ECNU: One Stone Two Birds: Ensemble of Heterogenous Measures for Semantic Relatedness and Textual Entailment

Jiang Zhao, Tian Tian Zhu, Man Lan*
Department of Computer Science and Technology
East China Normal University
`51121201042,51111201046@ecnu.cn; mlan@cs.ecnu.edu.cn`*

Abstract

This paper presents our approach to semantic relatedness and textual entailment subtasks organized as task 1 in SemEval 2014. Specifically, we address two questions: (1) Can we solve these two subtasks together? (2) Are features proposed for textual entailment task still effective for semantic relatedness task? To address them, we extracted seven types of features including text difference measures proposed in entailment judgement subtask, as well as common text similarity measures used in both subtasks. Then we exploited the same feature set to solve the both subtasks by considering them as a regression and a classification task respectively and performed a study of influence of different features. We achieved the first and the second rank for relatedness and entailment task respectively.

1 Introduction

Distributional Semantic Models (DSMs)(surveyed in (Turney et al., 2010)) exploit the co-occurrences of other words with the word being modeled to compute the semantic meaning of the word under the distributional hypothesis: "similar words share similar contexts" (Harris, 1954). Despite their success, DSMs are severely limited to model the semantic of long phrases or sentences since they ignore grammatical structures and logical words. Compositional Distributional Semantic Models (CDSMs)(Zanzotto et al., 2010; Socher et al., 2012) extend DSMs to sentence level to capture the *compositionality* in the semantic vector space, which has seen a rapidly growing interest in recent years. Although several CDSMs have been proposed, benchmarks are lagging behind. Previous work (Grefenstette and Sadrzadeh, 2011; Socher et al., 2012) performed experiments on their own datasets or on the same datasets which are limited to a few hundred instances of very short sentences with a fixed structure.

To provide a benchmark so as to compare different CDSMs, the sentences involving compositional knowledge task in SemEval 2014 (Marelli et al., 2014) develops a large dataset which is full of lexical, syntactic and semantic phenomena. It consists of two subtasks: semantic relatedness task, which measures the degree of semantic relatedness of a sentence pair by assigning a relatedness score ranging from 1 (completely unrelated) to 5 (very related); and textual entailment (TE) task, which determines whether one of the following three relationships holds between two given sentences A and B: (1) *entailment*: the meaning of B can be inferred from A; (2) *contradiction*: A contradicts B; (3) *neutral*: the truth of B cannot be inferred on the basis of A.

Semantic textual similarity (STS) (Lintean and Rus, 2012) and semantic relatedness are closely related and interchangeably used in many literatures except that the concept of semantic similarity is more specific than semantic relatedness and the latter includes concepts as antonymy and meronymy. In this paper we regard the semantic relatedness task as a STS task. Besides, regardless of the original intention of this task, we adopted the mainstream machine learning methods instead of CDSMs to solve these two tasks by extracting heterogenous features.

Like semantic relatedness, TE task (surveyed in (Androutsopoulos and Malakasiotis, 2009)) is also closely related to STS task since in TE task lots of similarity measures at different levels are exploited to boost classification. For example, (Malakasiotis and Androutsopoulos, 2007) used ten string similarity measures such as cosine similarity at the word and the character level. Therefore, the first fundamental question arises, i.e., "Can we solve both of these two tasks together?" At the same time, since high similarity does not mean entailment holds, the TE task also utilizes other features besides similarity measures. For example, in our previous work (Zhao et al., 2014) text difference features were proposed and proved to be effective. Therefore, the second question surfaces here, i.e., "Are features proposed for TE task still effective for STS task?" To answer the first question, we extracted seven types of features including text similarity and text difference and then fed them to classifiers and regressors to solve TE and STS task respectively. Regarding the second question, we conducted a series of experiments to study the performance of different features for these two tasks.

The rest of the paper is organized as follows. Section 2 briefly describes the related work on STS and TE tasks. Section 3 presents our systems including features, learning methods, etc. Section 4 shows the experimental results on training data and Section 5 reports the results of our submitted systems on test data and gives a detailed analysis. Finally, Section 6 concludes this paper with future work.

2 Related Work

Existing work on STS can be divided into 4 categories according to the similarity measures used (Gomaa and Fahmy, 2013): (1) string-based method (Bär et al., 2012; Malakasiotis and Androutsopoulos, 2007) which calculates similarities using surface strings at either character level or word level; (2) corpus-based method (Li et al., 2006) which measures word or sentence similarities using the information gained from large corpora, including Latent Semantic Analysis (LSA), pointwise mutual information (PMI), etc. (3) knowledge-based method (Mihalcea et al., 2006) which estimates similarities with the aid of external resources, such as WordNet[1]; (4) hybrid method (Zhu and Lan, 2013; Croce et al., 2013) which integrates multiple similarity measures and adopts supervised machine learning algorithms to learn the different contributions of different features.

