NaCTeM at SemEval-2016 Task 1: Inferring sentence-level semantic similarity from an ensemble of complementary lexical and sentence-level features

Piotr Przybyła, Nhung T. H. Nguyen, Matthew Shardlow,
Georgios Kontonatsios and **Sophia Ananiadou**
National Centre for Text Mining
University of Manchester
Manchester, UK
{piotr.przybyla,nhung.nguyen,matthew.shardlow,
georgios.kontonatsios,sophia.ananiadou}@manchester.ac.uk

Abstract

We present a description of the system submitted to the Semantic Textual Similarity (STS) shared task at SemEval 2016. The task is to assess the degree to which two sentences carry the same meaning. We have designed two different methods to automatically compute a similarity score between sentences. The first method combines a variety of semantic similarity measures as features in a machine learning model. In our second approach, we employ training data from the Interpretable Similarity subtask to create a combined word-similarity measure and assess the importance of both aligned and unaligned words. Finally, we combine the two methods into a single hybrid model. Our best-performing run attains a score of 0.7732 on the 2015 STS evaluation data and 0.7488 on the 2016 STS evaluation data.

1 Introduction

If you ask a computer if two sentences are identical, it will return an accurate decision in a split-second. Ask it to do the same for a million sentence pairs and it will take a few seconds — far quicker than any human. But similarity has many dimensions. Ask a computer if two sentences mean the same and it will stall, yet the human can answer instantly. Now we have the edge. But what metrics do we use in our personal cognitive similarity-scoring systems? The answer is at the heart of the semantic similarity task. In our solution, we have incorporated several categories of features from a variety of sources. Our approach covers both low-level visual features such as length and edit-distance as well as high-level semantic features such as topic-models and alignment quality measures.

Throughout our approach, we have found it easier to consider similarity at the lexical level. This is unsurprising, as each lexeme may be seen as a semantic unit with the sentence's meaning built directly from a specific combination of lexemes. We have explored several methods for abstracting our lexical-level features to the sentence level as explained in Section 4. Our methods are built from both intuitive features as well as the results of error-analyses on our trial runs. We have combined multiple feature sources to build a powerful semantic-similarity classifier, discarding non-performing features as necessary.

2 Related Work

The presented system has been submitted to the Semantic Textual Similarity (STS) shared task at SemEval 2016, which has been organised since 2012 (Agirre et al., 2016). Existing approaches to the problem adopt a plethora of similarity measures including string-based, content-based and knowledge-based methods. String-based methods (Bär et al., 2012; Malakasiotis and Androutsopoulos, 2007; Jimenez et al., 2012) exploit surface features (e.g., character n-grams, lemmas) to compute a semantic similarity score between two sentences. Bär et al. (2012) showed that string-based features improve performance when using machine learning. Knowledge-based features (Mihalcea et al., 2006; Gabrilovich and Markovitch, 2007) estimate the semantic similarity of textual units using external

Proceedings of SemEval-2016, pages 614–620,
San Diego, California, June 16-17, 2016.

knowledge resources (e.g., WordNet). As an example, Liu et al. (2015) used the shortest path that links two words in the WordNet taxonomy as a similarity measure between words. To calculate a similarity score between sentences, the authors used the sum of the similarity scores of the constituent words. Content-based features are based upon the distributional similarity of words and sentences. Distributional semantics methods (Mikolov et al., 2013; Baroni et al., 2014) encode the lexical context of words into a vector representation. A vector representation of a sentence may be estimated as a function of the vectors of the constituent words (Mitchell and Lapata, 2010). The semantic relatedness between sentences is measured using the cosine of the angle of the composed sentence vectors.

3 Feature Sources

We have collected a variety of resources to help us create semantic similarity features. Some of these (Subsections 3.2 and 3.4) give features at the level of the sentence itself. Other resources (Subsections 3.1, 3.3, 3.5. 3.6 and 3.7) give features at the level of the individual words. We explain how we adapt these features to sentence-level metrics in Section 4.

3.1 Distributional semantics

We use a count-based distributional semantics model (Turney and Pantel, 2010) and the Continuous Bag-Of-Words (CBOW) model (Mikolov et al., 2013) to learn word vectors. The training corpus that we used is a combination of all monolingual texts provided by the organiser of the 2014 Machine Translation Shared Task[1], whose size is about 20 GB. Before training, we tokenised and transferred the corpus into lowercase text. The size of the context window is 5 for both of the models. The numbers of dimensions in the resulting vectors are 150,000 and 300 for the count-based and the CBOW models respectively.

