NUIG-UNLP at SemEval-2016 Task 1: Soft Alignment and Deep Learning for Semantic Textual Similarity

John P. McCrae and **Kartik Asooja** and **Nitish Aggarwal** and **Paul Buitelaar**
Insight Centre for Data Analytics
National University of Ireland, Galway
IDA Business Park
Galway, Ireland
{john.mccrae, katrik.asooja, nitish.aggarwal,
paul.buitelaar}@insight-centre.org

Abstract

We present a multi-feature system for computing the semantic similarity between two sentences. We introduce the use of soft alignment for computing text similarity, and also evaluate different methods to produce it. The main features used by our system are based on alignment and Explicit Semantic Analysis. Our system was above the median scores for 4 out of the 5 datasets at SemEval 2016 STS Task 1.

1 Introduction

Semantic textual similarity is the task of deciding if two sentences express a similar or identical meaning and requires a deep understanding of a sentence and its meaning in order to achieve high performance. Recent successful approaches to this problem have been based on the idea of creating monolingual alignments (Sultan et al., 2014a) indicating which words in each of the two sentences correspond to each other. This is quite successful in many cases where many words have the same lemma, however when synonymous and semantically similar terms are used, it is much harder to construct alignment. For this reason, we propose the use of *soft alignments*, where instead of producing a hard linking between individual words in the sentence, we instead produce a score indicating how likely one word in a sentence is to be aligned to another word in the other sentence. We examine several methods that can be used to learn these alignments including word embeddings (Mikolov et al., 2013; Pennington et al., 2014) and models based on deep learning that have been suggested for machine translation (Bahdanau et al., 2014; Cho et al., 2014). In addition, we look into recent models for sentence and document similarity that can leverage the large amount of loosely aligned text in particular those based on Explicit Semantic Analysis (Gabrilovich and Markovitch, 2007) and recent extensions aimed at generating orthogonal representations (McCrae et al., 2013; Aggarwal et al., 2015). While these novel techniques alone can achieve high performance on the task, we note that simple metrics such as the number of overlapping terms can produce reasonable performance. For added robustness we combine features based on simple metrics with novel methods explored in this work as a multi-feature regression problem, which we solve by means of an M5 Decision tree (Wang and Witten, 1996; Quinlan, 1992). The rest of the paper is structured as follows: we present our system in Section 2. We then present both our internal evaluation results and the official Task 1 results in Section 3 and finally we conclude in Section 4.

2 Methods

For convenience we assume that semantic textual similarity consists of finding a function that maps two strings, $\mathbf{a}$ and $\mathbf{b}$, of length n_a,n_b, to a single value $y \in [0, 1]$. We will use A to denote the set of words in $\mathbf{a}$ and B for the words in $\mathbf{b}$. We assume we have a dataset D consisting of triples of the form $(\mathbf{a}_i, \mathbf{b}_i, y_i)$.

2.1 Baseline features

The basic level of our system is the construction of baseline features that can be quickly and easily evaluated to quickly find candidates that are likely to be highly similar, which may be useful in search applications. Our features were partially based on those of Hänig et al. (2015), but simplified such that we do not require a part-of-speech tagger. The features we used are as follows:

Longest Common Subsequence The length in tokens of the longest common subsequence between the two sequences

n-gram Overlap The number of n-grams that occur in both sentences divided by the length of the shorter sentence.

Jaccard The Jaccard Index of the sentences using a bag-of-words model ($|A \cap B|/|A \cup B|$).

Dice The Dice Co-efficient of the sentences using a bag-of-words model ($2|A \cap B|/(|A| + |B|)$).

Containment The containment of the sentences using a bag-of-words model ($|A \cap B|/\min(|A|, |B|)$).

Sentence Length Ratio The length of the sentence using the following symmetrized ratio: $\min(|\mathbf{a}|, |\mathbf{b}|)/\max(|\mathbf{a}|, |\mathbf{b}|)$.

Average Word Length Ratio The average length of the words, using the symmetric ratio as above.

Greedy String Tiling As in Wise (1993), we used Arun Kumar Jayapal's implementation[1], where similarity is given as follows, where coverage is the number of tokens covered by the tiling.

$$\frac{2 \times \text{coverage}}{|A| + |B|}$$

Source and Target Length The length (in tokens) of each of the two strings

Keypairs For each word pair (a, b) where $a \in A$ and $b \in B$, we calculated

$$\lambda_{a,b} = \sum_{(\mathbf{a}_i, \mathbf{b}_i, y_i) \in D, a \in \mathbf{a}_i, b \in \mathbf{b}_i} y_i - \bar{y}$$

Where $\bar{y} = \frac{\sum_D y_i}{|D|}$. We took only the word pairs with the 20 highest absolute values for $\lambda_{a,b}$. The feature consisted of the number of occurrences of these keypairs.