The approaches to the task of TE can be roughly divided into two groups: (1) logic inference method (Bos and Markert, 2005) where automatic reasoning tools are used to check the logical representations derived from sentences and (2) machine learning method (Zhao et al., 2013; Gomaa and Fahmy, 2013) where a supervised model is built using a variety of similarity scores.

Unlike previous work which separately addressed these two closely related tasks by using simple feature types, in this paper we endeavor to simultaneously solve these two tasks by using heterogenous features.

3 Our Systems

We consider the two tasks as one by exploiting the same set of features but using different learning methods, i.e., classification and regression. Seven types of features are extracted and most of them are based on our previous work on TE (Zhao et al., 2014) and STS (Zhu and Lan, 2013). Many learning algorithms and parameters are examined and the final submitted systems are configured according to the preliminary results on training data.

3.1 Preprocessing

Three text preprocessing operations were performed before we extracted features, which included: (1) we converted the contractions to their formal writings, for example, *doesn't* is rewritten as *does not*. (2) the WordNet-based Lemmatizer implemented in Natural Language Toolkit[2] was used to lemmatize all words to their nearest base forms in WordNet, for example, *was* is lemmatized to *be*. (3) we replaced a word from one sentence with another word from the other sentence if the two words share the same meaning, where WordNet was used to look up synonyms. No word sense disambiguation was performed and all synsets for a particular lemma were considered.

3.2 Feature Representations

3.2.1 Length Features (len)

Given two sentences A and B, this feature type records the length information using the follow-

[1]http://wordnet.princeton.edu/

[2]http://nltk.org/

ing eight measure functions:
$|A|, |B|, |A-B|, |B-A|, |A \cup B|, |A \cap B|, \frac{(|A|-|B|)}{|B|}, \frac{(|B|-|A|)}{|A|}$
where $|A|$ stands for the number of non-repeated words in sentence A , $|A-B|$ means the number of unmatched words found in A but not in B , $|A \cup B|$ stands for the set size of non-repeated words found in either A or B and $|A \cap B|$ means the set size of shared words found in both A and B .

Moreover, in consideration of different types of words make different contributions to text similarity, we also recorded the number of words in set $A-B$ and $B-A$ whose POS tags are noun, verb, adjective and adverb respectively. We used Stanford POS Tagger[3] for POS tagging. Finally, we collected a total of sixteen features.

3.2.2 Surface Text Similarity (st)

As shown in Table 1, we adopted six commonly used functions to calculate the similarity between sentence A and B based on their surface forms, where $\overrightarrow{x}$ and $\overrightarrow{y}$ are vectorial representations of sentences A and B in $tf * idf$ schema.

Measure	Definition
Jaccard	$S_{jacc} = \lvert A \cap B\rvert / \lvert A \cup B\rvert$
Dice	$S_{dice} = 2 * \lvert A \cap B\rvert / (\lvert A\rvert + \lvert B\rvert)$
Overlap	$S_{over} = \lvert A \cap B\rvert / \lvert A\rvert$ and $\lvert A \cap B\rvert / \lvert B\rvert$
Cosine	$S_{cos} = \overrightarrow{x} \cdot \overrightarrow{y} / (\lVert \overrightarrow{x} \rVert \cdot \lVert \overrightarrow{y} \rVert)$
Manhattan	$M(\overrightarrow{x}, \overrightarrow{y}) = \sum_{i=1}^{n} \lvert x_i - y_i\rvert$
Euclidean	$E(\overrightarrow{x}, \overrightarrow{y}) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$

Table 1: Surface text similarity measures and their definitions used in our experiments.

We also used three statistical correlation coefficients (i.e., Pearson, Spearmanr, Kendalltau) to measure similarity by regarding the vectorial representations as different variables. Thus we got ten features at last.

3.2.3 Semantic Similarity (ss)

The above surface text similarity features only consider the surface words rather than their actual meanings in sentences. In order to build the semantic representations of sentences, we used a latent model to capture the contextual meanings of words. Specifically, we adopted the weighted textual matrix factorization (WTMF) (Guo and Diab, 2012) to model the semantics of sentences due to its reported good ability to model short texts. This model first factorizes the original term-sentence matrix X into two matrices such that $X_{i,j} \approx P_{*,i}^{T} \cdot Q_{*,j}$, where $P_{*,i}$ is a latent semantic vector profile for word w_i and $Q_{*,j}$ is a vector profile that represents the sentence s_j. Then we employed the new representations of sentences, i.e., Q, to calculate the semantic similarity between sentences using Cosine, Manhattan, Euclidean, Pearson, Spearmanr, Kendalltau measures respectively, which results in six features.

