MayoNLP at SemEval-2016 Task 1: Semantic Textual Similarity based on Lexical Semantic Net and Deep Learning Semantic Model

Naveed Afzal *, Yanshan Wang *, Hongfang Liu
Department of Health Sciences Research
Mayo Clinic, Rochester, MN
{Afzal.Naveed, Wang.Yanshan, Liu.Hongfang}@mayo.edu

Abstract

Given two sentences, participating systems assign a semantic similarity score in the range of 0-5. We applied two different techniques for the task: one is based on lexical semantic net (corresponding to run 1) and the other is based on deep learning semantic model (corresponding to run 2). We also combined these two runs linearly (corresponding to run 3). Our results indicate that the two techniques perform comparably while the combination outperforms the individual ones on four out of five datasets, namely *answer-answer, headlines, plagiarism,* and *question-question*, and on the overall weighted mean of STS 2016 and 2015 datasets.

1 Introduction

There is a growing need for an effective method to compute semantic similarity between two text snippets. Many natural language processing (NLP) applications can benefit from effective semantic textual similarity (STS) techniques such as paraphrase recognition (Dolan et al., 2004), textual summarization (Aliguliyev, 2009), automatic machine translation evaluation (Kauchak and Barzilay, 2006), tweets search (Sriram et al., 2010), and student answer assessment (Rus and Lintean, 2012; Niraula et al., 2013).

The SemEval STS task series (Agirre et al., 2012; Agirre et al., 2013; Agirre et al., 2014; Agirre et al., 2015) have provided a vital platform for this task by making available a huge collection of sentence pairs with manual annotation for each sentence pair. The objective of the task is to compute semantic similarity between a pair of sentences in the range [0, 5], and where 0 indicates no similarity and 5 indicates complete similarity.

The approaches for computing semantic similarity between text snippets can be broadly classified into three approaches: information retrieval vector space model (e.g. Meadow, 1992; Sahami and Heilman, 2006), text alignment on the basis of semantic equivalence (e.g. Mihalcea et al., 2006) and machine learning models on the basis of lexical, syntactic and semantic features (e.g. Saric et al., 2012).

In this paper, we describe two techniques for the SemEval 2016 English STS task: the first one adopts the text alignment technique on the basis of semantic equivalence by leveraging a semantic net and corpus statistics; and the second technique is a machine learning models where we utilize a deep learning semantic model.

In the following, we present our techniques in detail and provide the evaluation results of our systems on STS 2016 and 2015 datasets. Conclusions and future directions are also provided.

2 System Description

This section describes the data and our systems in detail.

2.1 Data

There are five datasets in the SemEval 2016 English STS task: *answer-answer, headlines, plagiarism, postediting* and *question-question* collected form different sources in this year's English STS task. Each dataset consists of sentence pairs with human-assigned similarity score in the range of 0-5.

* indicates joint first author's

Proceedings of SemEval-2016, pages 674–679,
San Diego, California, June 16-17, 2016.

2.2 Preprocessing

In the preprocessing phase, for each sentence, we use regular expressions in Python[1] to find and replace all text contractions with their respective complete text. For example, "wouldn't" is replaced with "would not" All the URLs, email addresses, and parenthetical expressions (in the cases for abbreviation definition) are removed. Then we normalize common abbreviations with their expansions by using synonyms of those abbreviations in a lexical database, WordNet[2], (for example, "USA" is replaced with "United States of America") and a manually crafted abbreviation list based on STS 2015 data (for example, "govt" is replaced with "government", "1.6lb" and "5m" are replaced with "1.6 pound" and "5 million"). Moreover, we replace all the negations such as 'not present' with 'absent' using WordNet antonyms, and correct misspelled words using an English dictionary provided by a dictionary module pyenchant[3] in Python. Stopwords are removed for datasets *headlines, plagiarism* and *postediting,* according to the Stanford stopwords list. For the datasets of *answer-answer* and *question-question*, stopwords are not removed since many pairs only contain stopwords, such as sentence pair "*Can you do this?" "You can do it, too", etc.* Finally, the remaining words in each sentence are lemmatized to their base forms using the lemmatizer provided by natural language toolkit (NLTK) in Python. The entire preprocessing workflow is illustrated in Figure 1.

