UoB-UK at SemEval-2016 Task 1: A Flexible and Extendable System for Semantic Text Similarity using Types, Surprise and Phrase Linking

Harish Tayyar Madabushi, Mark Buhagiar, Mark Lee
School of Computer Science,
University of Birmingham,
Birmingham, United Kingdom.
H.T.Madabushi@cs.bham.ac.uk, Buhagiar.Mark.A@gmail.com, M.G.Lee@cs.bham.ac.uk

Abstract

We present in this paper a system for measuring Semantic Text Similarity (STS) in English. We introduce three novel techniques: the use of *Types*, methods of linking phrases, and the use of a *Surprise Factor* to generate 8,370 similarity measures, which we then combine using Support Vector and Kernel Ridge Regression. Our system out performs the State of the Art in SemEval 2015, and our best performing run achieved a score of .7094 on the 2016 test set as a whole, and over 0.8 on the majority of the datasets. Additionally, the use of Surprise, Types and phrase linking is not limited to STS and can be used across various Natural Language Processing tasks, while our method of combining scores provides a flexible way of combining variously generated Similarity Scores.

1 Introduction and Motivation

The goal of Semantic Text Similarity (STS) is to find the degree of overlap in the meaning of two pieces of text. This ranges from text fragments that are exact semantic equivalents, to others that have no semantic relation. STS has a wide variety of applications, including text summarisation (Aliguliyev, 2009), machine translation (Kauchak and Barzilay, 2006), and search optimisation (Sriram et al., 2010).

The STS task, which has been set by the SemEval conference for the past number of years (Agirre et al., 2014; Agirre et al., 2015), requires that submitted systems assign a score between 0 (the sentences are on different topics) and 5 (the sentences mean exactly the same thing) that reflects how similar two sentences are.

Most systems that tackled SemEval's STS task in previous years have involved three main approaches: The first is text alignment, based on the content words' meaning (Sultan et al., 2015; Sultan et al., 2014b). The second represents text as vectors, which are used to find the similarity score using a vector similarity metric (such as cosine). Third, machine learning approaches are used to compute multiple lexical, semantic, and syntactic features to classify each sentence pair's similarity.

We make use of both text alignments and vector representations, while limiting comparisons to words of the same *Type* (Section 4.1), a novel concept we introduce in addition to methods of phrase linking (Section 4.2) and establishing common noun importance (Section 4.3). These, combined with several different weight combinations we pick for each word Type, provide us with *8,370* semantic relation measures (Section 5). The overall algorithm for generating the several similarity measures is presented in Algorithm 1. We choose a subset of these measures using methods detailed in Section 6.1, combine them with a limited set of features and use Support Vector Regression and Kernel Ridge Regression to generate a Similarity Score (Section 6.2).

Our approach also handles definitions separately from arbitrary sentences, as we observed that their structure is significantly different. Since the test data provided this year did not contain a definition data set, this paper focuses on our generic approach, with definition similarity discussed briefly in Section 7.

Proceedings of SemEval-2016, pages 1264–1269,
San Diego, California, June 16-17, 2016.

2 Preprocessing

Due to the varied nature of the input presented we perform various data cleaning operations. We start by expansion of common contractions (e.g. "isn't") and informal contractions (e.g. "howz", "couldve"). We then perform a spell check and hyphen removal, which are conditional, in the sense that a word is not modified unless the modified form appears in the other sentence. All remaining hyphens are replaced by spaces, a method different from those that previously handled hyphens (Han et al., 2013).

We also perform case correction, as has been done previously (Hänig et al., 2015), since we observe several instances wherein sentence capitalisation is not suitable for parsing (e.g. headlines and forums).

3 Similarity Measures

We use two measures, which are boosted based on different parameters described in Section 4.

3.1 Alignments

The first measure makes use of the aligner developed by Sultan et al. (2014a), which was used to achieve State of the Art results in 2014 and 2015 (Sultan et al., 2014b; Sultan et al., 2015).

Our use of the aligner disregards sequences thus making use of the aligner more as a synonym finder, with the additional power of the Paraphrase Database (PPDB) (Ganitkevitch et al., 2013).