3.2 Machine translation

A pair of input sentences can be considered as the input and output of a machine translation system. Therefore, we can apply machine translation (MT) metrics to estimate the semantic relatedness of the input pair. Specifically, we used three popular MT metrics: BLEU (Papineni et al., 2002), Translation Edit Rate (TER) (Snover et al., 2006), and METEOR (Denkowski and Lavie, 2014).

3.3 Lexical paraphrase scores

Another promising resource for similarity estimation is the lexical paraphrase database (PPDB) by Ganitkevitch et al. (2013). Each of the word pairs in the PPDB has a set of 31 different features (Ganitkevitch and Callison-Burch, 2014). In this work, we calculate the similarity of a word pair by using the formula that Ganitkevitch and Callison-Burch (2014) recommended to measure paraphrases' quality. However, for word pairs that have been seen very rarely, i.e., their rarity penalty score in PPDB is higher than 0.1, we simply set the similarity score at 0 instead of applying the formula.

3.4 Topic modelling

We induce a topic-based vector representation of sentences by applying the Latent Dirichlet Allocation (LDA) method (Blei et al., 2003). We hypothesise that a varying granularity of vector-representations provide complementary information to the machine learning system. Based upon this, we extract 26 different topic-based vector representations by varying the number of topics; starting from a small number of 5 topics which resulted in a coarse-grained topic-based representation to a larger number of 800 topics which produced a more fine-grained representation. In our experiments, we used the freely available MALLET toolkit (McCallum, 2002). Additionally, we performed hyper-parameter optimisation for every 10 Gibbs sampling iterations and set the total number of iterations to $2,000$.

3.5 WordNet

For WordNet-based similarity between a pair of words we have chosen Jiang-Conrath (Jiang and Conrath, 1997) similarity based on an evaluation by Budanitsky and Hirst (2006). To compute the score, we lemmatise the words using Stanford CoreNLP (Manning et al., 2014), find corresponding synsets in Princeton WordNet (Fellbaum, 1998) and obtain the Jiang-Conrath value using the WS4J library[2].

[1] http://www.statmt.org/wmt14/translation-task.html#download

[2] https://code.google.com/archive/p/ws4j/

3.6 Character string

Sometimes semantically related words are very similar as sequences of characters, e.g. *cooperate* and *co-operate* or *recover* and *recovery*. To handle such cases we compute Levenshtein distance (Levenshtein, 1966) between words. To keep the similarity score x_l in the $[0, 1]$ range, we adjust the obtained distance d by computing $x_l = (l - d)/l$, where l denotes the length of the longer word.

3.7 Word importance measures

We calculated the combined probability of each word occurring in a given context. We used smoothed unigram and bigram probabilities taken from the Google Web1T data (Brants and Franz, 2006). We multiplied the smoothed unigram probability of a word together with the smoothed bigram probability of both the word itself and the word immediately before it to give the 2-word contextual probability of a word's appearance. Whilst this model could theoretically be extended to longer sequences, we found that memory resources limited our capacity to a sequence of 2 words.

$$p(w_i|w_{i-1}, w_i) = p(w_{i-1}, w_i) \times p(w_i)$$

We also investigated psycholinguistic properties as measures of word importance. We used the MRC Psycholinguistic norms (Wilson, 1988) to attain values for the 'familiarity', 'imagery' and 'concreteness' of each word. These metrics can be defined as follows:

Familiarity: This indicates how likely a word is to be recognised by a user. Words which occur very often such as *cat* are likely to be more familiar than less common words such as *feline*.

Concreteness: This indicates whether a reader perceives a word as referring to a physical entity. A conceptual phrase such as *truth* will have a lower value for concreteness than an object which is more easily relatable such as *house* or *man*.

Imagery: This metric indicates how easy it is to call-up images associated with a word. This is related to concreteness, but may differ in some cases. For example, some actions (*jumping*, *flying*) or common emotions (*joy*, *sadness*, *fear*) may have high imagery, but low concreteness.