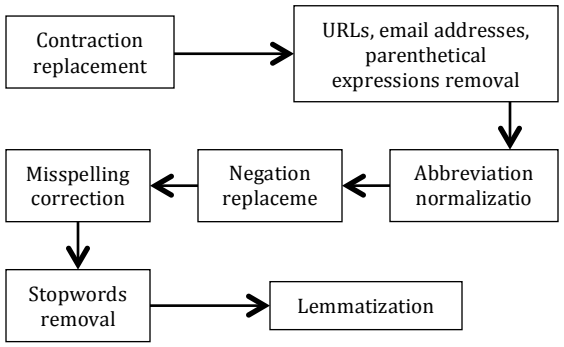

Figure 1: Workflow of preprocessing

[1] https://www.python.org/
[2] https://wordnet.princeton.edu/
[3] https://pypi.python.org/pypi/pyenchant/

2.3 Run 1

Our first technique (official run 1), illustrated in Figure 2, is inspired by the work of Li et al. (Li et al., 2006) where the textual similarity computation is based on both syntactic and semantic similarities.

The main differences of our technique comparing to theirs include preprocessing and synsets selection. We select the most similar pair of synsets from WordNet instead of just picking the first noun synset for each word.

After the preprocessing phase, the textual similarity is derived from the combination of semantic (semantic vectors) and syntactic information (syntactic vectors) present in a sentence pair as detailed in the following.

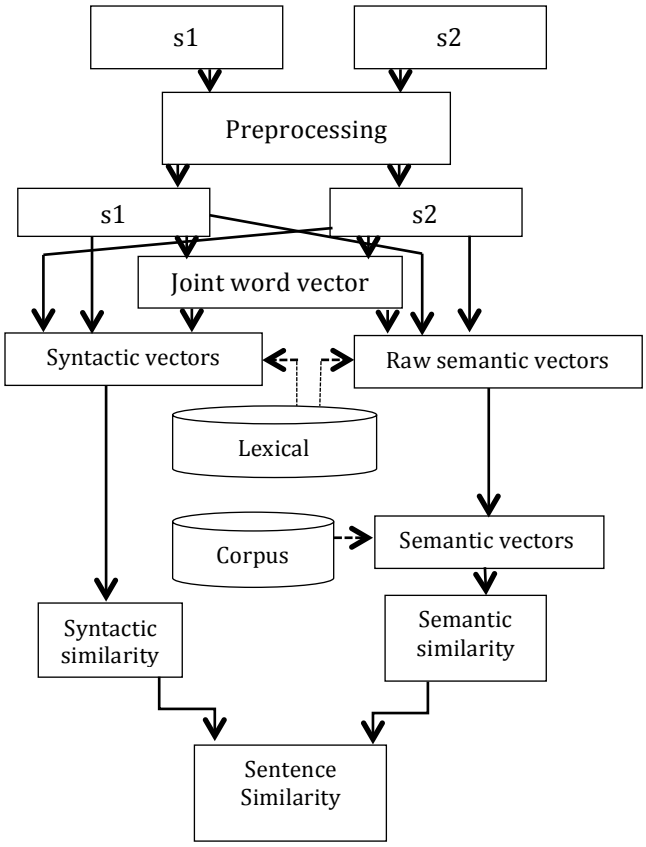

Figure 2: System architecture (Run 1)

We first form a joint word vector (JWV) by collecting unique words that occur in the sentence pair. The semantic similarity is computed by comparing the semantic meaning hidden in the

sentences and the syntactic similarity is computed according to the word order in the two sentences.

To compute semantic similarity of the pair, each sentence is mapped to a vector: if *i*th word in JWV is present in the sentence, we assign a value 1 to the *i*th entry; otherwise, we calculate the semantic similarity score between the *i*th word and each word in the sentence and choose the highest score to be the *i*th entry with the aid of the WordNet using the following strategy.

