# KLUE-CORE: A regression model of semantic textual similarity

**Paul Greiner** and **Thomas Proisl** and **Stefan Evert** and **Besim Kabashi**
Friedrich-Alexander-Universität Erlangen-Nürnberg
Department Germanistik und Komparatistik
Professur für Korpuslinguistik
Bismarckstr. 6
91054 Erlangen, Germany
{paul.greiner,thomas.proisl,stefan.evert,besim.kabashi}@fau.de

## Abstract

This paper describes our system entered for the *SEM 2013 shared task on Semantic Textual Similarity (STS). We focus on the core task of predicting the semantic textual similarity of sentence pairs.

The current system utilizes machine learning techniques trained on semantic similarity ratings from the *SEM 2012 shared task; it achieved rank 20 out of 90 submissions from 35 different teams. Given the simple nature of our approach, which uses only WordNet and unannotated corpus data as external resources, we consider this a remarkably good result, making the system an interesting tool for a wide range of practical applications.

## 1 Introduction

The *SEM 2013 shared task on Semantic Textual Similarity (Agirre et al., 2013) required participants to implement a software system that is able to predict the semantic textual similarity (STS) of sentence pairs. Being able to reliably measure semantic similarity can be beneficial for many applications, e.g. in the domains of MT evaluation, information extraction, question answering, and summarization.

For the shared task, STS was measured on a scale ranging from 0 (indicating no similarity at all) to 5 (semantic equivalence). The system predictions were evaluated against manually annotated data.

## 2 Description of our approach

Our system KLUE-CORE uses two approaches to estimate STS between pairs of sentences: a distributional bag-of-words model inspired by Schütze (1998), and a simple alignment model that links each word in one sentence to the semantically most similar word in the other sentence. For the alignment model, word similarities were obtained from WordNet (using a range of state-of-the-art path-based similarity measures) and from two distributional semantic models (DSM).

All similarity scores obtained in this way were passed to a ridge regression learner in order to obtain a final STS score. The predictions for new sentence pairs were then transformed to the range $[0,5]$, as required by the task definition.

### 2.1 The training data

We trained our system on manually annotated sentence pairs from the STS task at SemEval 2012 (Agirre et al., 2012). Pooling the STS 2012 training and test data, we obtained 5 data sets from different domains, comprising a total of 5 343 sentence pairs annotated with a semantic similarity score in the range $[0,5]$. The data sets are paraphrase sentence pairs (MSRpar), sentence pairs from video descriptions (MSRvid), MT evaluation sentence pairs (MTnews and MTeuroparl), and glosses from two different lexical semantic resources (OnWN).

All sentence pairs were pre-processed with TreeTagger (Schmid, 1995)[1] for part-of-speech annotation and lemmatization.

[1] http://www.ims.uni-stuttgart.de/forschung/ressourcen/werkzeuge/treetagger.html



*Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task*, pages 181–186, Atlanta, Georgia, June 13-14, 2013. 

### 2.2 Similarity on word level

Our alignment model (Sec. 2.3.1) is based on similarity scores for pairs of words. We obtained a total of 11 different word similarity measures from WordNet (Miller et al., 1990) and in a completely unsupervised manner from distributional semantic models.

#### 2.2.1 WordNet

We computed three state-of-the-art WordNet similarity measures, namely path similarity, Wu-Palmer similarity and Leacock-Chodorow similarity (Budanitsky and Hirst, 2006). As usual, for each pair of words the synsets with the highest similarity score were selected. For all three measures, we made use of the implementations provided as part of the Natural Language ToolKit for Python (Bird et al., 2009).