2.2 Hard alignment

As we believe that hard alignments are also useful, we included features from hard alignment, firstly using a model based on Sultan et al. (2014b)'s system. We simplified this method using only the Word Similarity Aligner (*wsAlign*) and Named Entity Aligner (*neAlign*) parts of Sultan's method, we found that this agreed with Sultan's implementation[2] to an F-Measure of 93.8% and our observations and internal results suggested that the differences in alignments did not correspond to obvious improvements in alignment accuracy.[3]

We also use the alignments given by Jacana aligner (Yao et al., 2013)[4] directly as further alignments in our system. Jacana is a discriminatively trained monolingual word aligner that uses Conditional Random Field (CRF) model to globally decode the best alignment. It uses features based on WordNet and part-of-speech tags.

2.3 Soft Alignment

2.3.1 WordSim

Semantic relatedness measures can be directly used to compute the soft alignments between the sentences. In this approach, we compare the pretrained neural word embeddings to compute the relatedness between words across both the sentences, thus producing the soft alignment matrix. We use cosine similarity for this purpose. We use the neural embeddings[5] developed by Baroni et al. (2014).

[1]https://github.com/arunjeyapal/GreedyStringTiling

[2]https://github.com/ma-sultan/monolingual-word-aligner

[3]For example in the pair "Being against nukes does not mean not wanting to use nukes" and "Being against using nukes means not wanting to use nukes" the systems differed in the alignment of the word "use"

[4]https://github.com/chetannaik/jacana

[5]Best predict vectors on http://clic.cimec.unitn.it/composes/semantic-vectors.html

Method	Micro	Dataset						Weighted Mean
		DEFT forum	DEFT news	headlines	images	Onto-WordNet	Tweet News	
Sultan Only (-DF)	N	0.456	0.699	0.682	0.790	0.619	0.655	0.607
Sultan Only	N	0.419	0.689	0.706	0.800	0.720	0.683	0.644
Sultan + Jacana	N	0.430	0.708	0.721	0.808	0.819	0.756	0.673
Sultan + WordSim	N	0.423	0.756	0.744	0.816	0.816	0.730	0.676
Sultan + WordSim + WordPairs	N	0.434	0.743	0.743	0.825	0.833	0.743	0.689
All aligners	N	0.464	0.713	0.732	0.821	0.834	0.733	0.686
All aligners + ESA	N	0.490	0.741	0.740	0.841	0.826	0.756	0.699
All aligners + ESA	Y	0.554	0.712	0.749	0.836	0.859	0.761	0.737
Sultan + WordSim	Y	0.555	0.733	0.747	0.827	0.844	0.753	0.728

Table 1: Pearson's Correlation achieved by configurations of our system during development on the SemEval 2014 dataset

For each word pair (a, b) where $a \in A$ and $b \in B$, lets consider $\vec{a}$ and $\vec{b}$ as neural word embeddings respectively for a and b. Thus, soft alignment can be defined formally as a matrix **S** of size n_a x n_b, where $s_{ij} = \vec{a}.\vec{b}$ giving similarity between the word vectors.

2.3.2 BiLSTM

Soft alignments based only on the word similarity do not consider the word context in the sentence. Therefore, with the help of bidirectional recurrent neural network (BiRNN), we produce context-dependent word representations. BiRNN consists of a forward recurrent neural network (RNN) and a separate backward RNN to read the sentence in forward and backward directions respectively. Figure 1 shows a BiRNN representation for encoding a sentence. Here, $\vec{a_i}$ refers to the pre-trained neural word embedding, and $\vec{h_i}$ refers to the BiRNN representation for i^{th} word in the sentence, produced by the concatenation of forward and backward RNN representations. The approach being followed here is the same used in order to produce variable length sentence representation for soft attention mechanism in neural machine translation (Bahdanau et al., 2014). However, here we just use the BiRNN representations of the words for producing the soft alignments. Such representations can be considered as context-dependent representation as they carry the sentence summary at the position of each word. After producing the BiRNN representations for both the sentences, we simply compute cosine similarities between these representations at each word position across both the sentences, to produce the soft alignment matrix. Similarly as above in section 2.3.1, soft alignment can be defined as a matrix **S** of size n_a x n_b, where $s_{ij} = \vec{h}.\vec{g}$ and $\vec{h}$ and $\vec{g}$ are the BiRNN based contextual word representations. We use long short-term memory (Hochreiter and Schmidhuber, 1997, LSTM) for our experiments using our own implementation. For learning the BiLSTM based representations, we use a collection of the sentences from the previous years Semantic Textual Similarity Task.