For every two words in the WordNet, we use the path length and depth of two words in the WordNet to calculate *i*th entry:

$$s\left(w_i, w_j\right) = e^{\alpha l} \cdot \frac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}},$$

where l is the length of the shortest path between words w_i and w_j, h returns a measure of depth in the WordNet, and α, β are constants (0.2 and 0.45 respectively in our experiment). α and β are dependent on the knowledge base and for WordNet we have used the values for alpha and beta found to be best by Li et al., (2003). The reason behind using depth along with the length of the shortest path is that in the WordNet, words at upper layers are more generic comparing to words at lower layers. If $s\left(w_i, w_j\right)$ is greater than a threshold (0.2 in our experiment) then we set the value to be the score, otherwise we set the value to 0.

The similarity score of the *i*th entry is then normalized by multiplying information content score of the *i*th word. The information content of word is defined by (Resnik, 1995):

$$I(w) = 1 - \frac{\log(n+1)}{\log\,(N+1)},$$

where N is the total number of words in the Brown corpus, n is the frequency of the word w in the corpus. According to this definition, the information content score is between 0 and 1. In our system, the information content of each word was computed using the Brown corpus (Francis and Kučera, 1979).

The semantic similarity is then defined as S_{sem}, which is the cosine similarity between the two vectors corresponding to the sentence pair.

Similarly, to compute syntactic similarity, each sentence is mapped to a syntactic vector. The dimension of the syntactic vector is identical as the size of the JWV. If the *i*th word in the JWV occurs at the *j*th position in the sentence, then the value of the *i*th entry in the vector is *j*. If the *i*th word does not exist in the sentence, the value of the *i*th entry is the position of the most similar word obtained from WordNet using $s\,(w1, w2)$. Different from the form of semantic similarity measure, syntactic similarity should take word order information into consideration. Therefore, the syntactic similarity is defined by:

$$s_{syn} = 1 - \frac{|o_1 - o_2|}{|o_1 + o_2|},$$

where o_1 and o_2 represent syntactic vector for sentences s1 and s2 respectively. From the definition of syntactic vector, we see that it contains the basic structural information of a sentence and measure the word order similarity between two sentences.

Since semantic similarity represents the lexical similarity while syntactic similarity contains information about the orders between words, both contribute in conveying the similarity of the sentences. Therefore, the final sentence pair similarity is defined by combining semantic similarity and syntactic similarity, i.e.

$$S\,(s_1\,, s_2) = \Omega\, s_{sem} + (1 - \Omega\,) s_{syn}$$

Since syntax plays a subordinate role for semantic processing of text we used Ω ($\Omega = 0.85$) in the final similarity.

2.4 Run 2

Our second system (run 2) is built upon a Deep Structured Semantic Model (DSSM) proposed by Huang et al., (2013). DSSM is a deep learning based technique that is proposed for semantic understanding of textual data. It maps short textual strings, such as sentences, to feature vectors in a low-dimensional semantic space. Then the vector representations are utilized for document retrieval by comparing the similarity between documents and queries. DSSM is reported to outperform other semantic models applying to document retrieval (Huang et al., 2013). However, the performance has not been evaluated to measure the degree of

similarity in the underlying semantics of paired snippets of text.

DSSM uses the typical deep neural network (DNN) architecture to represent a sentence (document) in the semantic vector space. Unlike other DNN where bag-of-word (BoW) term vectors are used as input, DSSM employ a novel word hashing method to reduce the dimensionality of the BoW vectors. Specifically, each word (for example, 'girl') is attached with a word starting mark and an ending mark (i.e., '#girl#') and split into letter trigrams (i.e., #gi, gir, irl, rl#). By doing so, the dimension is reduced to the number of trigrams, which is 3073 in our system. Then, each word representing by a vector of letter trigrams is used as input to the DNN.

The layers in DNN consists of three parts, namely, word hashing layer, hidden layers, and top layer, where the layer functions are:

$$l_1 = W_1 x,$$
$$l_i = f(W_i l_{i-1} + b_i), i = 2, \dots, N-1,$$

and

$$y = f(W_N l_{N-1} + b_N),$$

where x is the input term vector, y the output vector, $l_i, i = 1, \dots, N-1$ the hidden layers, W_i the *i*th weight matrix, b_i the *i*th bias, and $f(\cdot)$ the *tanh* activation function. The feature vectors generated by the word-hashing layer are projected through the hidden layers, and formed as semantic feature vectors in the top layer.