#### 2.2.2 Distributional semantics

Word similarity scores were also obtained from two DSM: Distributional Memory (Baroni and Lenci, 2010) and a model compiled from a version of the English Wikipedia.[2] For Distributional Memory, we chose the collapsed $W \times W$ matricization, resulting in a $30\,686 \times 30\,686$ matrix that was further reduced to 300 latent dimensions using randomized SVD (Halko et al., 2009). For the Wikipedia DSM, we used a L2/R2 context window and mid-frequency feature terms, resulting in a $77\,598 \times 30\,484$ matrix. Co-occurrence frequency counts were weighted using sparse log-likelihood association scores with a square root transformation, and reduced to 300 latent dimensions with randomized SVD. In both cases, target terms are POS-disambiguated lemmas of content words, and the angle between vectors was used as a distance measure (equivalent to cosine similarity).

For each DSM, we computed the following semantic distances: (i) *angle*: the angle between the two word vectors; (ii) *fwdrank*: the (logarithm of the) forward neighbour rank, i.e. which rank the second word occupies among the nearest neighbours of the first word; (iii) *bwdrank*: the (logarithm of the) backward neighbour rank, i.e. which rank the first word occupies among the nearest neighbours of the second word; (iv) *rank*: the (logarithm of the) arithmetic mean of forward and backward neighbour rank; (v) *lowrank*: the (logarithm of the) harmonic mean of forward and backward neighbour rank.

A composite *similarity* score in the range $[0, 1]$ was obtained by linear regression on all five distance measures, using the WordSim-353 noun similarity ratings (Finkelstein et al., 2002) for parameter estimation. This score is referred to as *similarity* below. Manual inspection showed that word pairs with *similarity* $< 0.7$ were completely unrelated in many cases, so we also included a "strict" version of *similarity* with all lower scores set to 0. We further included *rank* and *angle*, which were linearly transformed to similarity values in the range $[0, 1]$.

### 2.3 Similarity on sentence level

Similarity scores for sentence pairs were obtained in two different ways: with a simple alignment model based on the word similarity scores from Sec. 2.2 (described in Sec. 2.3.1) and with a distributional bag-of-words model (described in Sec. 2.3.2).

#### 2.3.1 Similarity by word alignment

The sentence pairs were preprocessed in the following way: input words were transformed to lowercase; common stopwords were eliminated; and duplicate words within each sentence were deleted. For the word similarity scores from Sec. 2.2.2, POS-disambiguated lemmas according to the TreeTagger annotation were used.

Every word of the first sentence in a given pair was then compared with every word of the second sentence, resulting in a matrix of similarity scores for each of the word similarity measures described in Sec. 2.2. Since we were not interested in an asymmetric notion of similarity, matrices were set up so that the shorter sentence in a pair always corresponds to the rows of the matrix, transposing the similarity matrix if necessary. From each matrix, two similarity scores for the sentence pair were computed: the arithmetic mean of the row maxima (marked as *short* in Tab. 4), and the artihmetic mean of the column maxima (marked as *long* in Tab. 4).

This approach corresponds to a simple word alignment model where each word in the shorter sentence is aligned to the semantically most similar word in the longer sentence (*short*), and vice versa (*long*). Note that multiple source words may be aligned to the same target word, and target words can remain

[2] For this purpose, we used the pre-processed and linguistically annotated Wackypedia corpus available from `http://wacky.sslmit.unibo.it/`.

unaligned without penalty. Semantic similarities are then averaged across all alignment pairs.

In total, we obtained 22 sentence similarity scores from this approach.

#### 2.3.2 Distributional similarity

We computed distributional similarity between the sentences in each pair directly using bag-of-words centroid vectors as suggested by Schütze (1998), based on the two word-level DSM introduced in Sec. 2.2.2.

For each sentence pair and DSM, we computed (i) the angle between the centroid vectors of the two sentences and (ii) a z-score relative to all other sentences in the same data set of the training or test collection. Both values are measures of semantic distance, but are automatically transformed into similarity measures by the regression learner (Sec. 2.4).

For the z-scores, we computed the semantic distance (i.e. angle) between the first sentence of a given pair and the second sentences of *all* word pairs in the same data set. The resulting list of angles was standardized to z-scores, and the z-score corresponding to the second sentence from the given pair was used as a measure of forward similarity between the first and second sentence. In the same way, a backward z-score between the second and first sentence was determined. We used the average of the forward and backward z-score as our second STS measure.

