similar but not semantically similar terms goes far beyond antonyms. Hereafter, we refer to this new category of terms as *Disjoint Similar Concepts (DSCs)*.

To the best of our knowledge, collecting *Disjoint Similar Concepts* is a novel research problem. General statistical methods are not easily available, but we can extract such information from human-crafted ontologies, such as WordNet. For this work, we identify *Disjoint Similar Concepts* as siblings under a common parent in an ontology, such as WordNet. For example, in the electronics domain, we can assert that *smart phone* and *tablet* are *DSCs* if they are siblings with the same parent *electronics* in the ontology.

We use a semi-automatic method to produce several sets of potential *DSCs* for our STS system. The sets include animals, countries, vehicles, weekdays, colors and etc. First, we decide what types of *DSCs* are likely to appear in a dataset. For example, animals and vehicles will likely appear in the *images* training dataset.

We penalize each aligned word pair that has *Disjoint Similar Concepts*. If both words are antonyms then they are *DSCs*. If both words share the same hypernym in WordNet, and that hypernym is a potential *DSC*, then they are *DSCs*. Otherwise, the concepts are considered semantically similar.

3.5 Score

We create a base similarity score E_i, and then apply penalties for OOC words O_i and *Disjoint Similar Concepts* D_i.

$$T_i = \frac{E_i - O_i - D_i}{2 \cdot |S_i|} \qquad i \in \{1, 2\} \quad (3)$$

$$E_i = \sum_{\langle t, g(t)\rangle \in SS_i} sim(t, g(t)) \qquad i \in \{1, 2\} \quad (4)$$

$$O_i = \sum_{t \in OOC_i} \alpha(t) \qquad i \in \{1, 2\} \quad (5)$$

$$D_i = \sum_{\langle t, g(t)\rangle \in DSC_i} \beta(\langle t, g(t)\rangle) \qquad i \in \{1, 2\} \quad (6)$$

$$STS = T_1 + T_2 \quad (7)$$

Our primary method of producing the STS score is shown in Equations 3 to 7. The method is based on the directional alignment function described in Section 3.3. E_i is the base score where i indicates the alignment direction and SS_i represents the collection of pairs of semantically similar terms for direction i. O_i is the sum of penalties applied to OOC terms for direction i. In our current system, the function $\alpha(t)$ has a constant value 1.0. D_i is the sum of penalties applied to *Disjoint Similar Concepts* for direction i. We normally set $\beta(\langle t, g(t)\rangle)$ to 0.5 but we can also tune β coefficient depending on different types of *Disjoint Similar Concepts* (e.g.*animal* and *color*), if a training dataset is available.

3.6 Rescore by Learning STS Offset Scores

We learn an offset score to account for and correct systemic biases in the Align and Differentiate algorithm using supervised machine learning. For domains with labeled data we used bag-of-words Support Vector Machines (SVMs) in regression mode, with a linear kernel, to compute an offset score measuring the difference between our Equation 7 STS score and the gold standard training STS score. We add this offset score to the Equation 7 STS score. This process improved our Pearson Correlation scores from .7936 to .8162 on the 2014 STS data in a ten fold cross-validation setting.

The SVM was trained on a length normalized bag-of-words with additional non-normalized meta

Dataset	alpha	beta	delta
headlines (750 pairs)	0.8342 (2)	0.8342	**0.8417 (1)**
images (750 pairs)	0.8701 (2)	**0.8713 (1)**	0.8634
students (750 pairs)	0.7827 (2)	0.7819	0.7825
forums (375 pairs)	0.6589	0.6586	0.6639
belief (375 pairs)	0.7029	0.6995	0.6952
weighted mean	0.7920 (4)	0.7916 (7)	0.7918 (6)

Table 1: Pearson correlation and STS 2015 Competition Rank of our three runs on test sets.

features for (1) the length difference between sentence pairs, (2) the percentage of exact word to word matches between both sentences, and (3) the STS score produced in Equation 7. The bag-of-words feature values were calculated by taking the absolute value of the difference between the number of times a word occurred in the first sentence versus the number of occurrences in the paired sentence. The bag-of-words was created with both words and bi-gram word sequences.