After obtaining the semantic feature vectors for each paired snippets of text, cosine similarity is utilized to measure the semantic similarity between the pair. Since the elements of semantic vector are nonnegative and the range of cosine similarity is [0, 1], we used the simplest linear transformation to map the range into [0, 5].

Regarding the implementation of DSSM, we exploit the Sent2Vec tool. Sent2Vec provides DSSM and convolutional DSSM (C-DSSM) (Gao et al., 2014; Shen et al., 2014). Based on our experiments on the SemEval 2015 English STS datasets, the performance of DSSM is slightly superior to C-DSSM without statistical significance. Therefore, DSSM is utilized in the official run 2. In the DSSM, we used one hidden layer with 1000 hidden nodes in the neural networks. A larger number of hidden layers or hidden nodes may be applied though we used the simplest one due to the computational complexity.

2.5 Run 3

Our run 3 is a linear combination of run 1 and run 2, i.e.,

$$S_{run3} = \alpha \cdot S_{run1} + \beta \cdot S_{run2}$$

where the equal weights are assigned to run 1 and run 2, i.e., $\alpha = \beta = 0.5$

3 Evaluation

We submitted 3 runs at SemEval 2016 English STS task based on the Section 2.

Dataset	Run 1	Run 2	Run 3	Best
answer-answer (254 pairs)	0.58873	0.57739	0.61426	0.69235
headlines (249 pairs)	0.73458	0.75061	0.77263	0.82749
plagiarism (230 pairs)	0.76887	0.80068	0.80500	0.84138
postediting (244 pairs)	0.85020	0.82857	0.84840	0.86690
question-question (209 pairs)	0.69306	0.73035	0.74705	0.74705
Weighted Mean	0.72646	0.73569	0.75607	0.77807

Table 1: Official Results of evaluation on STS 2016 data

Table 1 shows the performance of each official run. This is reported as the Pearson correlation between the scores produced by our systems and the human annotations. The last row shows the value of weighted mean that is the sum of all datasets correlation scores. The weight assigned to a dataset is proportional to its number of pairs.

Our run 1 performed better than run 2 on the following datasets: *answer-answ*er and *postediting* while run 2 performed better than run 1 on the following datasets: *headlines, plagiarism* and *question-question*. Our run 3 that is a linear combination of run 1 and run 2 is the best performing system in terms of weighted mean.

3.1 Performance on STS 2015 Data

We also applied our systems on the SemEval 2015 English STS task data. Table 2 shows the evaluation results on STS 2015 data. We can observe that mean run 3 outperforms run 1 and run 2 in terms of overall weighted mean. However, for *answers-students* dataset the performance of run 1 is better than other runs, and for datasets *belief* and *images* run 2 perform better than other two runs. The reason might be that run 1 considers word order information, which is crucial information when describing math or chemistry problems in dataset *answers-students*.

Dataset	Run 1	Run 2	Run 3	Best
answers-forums (375 pairs)	0.7049	0.6660	0.7246	0.7390
answers-students (750 pairs)	0.7422	0.6770	0.7416	0.7879
belief (375 pairs)	0.7287	0.7439	0.7389	0.7717
headlines (750 pairs)	0.7630	0.7749	0.7776	0.8417
images (750 pairs)	0.8198	0.8576	0.8367	0.8713
Weighted Mean	0.7604	0.7536	0.7719	0.8015

Table 2: Results of evaluation on STS 2015 data

4 Conclusions and Future Work

In this paper, we presented three systems for measuring the semantic similarity between two sentences. The systems will be released in the future at GitHub.[1] Our systems give promising results on both English STS datasets 2015 and 2016. Our first system relies on a lexical database and corpus statistics. A lexical database represents common human knowledge about a natural language while a corpus reflects the actual usage of language and words. Our second system utilizes deep learning semantic model while our third system is an ensemble of system 1 and 2.