3.2 Word Embeddings

Word embeddings provide a method of mapping words or phrases to vectors, whose cosine distance represents semantic similarity. They have proved to be powerful in many NLP tasks, and have been used by top ranking systems at SemEval STS (Sultan et al., 2015; Hänig et al., 2015). We use word2vec[1], with the model trained by Google on the Google News dataset, through its Python interface Gensim[2].

We make use of word2vec in two distinct ways. The first is by extracting the mean of the vector representation of each word in a Type and finding its cosine similarity between the two sentences. The second is by adding the word2vec similarity scores of words not aligned within the same Type. We also provide the option of disregarding word pairs that have a score of less than 0.3, a method similar to that by Hänig et al. (2015).

[1] https://code.google.com/p/word2vec/

[2] https://radimrehurek.com/gensim/models/word2vec.html

4 Boosting Similarity

In this section, we detail the variations used to generate different similarity measures. These variations are not used simultaneously, but are instead combined as described in Algorithm 1 (Section 5), which iterates through all possible variations to generate a different similarity score associated with each combination.

4.1 Type Specific Comparison

Given a sentence pair, we calculate their similarity based only on how similar corresponding Parts-of-Speech (POS) are, a method previous systems have made use of, either implicitly (Kashyap et al., 2014; Sultan et al., 2015) or explicitly (Hänig et al., 2015).

We extend this idea by defining what we call word *Types*, which further subdivide each POS. A Type represents an abstract concept that several words can share. Consider the sentence pair "A man is sitting on a stool", "A boy is sitting on a chair". Although the words "man", "boy", "stool" and "chair" are all nouns, an effective strategy for comparing these sentences would be to compare the first two and the last two words independently, before then adding up their similarity. To achieve this we categorise words into different Types, which are then compared across sentences. In this case, such a categorisation might place the first two into the Type "Person" and the others into the category "Artifact". This problem could very easily extend to the problem of Word Sense Disambiguation, which we avoid by use of a heuristic.

We calculate the Type of a noun by the use of WordNet (Miller, 1995) hypernyms: W_1 is considered a hypernym of W_2 if $\forall e \in W_2$, e is an instance of W_1. We recursively find hypernyms until we reach a manually selected set of concepts (such as food.n.02). We manually combine sets of such concepts to define a Type. As a concrete example, we combine the WordNet concepts "communication.n.02", "food.n.02" and other similar concepts into the Type "thing_r1". As a single word can be part of several Types, based on the particular sense

of the word, we pick the most frequently occurring Type for each word.[3]

4.2 Phrase Linking

Consider sentences with the the phrases "Prime Minister" and "Prime Number". Although the word "Prime" is present in both sentences, the context in which it is being used makes this irrelevant. In this particular case, the semantic similarity of the sentences is dependent on the head of the phrase that the word "Prime" is contained in (i.e. "Minister" and "Number"). This is also the case with phrases that contain adjectives and adverbs.

We address this by finding phrases that consist of adjectives, adverbs and nouns, and varying the importance of the semantic similarity between words that are not the head of that phrase. The similarity of each word, that is part of such a phrase, but not the head of the phrase, is additionally weighted in three different ways: The first assigns a zero or one weight based on whether or not the head of the phrase is aligned, the second provides a weight based on the number of words, following this word, that are aligned in the phrase and the third simply ignores the phrase structure.

4.3 Noun Importance

Consider the following sentence pairs with relations assigned by human annotators: "A *boy* is playing a guitar.", "A *man* is playing a guitar.", rel: 3.2; and "A man is cutting up a *potato*.", "A man is cutting up *carrots*.", rel: 2.4. Although both pairs of sentences differ by exactly one noun, the first pair was considered to be more closely associated than the second. We associate this to what we call the "Surprise" and assign a value to this, which we call the "Surprise Factor".