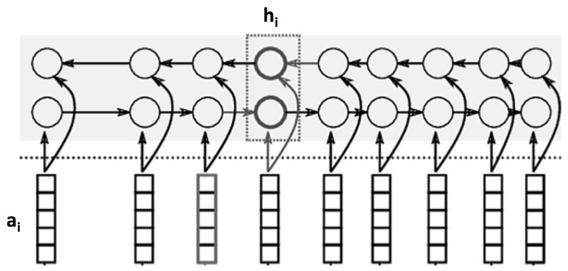

Figure 1: BiRNN representation for encoding a sentence

2.3.3 Features

In order to apply soft alignment in a machine learning setting, we wish to transform these alignments into a set of features. In particular, these features should be able to calculate a single value bounded in some range regardless of the relative size of the two target sentences. As we generate a non-square matrix of variable size, we hypothesize that good alignments should resemble hard alignments.

System	plagiarism	answer-answer	postediting	headlines	question-question	All
m5all3	0.80332	0.40165	0.81606	0.75400	0.72228	0.69528
m5dom1	0.75539	0.41211	0.80086	0.76778	0.69782	0.68368
m5dom2	0.74351	0.38303	0.76549	0.76485	0.57263	0.64520
Median	0.78949	0.48018	0.81241	0.76439	0.57140	0.68923
Best system on dataset	0.84138	0.69235	0.86690	0.82749	0.74705	0.77807

Table 2: Official results from SemEval, giving Pearson's correlation on each dataset

For hard alignments, we used the number of rows (i.e., tokens) that had at least one alignment, thus for the soft alignment we used the row max of the alignment as follows:

$$m_\phi = \frac{\sum_{i=1\ldots n_a} \frac{\max_{j=1\ldots n_b} |\alpha_{ij}|^\phi}{\sum_{j=1\ldots n_b} \alpha_{ij}^\phi}}{n_a}$$

For out experiments we calculated four features with the following values of $\phi = 0.1, 0.5, 1, 2$.

In addition, we experimented with other sparsity features proposed by Hurley and Rickard (2009) and included the H_G metric and a modified version of the $-l^p$ metric we call col-l^p as they gave good correlations with the sentence scores.

$$\begin{aligned} H_G = &\ -\textstyle\sum_{i=1,\ldots,n_a;j=1,\ldots,n_b} \log \alpha_{ij}^2 \\ -l^p = &\ (\textstyle\sum_{i=1,\ldots,n_a;j=1,\ldots,n_b} \alpha_{ij}^p)^{\frac{1}{p}} \\ \text{col-}l^p = &\ \frac{\sum_{i=1,\ldots,n_a}(\sum_{j=1,\ldots,n_b} \alpha_{ij}^p)^{\frac{1}{p}}}{n_a} \end{aligned}$$

We used $p = 2, 10$ to give two features for $\mathrm{col}{-}l^p$.

Finally, we noted that in many cases the most important alignments were those between low frequencies word, therefore we transformed our alignments by multiplying them with the inverse document frequency of the words, e.g.,

$$\alpha'_{ij} = \frac{\alpha_{ij}}{df(a_i)df(b_j)}$$

2.4 ESA Similarity

Gabrilovich and Markovitch (2007) introduced the ESA model that represents the semantics of a word with a distributional vector over the Wikipedia concepts. We use a snapshot of English Wikipedia from 1^{st} October, 2013 which contains 13.9 million articles (concepts). We built an index of all Wikipedia articles using Lucene. We retrieve distributional vector of a sentence by searching over a Lucene index. Thus, a Lucene ranking score represents the magnitude of a vector dimension that corresponds to a retrieved Wikipedia article. We used Lucene's built-in scoring function to obtain the top $K = 1000$ articles and then to obtain the semantic relatedness between two sentences, we compute cosine similarity between their distributional vectors.

2.5 Classifying with M5 Trees

Finally, having baseline features, extracted features from the hard and soft alignments, and the ESA similarity, we combine all of our features into a single vector and thus transform the problem into that of a traditional regression task. We experimented with various classifiers using the Weka toolkit (Hall et al., 2009) and found that in nearly all experiments, the strongest performance was obtained using the M5 Decision Tree method (Wang and Witten, 1996; Quinlan, 1992) and so we adopted this for all our experiments.