4 Feature Generation

We found that the greatest challenge of the task was to combine different existing word-level relatedness cues into a sentence-level similarity measure. We have submitted three permutations of our system, as allowed by the task. The first system 'Aggregation (*Macro*)' passes a set of 36 features through a random forest classifier. The second system 'Alignment (*Micro*)' first aligns the sentences using word-level metrics and then calculates 49 features describing this alignment. Our final system '*Hybrid*' is simply the combination of the former two approaches.

4.1 Aggregation (Macro)

In this approach feature sources are used separately, each applied as a sentence similarity measure that is further represented as a single feature.

- *Compositional semantic vectors.* A sentence vector is simply calculated by cumulatively adding its component word vectors. The similarity of a sentence pair is then estimated by the cosine of the angle between the corresponding vectors.

- *Average maximum cosine.* Given a sentence pair of (s, t) whose number of words are m and n respectively, the similarity score is calculated as the average of the maximum cosine similarity of word pairs as follows:

$$score(s, t) = \frac{\sum_{i=1}^{m} \max_{j=1}^{n} \cos(\overrightarrow{w_i}, \overrightarrow{w_j})}{\max(m, n)}$$

- *MT metrics.* We apply MT metrics at the sentence level. We used smoothed BLEU-4 and TER implemented within Stanford Phrasal[3] while the METEOR score was provided using techniques from Denkowski and Lavie (2014).

- *Paraphrase DB.* To compute sentence similarity using PPDB, we find the most similar counterpart for each of the words and return an average of obtained similarity scores, weighted by word length.

[3] http://nlp.stanford.edu/phrasal/

- *Topic modelling metric.* Given that each sentence is represented by a topic-based vector, we can compute a similarity score for a sentence pair using the cosine similarity.

This leads to 36 features, each expected to be positively correlated with sentence similarity.

4.2 Alignment (Micro)

In this approach we combine different techniques of assessing relatedness to create a single word similarity measure, use it to align compared sentences, and compute features describing a quality of alignment.

4.2.1 Word-level similarity

We employ a machine learning model to create a measure of semantic similarity between words. The model uses four measures:

1. adjusted Levenshtein distance x_l,
2. word-vector cosine similarity x_{wv},
3. WordNet Jiang-Conrath similarity x_{wn},
4. paraphrase database score x_p.

Each measure returns values between 0 (no similarity) and 1 (identical words). As a training set, we have used all the data from the previous year's interpretable subtask (Agirre et al., 2015), which includes pairs of chunks with assigned similarity between 0 and 5. As this set contain pairs of chunks, not words, we extended these measures for multi-word arguments. This has been done by (1) using the whole chunk for Levenshtein distance, (2) finding the best pair for WordNet similarity and (3) using solutions for sentence-level aggregation (described in the previous section) for word vectors and paraphrase database.

Negative examples have been created by selecting unaligned pairs and assigning to them a score equal to 0. In that way we obtain 19,673 training cases, from which the following linear regression model has been deduced:

$$y = -0.3285 + 1.3343 \times x_l + 0.8625 \times x_{wv} \\ +0.9875 \times x_{wn} + 2.1496 \times x_p$$

The coefficients of this model show that the word vectors and WordNet features have a lower influence on the final score output by the model, whereas the paraphrase database score has a greater influence. Although the final model was trained on all the data from last year's interpretable similarity subtask, we performed a separate evaluation on this data in which we partitioned the data into train and test subsets. The resulting correlation on the test subset was 0.8964, which we consider to be very reasonable.

4.2.2 Finding alignment

Having a universal similarity measure between words, we can compute an alignment of sentences. To do this, we tokenise each sentence and compute similarity value between every pair of words. Then we find an alignment in a greedy way, by pairing free words in order of decreasing similarity of pairs. This process stops when we reach 1 (in a 0-5 scale, see previous section), which usually leaves some of the words unaligned.

4.2.3 Features

The features generated in this approach describe the quality of the alignment of a pair of sentences. The simple measures are: mean similarity between aligned pairs, length of aligned parts as a proportion of sentence length (average from two sentences) and a number of crossings in permutation defined by the alignment, i.e. Kendall's tau (Kendall, 1955). Secondly, we also include sums of lengths of aligned and unaligned words with particular tags, using Stanford CoreNLP (Manning et al., 2014) with slightly simplified Brown tagset (19 tags). This follows our intuition that significance of success or failure of alignment of a particular word is different for different parts of speech (e.g proper nouns vs. determiners). Finally, we measure the importance of matched and unmatched words by their probability (sum of logarithms) and psycholinguistic metrics – familiarity, concreteness and imagery (sum of values). As a result of this process we get 49 features.