Surprise is based on the work by Dunning (1993), who observed that the assumption of normality of data is invalid as "simple word counts made on a moderate-sized corpus show that words that have a frequency of less than one in 50,000 words make up about 20-30% of typical English language newswire reports. This 'rare' quarter of English includes many of the content-bearing words . . . "

[3]The termination concepts, corresponding Types, definitions for non-noun types, and Weights are provided online at: www.harishmadabushi.com/SemEval2016/Appendix.pdf

We define the Surprise Factor of a noun or phrase to be proportional to the number of Web Search Hits for that phrase or term, while inversely proportional to the Search Hits in the case of proper nouns. Intuitively this makes sense, as words that are more common will generate less Surprise, carry less information, and will also be more widely used on the Internet.

We incorporate this idea of Surprise by adding the option of additionally weighting nouns by the total number of Web Search Hits or Results[4]. We define, H_i to be the the number of Web Search Hits for the noun i, HT the total number of hits for all nouns $HT = \sum_{i=0}^{N} H_i$, N_i the fraction of the Search Hits that noun i captures $N_i = \frac{H_n}{HT}$, and NT the normalised total of all nouns (C) in a given sentence $NT = \sum_{i=0}^{C} N_i$. We define the Surprise of word i in terms of the above in Equation 1.

$$S_i = \frac{N_i}{NT} \tag{1}$$

5 System Overview

Algorithm 1 provides an overview of the system we use to generate the various Similarity Scores, We call each combination that generates a score a "Method". We use thirty weights for Types[3], while providing the option of dividing the scores by the number of WordNet Synsets (UseSSToWeight), which captures any dilution due to a word's different senses. We also scale word2vec scores by different values. This gives us a total of 8,370 "Methods".

In calculating the similarity score, we capture the fraction of each Type that is aligned and scale it by the weight of that Type. This is captured in Equation 2 where $score_t$ represents the Similarity Score assigned to Type t by either of the measures detailed in Section 3, $count_t$ represents the number of words of Type t in *both* sentences, w_t the weight of Type t in the current iteration, and T is the total number of Types.

$$5 \times \left(\frac{\sum_{t=0}^{T} score_t \times w_t \times 2}{\sum_{t=0}^{T} count_t \times w_t}\right) \tag{2}$$

[4]We use the Bing Web Search API: http://www.bing.com/toolbox/bingsearchapi

```
Data: Sentence Pairs
Result: List of Similarity Scores
Initialise list of similarity scores "SimScores" to empty list;
for w in Type-Weights do
    for n in NounHandleMethod do
        for av in Adjective-AdverbHandleMethod do
            for UseSearchHits in [True,False] do
                if UseSearchHits == True then
                    Calculate Similarity Score (SS) using
                      Alignments;
                    Append SS to SimScores;
                    Continue;
                for UseSSToWeight in [True, False] do
                    Calculate Similarity Score (SS) using
                      Alignments;
                    Append SS to SimScores;
            for UseWeightCutOff in [True,False] do
                for VectorCombineMethod in
                  [UseAlignments, UseMeanVector] do
                    for ScaleVortorSimBy in [ 6, 5, 4, 3, 2,
                      1, "log" ] do
                        Calculate Similarity Score (SS)
                          using Word Embeddings;
                        Append SS to SimScores;
Return SimScores (the list of similarity scores);
```

Algorithm 1: Calculating Semantic Similarity Scores

6 Combining Similarity Scores

As described above, we use variations to generate thousands of Similarity Scores, each of which we call a "Method". Each Method's performance varies depending on the input. In this section, we detail the process for combining these Methods, which is performed using either Support Vector Regression (SVR) or Kernel Ridge Regression (KRR).

6.1 Picking a Subset of Methods

We first select a subset of the Methods, which are then passed on to either the SVR or KRR model. To do this, each of our Methods is ranked using three metrics with respect to the training set: The first is by use of the Pearson Correlation (a criterion we call "Method"), the second is by the sum of the absolute error between Similarity Scores (a criterion we call "Error"). The third metric aggregates the the rankings from the two criterion described above, and we call this criterion "Combine". We select the top 50 methods using one of the three selection criterion.