The z-transformation was motivated by our observation that there are substantial differences between the individual data sets in the STS 2012 training and test data. For some data sets (MSRpar and MSRvid), sentences are often almost identical and even a single-word difference can result in low similarity ratings; for other data sets (e.g. OnWN), similarity ratings seem to be based on the general state of affairs described by the two sentences rather than their particular wording of propositional content. By using other sentences in the same data set as a frame of reference, corpus-based similarity scores can roughly be calibrated to the respective notion of STS.

In total, we obtained 4 sentence (dis)similarity scores from this approach. Because of technical issues, only the z-score measures were used in the submitted system. The experiments in Sec. 3 also focus on these z-scores.

### 2.4 The regression model

The 24 individual similarity scores described in Sec. 2.3.1 and 2.3.2 were combined into a single STS prediction by supervised regression.

We conducted experiments with various machine learning algorithms implemented in the Python library scikit-learn (Pedregosa et al., 2011). In particular, we tested linear regression, regularized linear regression (ridge regression), Bayesian ridge regression, support vector regression and regression trees. Our final system submitted to the shared task uses ridge regression, a shrinkage method applied to linear regression that uses a least-squares regularization on the regression coefficients (Hastie et al., 2001, 59). Intuitively speaking, the regularization term discourages large value of the regression coefficients, which makes the learning technique less prone to overfitting quirks of the training data, especially with large numbers of features.

We tried to optimise our results by training the individual regressors for each test data set on appropriate portions of the training data. For our task submission, we used the following training data based on educated guesses inspired by the very small amount of development data provied: for the headlines test set we trained on both glosses and statistical MT data, for the OnWN and FNWN test sets we trained on glosses only (OnWN), and for the SMT test set we trained on statistical MT data only (MTnews and MTeuroparl). We decided to omit the Microsoft Research Paraphrase Corpus (MSRpar and MSRvid) because we felt that the types of sentence pairs in this corpus were too different from the development data.

For our submission, we used all 24 features described in Sec. 2.3 as input for the ridge regression algorithm. Out of 90 submissions by 35 teams, our system ranked on place 20.[3]

## 3 Experiments

In this section, we describe some post-hoc experiments on the STS 2013 test data, which we performed in order to find out whether we made good decisions regarding the machine learning method, training data,

[3]This paper describes the run listed as KLUE-approach_2 in the official results. The run KLUE-approach_1 was produced by the same system without the bag-of-words features (Sec. 2.3.2); it was only submitted as a safety backup.

similarity features, and other parameters. Results of our submitted system are typeset in italics, the best results in each column are typeset in bold font.

### 3.1 Machine learning algorithms

Tab. 1 gives an overview of the performance of various machine learning algorithms. All regressors were trained on the same combinations of data sets (see Sec. 2.4 above) using all available features, and evaluated on the STS 2013 test data. Overall, our choice of ridge regression is justified. Especially for the OnWN test set, however, support vector regression is considerably better (it would have achieved rank 11 instead of 17 on this test set). If we had happened to use the best learning algorithm for each test set, we would have achieved a mean score of 0.54768 (putting our submission at rank 14 instead of 20).

### 3.2 Regularization strength

We also experimented with different regularization strengths, as determined by the parameter $\alpha$ of the ridge regression algorithm (see Tab. 2). Changing $\alpha$ from its default value $\alpha = 1$ does not seem to have a large impact on the performance of the regressor. Setting $\alpha = 2$ for all test sets would have minimally improved the mean score (rank 19 instead of 20). Even choosing the optimal $\alpha$ for each test set would only have resulted in a slightly improved mean score of 0.53811 (also putting our submission at rank 19).