4.3 Hybrid

The hybrid approach simply combines outputs from the two feature generation solutions described above. This leads to 85 features, some of which may be redundant.

Feature	Impurity decrease
Paraphrase DB score	8082.91
METEOR	4611.94
Average max. cosine	2262.56
Compositional sem. cosine	1895.34
LDA (800 topics)	967.74
LDA (675 topics)	907.32
LDA (600 topics)	852.86
Smoothed BLEU	753.78
LDA (525 topics)	735.61
Translation Edit Rate	706.48

Table 1: Ten most useful variables from the aggregation approach according to random forest importance measure, i.e. mean decrease in residual sum of squares.

Feature	Impurity decrease
Alignment ratio	9350.65
Prob. of aligned words	4809.45
Mean similarity	2361.40
Unaligned nouns	1691.97
Prob. of unaligned words	1576.07
Aligned nouns	1258.54
Fam. of aligned words	1224.19
Im. of aligned words	1073.80
Fam. of unaligned words	905.14
Con. of unaligned words	874.51

Table 2: Ten most useful variables from the alignment approach according to random forest importance measure, i.e. mean decrease in residual sum of squares (*Prob.* – probability, *Fam.* – familiarity, *Im.* – imagery, *Con.* – concreteness).

5 Learning

Each of the three feature sets described above has been used to create a single regression model. For this purpose we explored different methods available in the R environment (R Core Team, 2013): linear regression, decision trees (Breiman et al., 1984), multivariate adaptive regression splines (Friedman, 1991) and generalised additive models (Hastie and Tibshirani, 1990), but internal evaluation has shown that random forests (Breiman, 2001) perform the best in this task. Since the dimensionality of the task is not very high, we have not performed any feature selection, leaving this to the random forests. They offer a useful measure of variable importance, namely the decrease in residual sum of squares averaged over all trees. Tables 1 and 2 show the ten most useful variables for this purpose from the aggregation and alignment approaches, respectively.

6 Evaluation

As explained in previous sections, we have used three features sets to create three classification models, called *Macro* (from aggregation-based features), *Micro* (from alignment-based features) and *Hybrid* (including both feature sets). According to the shared task guidelines, the performance has been measured by computing a correlation between predicted and gold standard (assigned by humans) scores in each data set, and then obtaining their average.

We have performed two main experiments. Firstly, we have used the data that have been available for training and testing in the 2015 competition (Agirre et al., 2015) to select the best approach. Then, we have created three final models on all available 2015 data and used them to label the test data provided by the organisers of the 2016 task. Table 3 shows the results of both experiments.

7 Discussion

What may seem the most surprising in the results is their diversity – some of the methods achieve correlation well over 0.8 for one data set and below 0.6 for another. In our opinion, that not only shows that there is room for improvement but also reveals a fundamental problem in this task, namely that training and test sets come from different sources. The distribution of features also differs. This situation simulates many real-world scenarios, but also impedes any ML-based approach. In this light, our drop in performance between development (2015) and evaluation (2016) sets seems satisfactorily small.

The data sets related to questions and answers have turned out to cause the most difficulties for our system. We have performed a post-hoc error analysis to understand why. We found that these sets expose weaknesses of our reliance on abstracting from word-level similarity to the sentence-level. For example, consider the following two questions: *What is the difference between Erebor and Moria?* and *What is the difference between splicing and superim-*

	Features set		
Data set	**Macro**	**Micro**	**Hybrid**
2015			
images	0.8022	0.8129	0.8457
headlines	0.7963	0.8183	0.8263
belief	0.7107	0.7031	0.7455
answers-students	0.6994	0.7264	0.7378
answers-forum	0.6186	0.6797	0.7108
Mean	0.7254	0.7481	**0.7732**
2016			
headlines	0.7976	0.7914	0.8046
plagiarism	0.7895	0.8313	0.8148
postediting	0.8261	0.8266	0.8286
answer-answer	0.5848	0.5521	0.6024
question-question	0.6704	0.6124	0.6937
Mean	0.7337	0.7228	**0.7488**

Table 3: Correlations of predicted and actual similarity scores measured for datasets available in 2015 and 2016 shared task.

position? As you can see, they have the same structure and a lot of identical words, but the two remaining words create a wholly different meaning. This means that our approach, confirmed by the ranking of features (see table 2), deserves further work.