6.2 Generating Similarity Scores

In addition to using scores from the chosen Methods, we add the following features to some of our submitted runs: a) a binary value to represent whether each of the sentences were case corrected, b) the length of each of the sentences, c) the number of continuous aligned or unaligned sequences, d) the maximum and minimum lengths of continuous aligned or unaligned sequences, and e) a binary value to represent alignments that are non-sequential.

It should be noted that the specific Methods we choose for use in the SVR or KRR will depend on the training data picked. We found, by testing our system using several different combinations of training data, that the best results were achieved when our system was trained on the headlines data from the years 2015, 2014 and 2013. The method selection criterion, regression model and parameters used for each of the runs submitted are detailed in Table 1. Although some of the settings are very similar (e.g. run2), we noticed that these minor changes translated to significant differences in performance.

Run	Headlines		Other Datasets	
	Model:	KRR	Model:	SVR
	Features:	False	Features:	False
	Train:	Headlines	Train:	Headlines
Run1	Picked:	Combine	Picked:	Combine
	Kernel:	Poly	C:	100
	Alpha:	50	Epsilon:	0.05
			Gamma:	9e-05
	Model:	SVR	Model:	SVR
	Features	True	Features:	True
	Train:	Headlines	Train:	Headlines
Run2	Picked:	Method	Picked:	Method
	C:	100	C:	100
	Epsilon:	0.01	Epsilon:	0.05
	Gamma:	9e-05	Gamma:	9e-06
	Model:	SVR	Model:	SVR
	Features	True	Features:	True
	Train:	Headlines	Train:	Headlines
Run3	Picked:	Method	Picked:	Combine
	C:	100	C:	100
	Epsilon:	0.01	Epsilon:	0.01
	Gamma:	9e-05	Gamma:	9e-06

Table 1: Parameters and models used for each run. The row Features represents if features were used, Train represents the training data used, and Picked represents the selection criterion (Method, Error or Combine).

7 Finding Similarities between Definitions

In order to find similarities between definitions, we first identify the word that a definition is defining. We achieve this by use of OneLook's reverse dictionary search[5], which returns a number of candidate

[5] http://www.onelook.com/reverse-dictionary.shtml

words for a given definition. For each definition, the similarity of the top 10 candidates is then computed using Word2Vec and five similarity metrics provided by WordNet: Path distance, Leacock-Chodorow, Wu and Palmer, Jiang-Conrath and Lin. The final score is scaled between 0 and 5 and averaged across the 10 candidates returned by OneLook.

We found this method of calculating similarities between definitions to be very good at telling if two definitions refer to the same word, but not ideally suited for measuring *how* similar they are. As a consequence, we found that results were clustered around 0 and 5. The system produced a Pearson correlation of 0.69 on the SemEval 2014 definitions data set.

8 Results and Analysis

Dataset	Best	Run1	Run2	Run3
Mean	.77807	.70940	.70168	.70911
postediting	.86690	.81272	.80835	.81333
ques-ques	.74705	.56040	.47904	.56451
headlines	.82749	.81894	.82352	.81894
plagiarism	.84138	.82066	.82406	.81958
ans-ans	.69235	.52460	.55217	.52028

Table 2: Performance on the 2016 STS Test Set

We list the performance of our system in Table 2. Our system's poor performance on the ans-ans and ques-ques datasets can be attributed to our choice of training data, which, although well suited for previous years, was not well suited for these datasets.

However, our system produces State of the Art results on the 2015 Test Sets. A breakdown of each of the run's performance against the 2015 STS data set is provided in Table 3. We note that the results we have reported for previous State of Art for individual data sources are not the results from just the winning system but the State of Art across all Systems for that data source. Our system also achieves comparable results (0.7793) to that presented by Sultan et al. (2015) (0.779) on the 2014 STS dataset. The weighted mean reported by us does not include definitions as we decided to consider them independently.

Table 4 provides a comparison of our system against the previous State of the Art for the STS 2014 data set. The overall State of Art across all data sets was reported by Sultan et al. (2015) based on their 2015 System.