### 3.3 Composition of training data

As described above, we suspected that using different combinations of the training data for different test sets might lead to better results. The overview in Tab. 3 confirms our expectations. We did, however, fail to correctly guess the optimal combinations for each test set. We would have obtained the best results by training on glosses (OnWN) for the headlines test set (rank 35 instead of 40 in this category), by training on MSR data (MSRpar and MSRvid) for the OnWN (rank 11 instead of 17) and FNWN test sets (rank 9 instead of 10), and by combining glosses and machine translation data (OnWN, MTnews MTeuroparl) for the SMT test set (rank 30 instead of 33). Had we found the optimal training data for each test set, our system would have achieved a mean score of 0.55021 (rank 11 instead of 20).

### 3.4 Features

For our submission, we used all the features described in Sec. 2. Tab. 4 shows what results each group of features would have achieved by itself (all runs use ridge regression, default $\alpha = 1$ and the same combinations of training data as in our submission).

In Tab. 4, the line labelled *wp500* shows the results obtained using only word-alignment similarity scores (Sec. 2.3.1) based on the Wikipedia DSM (Sec. 2.2.2) as features. The following two lines give separate results for the alignments from shorter to longer sentence, i.e. row maxima (*wp500-short*) and from longer to shorter sentence, i.e. column maxima (*wp500-long*), respectively. Below are corresponding results for word alignments based on Distributional Memory (*dm*, *dm-short*, *dm-long*) and WordNet similarity as described in Sec. 2.2.1 (*WN*, *WN-short*, *WN-long*). The line labelled *bow* represents the two z-score similarities obtained from distributional bag-of-words models (Sec. 2.3.2); *bow-wp500* (Wikipedia DSM) and *bow-dm* (Distributional Memory) each correspond to a single distributional feature.

Combining all the available features indeed results in the highest mean score. However, for OnWN and SMT a subset of the features would have led to better results. Using only the bag-of-words scores would have improved the results for the OnWN test set by a considerable margin (rank 8 instead of 17), using only the alignment scores based on WordNet would have improved the results for the SMT test set (rank 17 instead of 33). If we had used the optimal subset of features for each test set, the mean score would have increased to 0.55556 (rank 9 instead of 20).

## 4 Conclusion

Our experiments show that it is essential for high-quality semantic textual similarity to adapt a corpus-based system carefully to each particular data set (choice of training data, feature engineering, tuning of machine learning algorithm). Many of our educated guesses for parameter settings turned out to be fairly close to the optimal values, though there would have been some room for improvement.

Overall, our simple approach, which makes very limited use of external resources, performs quite well – achieving rank 20 out of 90 submissions – and will be a useful tool for many real-world applications.

| | headlines | OnWN | FNWN | SMT | mean |
|---|---|---|---|---|---|
| Ridge Regression | *0.65102* | *0.68693* | *0.41887* | ***0.33599*** | ***0.53546*** |
| Linear Regression | **0.65184** | 0.68118 | 0.39707 | 0.32756 | 0.52966 |
| Bayesian Ridge | 0.65164 | 0.68962 | **0.42344** | 0.33003 | 0.53474 |
| SVM SVR | 0.52208 | **0.73330** | 0.40479 | 0.30810 | 0.49357 |
| Decision Tree | 0.29320 | 0.50633 | 0.05022 | 0.17072 | 0.28510 |

Table 1: Evaluation results for different machine learning algorithms

| $\alpha$ | headlines | OnWN | FNWN | SMT | mean |
|---|---|---|---|---|---|
| 1 | *0.65102* | *0.68693* | *0.41887* | *0.33599* | *0.53546* |
| 0.01 | 0.65184 | 0.68129 | 0.39773 | 0.32773 | 0.52980 |
| 0.1 | **0.65186** | 0.68224 | 0.40246 | 0.32900 | 0.53087 |
| 0.5 | 0.65161 | 0.68492 | 0.41346 | 0.33311 | 0.53374 |
| 0.9 | 0.65114 | 0.68660 | 0.41816 | 0.33560 | 0.53523 |
| 2 | 0.64941 | 0.68917 | **0.42290** | **0.33830** | **0.53659** |
| 5 | 0.64394 | **0.69197** | 0.42265 | 0.33669 | 0.53491 |