4 Results and Discussion

Table 1 shows the official results of our three runs, alpha, beta and delta, in the 2015 STS task. Each entry supplies a run's Pearson correlation on a dataset and the rank of the run among all 73 runs submitted by the 28 teams. The last row shows the weighted mean and the overall ranks of our three runs.

The alpha run was produced by applying the align-and-differentiate algorithm to the five datasets with the same parameter settings. The beta run was produced without penalizing OOC terms, except for the *images* dataset. The result for penalizing OOC terms are slightly better, but are just shy of a 95% confidence interval (using paired T-tests). On the *images* dataset, we exploited dependency structure in the align and differentiate algorithm. We use the supervised ML model to rescuer our STS scores only for the delta run on the *Headlines* and *Images* datasets.

Our results on the *forums* and *beliefs* datasets were surprisingly much lower than other datasets due to the *PairingWords* system's poor baseline performance on these datasets as shown in Table 2. We speculate that this drop in performance is caused by the *PairingWords* system ignoring words that are not nouns, verbs, adjectives and limited adverbs. These include common meaningful words such as "how" and "why" in both datasets.

System	headline	image	student	forum	belief	*mean*
UMBC	.8059	.8431	.7588	.6646	.6996	.7725
alpha	.8342	.8701	.7827	.6589	.7029	.7920

Table 2: Our approach improves results by 2.5% in Pearson's correlation.

Our approach of semantically differentiating distributionally similar terms, as shown in Table 2 is a statistically significant improvement at the 95% confidence interval.

Acknowledgments

We thank Ebiquity lab, CSEE department, University of Maryland, Baltimore County for providing their 2013 STS code.

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*, pages 385–393.

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 **SEM 2013: The Second Joint Conference on Lexical and Computational Semantics*.

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.

C. Burgess, K. Livesay, and K. Lund. 1998. Explorations in context space: Words, sentences, discourse. *Discourse Processes*, 25:211–257.

T.A.S. Coelho, Pável Pereira Calado, Lamarque Vieira Souza, Berthier Ribeiro-Neto, and Richard Muntz. 2004. Image retrieval using multiple evidence ranking. *IEEE Trans. on Knowl. and Data Eng.*, 16(4):408–417.

Bill Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: exploiting massively parallel news sources. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING '04.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. The MIT Press, May.

Lushan Han, Abhay L. Kashyap, Tim Finin, James Mayfield, and Johnathan Weese. 2013.

UMBC_EBIQUITY-CORE: Semantic Textual Similarity Systems. In *Proceedings of the Second Joint Conference on Lexical and Computational Semantics*. Association for Computational Linguistics, June.

Lushan Han. 2014. *Schema Free Querying of Semantic Data*. Ph.D. thesis, University of Maryland, Baltimore County, August.

David Kauchak and Regina Barzilay. 2006. Paraphrasing for automatic evaluation. In *HLT-NAACL '06*, pages 455–462.

T. Landauer and S. Dumais. 1997. A solution to plato's problem: The latent semantic analysis theory of the acquisition, induction, and representation of knowledge. In *Psychological Review, 104*, pages 211–240.

Saif Mohammad, Bonnie Dorr, and Graeme Hirst. 2008. Computing word-pair antonymy. In *Proc. Conf. on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-2008)*, October.

Saif M. Mohammad, Bonnie J. Dorr, Graeme Hirst, and Peter D. Turney. 2013. Computing Lexical Contrast. *Computational Linguistics*, 39(July 2012):555–590.

S. Scheible, S. Schulte im Walde, and S. Springorum. 2013. Uncovering distributional differences between synonyms and antonyms in a word space model. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 489–497.

Bharath Sriram, Dave Fuhry, Engin Demir, Hakan Ferhatosmanoglu, and Murat Demirbas. 2010. Short text classification in twitter to improve information filtering. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 841–842.