3 Evaluation

3.1 Internal Evaluation

We conducted a series of evaluations using data from previous SemEval challenges (Agirre et al., 2014) as a baseline as shown in Table 1. These results present the following configurations using 10-fold cross-validation:

Sultan Only (-DF) Using baseline features, which are also used in all experiments, and Sultan et al.'s (2014a) aligner. Without accounting for term document frequency (see Section 2.3.3)

Sultan Only As above only with document frequency included as a feature

Sultan + Jacana Including the Jacana aligner (Section 2.2)

Sultan + WordSim Including the WordSim features (Section 2.3.3)

Sultan + WordSim + WordPairs Including key-pairs features (Section 2.3.3)

All aligners Using Sultan, Jacana, WordSim and BiLSTM aligners

All aligners + ESA Also including the ESA method (Section 2.4)

In addition, we noticed that different datasets tended to have a different distribution of scores. As such we tried evaluating in two modes *macro-training* where we trained the decision tree on all datasets simultaneously and *microtraining* where a decision tree was trained for each dataset and used only for this dataset. As the microtraining results are much stronger, for the task, we developed a lightweight domain classifier that found the nearest training dataset to the test dataset and used the classifier trained on the most appropriate dataset in our evaluation runs. This classifier used the size of the intersection of the set of 100 most frequent words in each dataset and we obtained 100% accuracy for this classifier in identifying datasets by 10-fold cross-validation, i.e., we choose the classifier trained on the training set T which maximizes the following similarity to the test set T':

$$|\text{top100words}(T) \cap \text{top100words}(T')|$$

3.2 SemEval results

We submitted three runs to the SemEval task and the results are reported in Table 2, the configurations were as follows:

m5all3 This run uses all the aligners (Sultan, Jacana, WordSim) including ESA, and trains a single decision tree on all datasets simultaneously, without using the domain classifier.

m5dom1 This run uses Sultan, WordSim as aligners including ESA, and trains a decision tree per test dataset on the nearest training datasets using the domain classifier.

m5dom2 This run uses all the aligners including ESA, and trains a decision tree per test dataset on the nearest training dataset using the domain classifier.

We notice that our system has above-median performance in most datasets however, we have significant difficulties in the *answer-answer*, this is because of the length and complexity of the sentences. We also see that while the domain classifier was helpful in a few cases, notably the headlines, it did not in general seem to improve the results. Finally, we find that the combination of all classifiers was helpful.

4 Conclusion

We propose a combination of different approaches and our internal results suggest that combining multiple approaches can improve overall system performance. However, we notice that for some datasets performance is still very low and we hypothesise that in this case a new approach is needed. We notice that adapting to the domain proved extremely effective in the cross-fold setting but not in the general case, however our proposed method was very simplistic and further improvement in the quality of the result may be achieved by a more sophisticated algorithm.

Acknowledgments

This research was supported in by funding from the Science Foundation Ireland under Grant Number SFI/12/RC/2289 (Insight).

References

Nitish Aggarwal, Kartik Asooja, Georgeta Bordea, and Paul Buitelaar. 2015. Non-orthogonal explicit semantic analysis. *Lexical and Computational Semantics (* SEM 2015)*, page 92.

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.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. 2014. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 238–247, Baltimore, Maryland, June. Association for Computational Linguistics.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

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

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.

Christian Hänig, Robert Remus, and Xose De La Puente. 2015. ExB Themis: Extensive feature extraction from word alignments for semantic textual similarity. *SemEval-2015*, page 264.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Niall Hurley and Scott Rickard. 2009. Comparing measures of sparsity. *Information Theory, IEEE Transactions on*, 55(10):4723–4741.

John McCrae, Philipp Cimiano, and Roman Klinger. 2013. Orthonormal explicit topic analysis for cross-lingual document matching. In *Proceedings of the 2013 Conference on Empirical Natural Language Processing*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543.

John R Quinlan. 1992. Learning with continuous classes. In *5th Australian joint conference on artificial intelligence*, volume 92, pages 343–348. Singapore.

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.

Yong Wang and Ian H Witten. 1996. *Induction of model trees for predicting continuous classes*. Department of Computer Science, University of Waikato.

Michael J Wise. 1993. String similarity via greedy string tiling and running Karp-Rabin matching. *Online Preprint, Dec*, 119.

Xuchen Yao, Benjamin Van Durme, Chris Callison-Burch, and Peter Clark. 2013. A lightweight and high performance monolingual word aligner. In *ACL (2)*, pages 702–707. The Association for Computer Linguistics.