Table 2: Evaluation results for different regularization strengths of the ridge regression learner

| | headlines | OnWN | FNWN | SMT | mean |
|---|---|---|---|---|---|
| def | **0.65440** | *0.68693* | *0.41887* | 0.32694 | 0.53357 |
| smt | 0.65322 | 0.62643 | 0.24895 | *0.33599* | 0.50684 |
| def+smt | *0.65102* | 0.59665 | 0.24953 | **0.33867** | 0.49962 |
| msr | 0.63633 | **0.73396** | **0.43073** | 0.33168 | **0.54185** |
| def+smt+msr | 0.65008 | 0.65093 | 0.39636 | 0.28645 | 0.50777 |
| approach$_2$ | *0.65102* | *0.68693* | *0.41887* | *0.33599* | *0.53546* |

Table 3: Evaluation results for different training sets ("approach$_2$" refers to our shared task submission, cf. Sec. 2.4)

| | headlines | OnWN | FNWN | SMT | mean |
|---|---|---|---|---|---|
| wp500 | 0.57099 | 0.59199 | 0.31740 | 0.31320 | 0.46899 |
| wp500-long | 0.57837 | 0.59012 | 0.30909 | 0.30075 | 0.46614 |
| wp500-short | 0.58271 | 0.58845 | 0.34205 | 0.29474 | 0.46794 |
| dm | 0.42129 | 0.55945 | 0.21139 | 0.27426 | 0.38910 |
| dm-long | 0.40709 | 0.56511 | 0.28993 | 0.23826 | 0.38037 |
| dm-short | 0.44780 | 0.53555 | 0.28709 | 0.24484 | 0.38853 |
| WN | 0.63654 | 0.65149 | 0.41025 | **0.35624** | 0.52783 |
| WN-long | 0.62749 | 0.63828 | 0.39684 | 0.33399 | 0.51297 |
| WN-short | 0.64986 | 0.66175 | 0.41441 | 0.33350 | 0.52759 |
| bow | 0.52384 | **0.74046** | 0.31917 | 0.24611 | 0.46808 |
| bow-wp500 | 0.52726 | 0.73624 | 0.32797 | 0.24460 | 0.46841 |
| bow-dm | 0.21908 | 0.66873 | 0.17096 | 0.20176 | 0.32138 |
| all | ***0.65102*** | *0.68693* | ***0.41887*** | *0.33599* | ***0.53546*** |

Table 4: Evaluation results for different sets of similarity scores as features (cf. Sec. 3.4)

## References


Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2012. Semeval-2012 task 6: A pilot on semantic textual similarity. In *First Joint Conference on Lexical and Computational Semantics*, pages 385–393. Association for Computational Linguistics.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *sem 2013 shared task: Semantic textual similarity, including a pilot on typed-similarity. In **SEM 2013: The Second Joint Conference on Lexical and Computational Semantics*. Association for Computational Linguistics.

Marco Baroni and Alessandro Lenci. 2010. Distributional Memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4):673–712.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'Reilly Media, Sebastopol, CA. Online version available at `http://www.nltk.org/book`.

Alexander Budanitsky and Graeme Hirst. 2006. Evaluating WordNet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32(1):13–47.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2002. Placing search in context: The concept revisited. *ACM Transactions on Information Systems*, 20(1):116–131.

N. Halko, P. G. Martinsson, and J. A. Tropp. 2009. Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions. Technical Report 2009-05, ACM, California Institute of Technology, September.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2001. *The Elements of Statistical Learning. Data Mining, Inference, and Prediction*. Springer, New York, NY.

George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J. Miller. 1990. Introduction to WordNet: An on-line lexical database. *International Journal of Lexicography*, 3(4):235–244.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Helmut Schmid. 1995. Improvements in part-of-speech tagging with an application to German. In *Proceedings of the EACL SIGDAT-Workshop*, pages 47–50, Dublin.

Hinrich Schütze. 1998. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123.