In our future work, we would like to investigate the different contributions of syntactic similarity and semantic similarity in run 1 and improve the performance of run 1 by adding new corpora to increase the impact of corpus statistics. For the second system, we would like to test whether adding additional hidden layers into the deep neural net improves the performance. Currently, our third system is a linear combination of system 1 and 2 by using heuristic weights since no training data were given in the task and we concerned about over-fitting if the data from previous years were used to train our system. In the future we would like to investigate whether the performance could be improved by applying supervised or unsupervised machine learning algorithms to calculate the optimal weights based on the previous data sets. Moreover, we would like to extend our systems into the biomedical domain with the use of biomedical lexical databases and biomedical corpus.

[1]https://github.com/naveedafzal/MayoClinic_SemEval_2016_Task-1

Acknowledgments

Our participation of STS is partially supported by the grants received from NIGMS R01 GM102282 and from National Library of Medicine R01LM11934.

References

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.

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.

C.T. Meadow. 1992. Text Information Retrieval Systems. Academic Press, Inc.

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

Eneko Agirre, Daniel Cer, Mona Diab, 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 2: Proceedings of the Sixth International Workshop on Semantic Evaluation, SemEval '12, pages 385-393, Montreal, Canada.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *SEM 2013 Shared Task: Semantic Textual Similarity. In Proceedings of the Second Joint Conference on Lexical and Computational Semantics, *SEM '13, pages 32-43, Atlanta, Georgia, USA.

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 '14, pages 81-91, Dublin, Ireland.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre,Weiwei Guo, I˜nigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, German Rigau, Larraitz Uria, and Janyce Wiebe. 2015. SemEval-2015 Task 2: Semantic Textual Similarity, English, Spanish and Pilot on Interpretability. In Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval '15, Denver, Colorado, USA.

Frane Saric, G. Glavas, M. Karan, J. Snajder, B.D. Basic. 2012. TakeLab: systems for measuring semantic text similarity. In Proceedings of 1st Joint Conference on Lexical and Computational Semantics, pp. 441–448.

Jianfeng Gao, Patrick Pantel, Michael Gamon, Xiaodong He, Li Deng, Yelong Shen. 2014. Modeling interestingness with deep neural networks. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Doha, Qatar.

M Sahami, T.D Heilman. 2006. A web-based kernel function for measuring the similarity of short text snippets. In Proceedings of 15th International Conference on World Wide Web, pp. 377–386.

Nobal B. Niraula, Rajendra Banjade, Dan Stefanescu, and Vasile Rus. 2013. Experiments with semantic similarity measures based on lda and lsa. In Statistical Language and Speech Processing , pages 188–199. Springer.

Philip Resnik. 1995. Using information content to evaluate semantic similarity in a taxonomy. In Proceedings of the 14th international joint conference on Artificial intelligence, pages 448-453, Montreal, Quebec, Canada.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng. 2013. Learning deep structured semantic models for web search using clickthrough data. In Proceedings of the 22nd ACM international conference on Conference on information & knowledge management, pages. 2333-2338, San Francisco, CA, USA.

Rada Mihalcea, Courtney Corley, Carlo Strapparava. 2006. Corpus-based and knowledge-based measures of text semantic similarity. In Proceedings of 21st National Conference on Artificial Intelligence, pp. 775–780.

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.

Vasile Rus and Mihai Lintean. 2012. A comparison of greedy and optimal assessment of natural language student input using word-to-word similarity metrics. In Proceedings of the Seventh Workshop on Building Educational Applications Using NLP, pages 157–162.

W. N. Francis and H. Kučera. 1964, 1971, 1979. A standard corpus of present-day edited American English, for use with digital computers (Brown). Compiled by Brown University. Providence, Rhode Island.

Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, Gregoire Mesnil. 2014. A latent semantic model with convolutional-pooling structure for information retrieval. In Proceedings of the 23rd ACM International Conference on Information and Knowledge Management, Shanghai, China.

Yuhua. Li, Zuhair. A. Bandar, David McLean. 2003. An approach for measuring semantic similarity using multiple information sources. IEEE Trans. Knowledge and Data Engineering, vol. 15, no.4, pages 871-882.

Yuhua. Li, David McLean, Zuhair. A. Bandar, James D. O'Shea, and Keeley. Crockett. 2006. Sentence similarity based on semantic nets and corpus statistics. IEEE Trans. Knowledge and Data Engineering, vol. 18, no. 8, pages 1138–1150.