We may be able to boost our performance on the 2016 task by a few simple measures. Firstly, we could include semantic features from a wider variety of sources. Especially for the word-importance metrics. Secondly, we could consider pre-filtering the sentences for a list of stop-words which typically do not contain much semantic importance to a sentence. Finally, we could attempt to weight our word-level measures based on the semantic importance of each word in a sentence. For example, verbs and nouns are probably more important than adjectives and adverbs, which in turn are likely to be more important than conjunctions.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Inigo 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 2015)*. Association for Computational Linguistics.

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, and Janyce Wiebe. 2016. SemEval-2016 Task 1: Semantic Textual Similarity - Monolingual and Cross-lingual Evaluation. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval 2016)*. Association for Computational Linguistics.

Daniel Bär, Chris Biemann, Iryna Gurevych, and Torsten Zesch. 2012. UKP: Computing semantic textual similarity by combining multiple content similarity measures. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics (SemEval '12)*. Association for Computational Linguistics.

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*. Association for Computational Linguistics.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022.

Thorsten Brants and Alex Franz. 2006. Web 1T 5-gram corpus version 1.1. *Linguistic Data Consortium*.

Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. 1984. *Classification and Regression Trees*. Wadsworth & Brooks.

Leo Breiman. 2001. Random Forests. *Machine Learning*, 45(1):5–32.

Alexander Budanitsky and Graeme Hirst. 2006. Evaluating WordNet-based Measures of Lexical Semantic Relatedness. *Computational Linguistics*, 32(1):13–47.

Michael Denkowski and Alon Lavie. 2014. Meteor Universal: Language Specific Translation Evaluation for Any Target Language. In *Proceedings of the EACL 2014 Workshop on Statistical Machine Translation*. Association for Computational Linguistics.

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

Jerome H. Friedman. 1991. Multivariate Adaptive Regression Splines. *The Annals of Statistics*, 19:1–67.

Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence (IJCAI'07)*. Morgan Kaufmann Publishers.

Juri Ganitkevitch and Chris Callison-Burch. 2014. The Multilingual Paraphrase Database. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*. European Language Resources Association.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The Paraphrase Database. In *The 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT '13)*. Association for Computational Linguistics.

Trevor J. Hastie and Robert J. Tibshirani. 1990. *Generalized Additive Models*. Chapman & Hall/CRC.

Jay J. Jiang and David W. Conrath. 1997. Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy. In *Proceedings of 10th International Conference on Research in Computational Linguistics (ROCLING'97)*.

Sergio Jimenez, Claudia Becerra, and Alexander Gelbukh. 2012. Soft cardinality: A parameterized similarity function for text comparison. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics (SemEval '12)*. Association for Computational Linguistics.

Maurice G. Kendall. 1955. *Rank correlation methods*. Hafner Publishing.

Vladimir Iosifovich Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10:707–710.

Yang Liu, Chengjie Sun, Lei Lin, and Xiaolong Wang. 2015. yiGou: A Semantic Text Similarity Computing System Based on SVM. *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*.

Prodromos Malakasiotis and Ion Androutsopoulos. 2007. Learning textual entailment using SVMs and string similarity measures. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*. Association for Computational Linguistics.

Christopher D. Manning, John Bauer, Jenny Finkel, Steven J. Bethard, Mihai Surdeanu, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Association for Computational Linguistics.

Andrew Kachites McCallum. 2002. MALLET: A machine learning for language toolkit.

Rada Mihalcea, Courtney Corley, and Carlo Strapparava. 2006. Corpus-based and knowledge-based measures of text semantic similarity. In *Proceedings of the 21st National Conference on Artificial intelligence (AAAI '06)*. AAAI Press.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *arXiv*.

Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1429.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL '02)*. Association for Computational Linguistics.

R Core Team. 2013. R: A Language and Environment for Statistical Computing.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas*, pages 223–231.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: vector space models of semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188.

Michael Wilson. 1988. MRC Psycholinguistic Database: Machine-usable dictionary , version 2.00.