Source	St. of Art	Run1	Run2	Run3
Mean	0.8015	0.8086	**0.8147**	0.8130
ans-std	0.7879	0.7919	**0.7965**	0.7953
ans-for	**0.739**	0.7184	0.7137	0.7090
belief	0.7717	0.7703	**0.7811**	0.7752
headlines	0.8417	0.8508	**0.8532**	**0.8532**
images	**0.8713**	0.8448	0.8617	0.8615

Table 3: Performance on the 2015 STS Test Set.

Source	St. of Art	Run1	Run2	Run3
Mean	0.779	0.7714	**0.7793**	0.7790
de-forum	0.504	0.5435	0.5630	**0.5636**
de-news	**0.785**	0.7718	0.7774	0.7756
headlines	0.765	**0.8082**	0.8055	0.8055
images	0.834	0.8340	0.8492	**0.8496**
OnWN	0.875	–	–	–
tweet-n	**0.792**	0.7551	0.7569	0.7573

Table 4: Performance on the 2014 STS Test Set.

9 Conclusion and Future Work

In this paper we have described the system we used for participation in the SemEval STS Monolingual Task which made use of Types, Phrase Linking, and a method of establishing common noun importance.

In the future, we intend to experiment with including features for each of the Methods during the training phase, other kinds of phrases, and different Type definitions. We also intend to use the STS data for learning the weights of different Types for use in other NLP applications.

We believe that Types have significant potential and intend to explore them in greater detail. Our immediate objectives will be in better defining types, re-categorising common noun Types based on clearer instructions to manual annotators, including finer definitions of Types for proper nouns using named entity recognition, and exploring methods of defining Types for verbs, adverbs and adjectives. We also intend to explore the use of Types in Question Classification and Question Answering.

Acknowledgments

This work was supported, in part, by the EPSRC, U.K., under grant number 1576491, and is also partially funded by the ENDEAVOUR Scholarships Scheme, which may be part-financed by the European Union - European Social Fund.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. Semeval-2014 task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 81–91.

Eneko Agirre, Carmen Banea, et al. 2015. Semeval-2015 task 2: Semantic textual similarity, english, s-panish and pilot on interpretability. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), June*.

Ramiz M Aliguliyev. 2009. A new sentence similarity measure and sentence based extractive technique for automatic text summarization. *Expert Systems with Applications*, 36(4):7764–7772.

Ted Dunning. 1993. Accurate methods for the statistics of surprise and coincidence. *Comput. Linguist.*, 19(1):61–74, March.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. Ppdb: The paraphrase database. In *HLT-NAACL*, pages 758–764.

Lushan Han, Abhay L. Kashyap, Tim Finin, James Mayfield, and Jonathan Weese. 2013. Umbc_ebiquity-core: . In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 44–52, Atlanta, Georgia, USA, June. Association for Computational Linguistics.

Christian Hänig, Robert Remus, and Xose de la Puente. 2015. Exb themis: Extensive feature extraction from word alignments for semantic textual similarity. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 264–268, Denver, Colorado, June. Association for Computational Linguistics.

Abhay Kashyap, Lushan Han, Roberto Yus, Jennifer Sleeman, Taneeya Satyapanich, Sunil Gandhi, and Tim Finin. 2014. Meerkat mafia: Multilingual and cross-level semantic textual similarity systems. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 416–423, Dublin, Ireland, August. Association for Computational Linguistics and Dublin City University.

David Kauchak and Regina Barzilay. 2006. Paraphrasing for automatic evaluation. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 455–462. Association for Computational Linguistics.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

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. ACM.

Md Arafat Sultan, Steven Bethard, and Tamara Sumner. 2014a. Back to basics for monolingual alignment: Exploiting word similarity and contextual evidence. *Transactions of the Association for Computational Linguistics*, 2:219–230.

Md Arafat Sultan, Steven Bethard, and Tamara Sumner. 2014b. DLS@CU: Sentence similarity from word alignment. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 241–246, Dublin, Ireland, August. Association for Computational Linguistics and Dublin City University.

Md Arafat Sultan, Steven Bethard, and Tamara Sumner. 2015. DLS@CU: Sentence similarity from word alignment and semantic vector composition. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 148–153, Denver, Colorado, June. Association for Computational Linguistics.