3.2.4 Grammatical Relationship (gr)

The grammatical relationship feature measures the semantic similarity between two sentences at the grammar level and this feature type was also explored in our previous work (Zhao et al., 2013; Zhu and Lan, 2013). We used Stanford Parser[4] to acquire the dependency information from sentences and the grammatical information are represented in the form of relation unit, e.g. *nsubj*(example, this), where *nsubj* stands for a dependency relationship between *example* and *this*. We obtained a sequence of relation units for each sentence and then used them to estimate similarity by adopting eight measure functions described in Section 3.2.1, resulting in eight features.

3.2.5 Text Difference Measures (td)

There are two types of text difference measures. The first feature type is specially designed for the *contradiction* entailment relationship, which is based on the following observation: there exist antonyms between two sentences or the negation status is not consistent (i.e., one sentence has a negation word while the other does not have) if *contradiction* holds. Therefore we examined each sentence pair and set this feature as 1 if at least one of these conditions is met, otherwise -1. WordNet was used to look up antonyms and a negation list with 28 words was used.

The second feature type is extracted from two word sets $A-B$ and $B-A$ as follows: we first calculated the similarities between every word from $A-B$ and every word from $B-A$, then took the maximum, minimum and average value of them as features. In our experiments, four WordNet-based similarity measures (i.e., *path, lch, wup, jcn* (Gomaa and Fahmy, 2013)) were used to calculate the similarity between two words.

Totally, we got 13 text difference features.

[3]http://nlp.stanford.edu/software/tagger.shtml

[4]http://nlp.stanford.edu/software/lex-parser.shtml

3.2.6 String Features (str)

This set of features is taken from our previous work (Zhu and Lan, 2013) due to its superior performance.

Longest common sequence (LCS) We computed the LCS similarity on the original and lemmatized sentences. It was calculated by finding the maximum length of a common contiguous subsequence of two strings and then dividing it by the smaller length of two strings to eliminate the impacts of length imbalance.

Jaccard similarity using *n*-grams We obtained *n*-grams at three different levels, i.e., the original word level, the lemmatized word level and the character level. Then these *n*-grams were used for calculating Jaccard similarity defined in Table 1. In our experiments, $n = \{1, 2, 3\}$ were used for the word level and $n = \{2, 3, 4\}$ were used for the character level.

Weighted word overlap (WWO) Since not all words are equally important, the traditional Overlap similarity may not be always reasonable. Thus we used the information content of word w to estimate the importance of word w as follows:

$$ic(w) = \ln \frac{\sum_{w' \in C} freq(w')}{freq(w)}$$

where C is the set of words in the corpus and $freq(w)$ is the frequency of the word w in the corpus. To compute $ic(w)$, we used the Web 1T 5-gram Corpus [5]. Then the WWO similarity of two sentence s_1 and s_2 was calculated as follows:

$$Sim_{wwo}(s_1, s_2) = \frac{\sum_{w \in s_1 \cap s_2} ic(w)}{\sum_{w' \in s_2} ic(w')}$$

Due to its asymmetry, we used the harmonic mean of $Sim_{wwo}(s_1, s_2)$ and $Sim_{wwo}(s_2, s_1)$ as the final WWO similarity. The WWO similarity is calculated on the original and lemmatized strings respectively.

Finally, we got two LCS features, nine Jaccard *n*-gram features and two WWO features.

3.2.7 Corpus-based Features (cps)

Two types of corpus-based feature are also borrowed from our previous work (Zhu and Lan, 2013), i.e., vector space sentence similarity and co-occurrence retrieval model (CRM), which results in six features.

[5]https://catalog.ldc.upenn.edu/LDC2006T13

Co-occurrence retrieval model (CRM) The CRM word similarity is calculated as follows:

$$Sim_{CRM}(w_1, w_2) = \frac{2 * |c(w_1) \cap c(w_2)|}{|c(w_1)| + |c(w_2)|}$$

where $c(w)$ is the set of words that co-occur with word w. We used the 5-gram part of the Web 1T 5-gram Corpus to obtain $c(w)$. We only considered the word w with $|c(w)| > T$ and then took the top 200 co-occurring words ranked by the co-occurrence frequency as its $c(w)$. In our experiment, we set $T = \{50, 200\}$. To propagate the similarity from words to sentences, we adopted the best alignment strategy used in (Banea et al., 2012) to align two sentences.

Vector space sentence similarity This feature set is taken from (Šarić et al., 2012), which is based on distributional vectors of words. First we performed latent semantic analysis (LSA) over two corpora, i.e., the New York Times Annotated Corpus (NYT) (Sandhaus, 2008) and Wikipedia, to estimate the distributions of words. Then we used two strategies to convert the distributional meanings of words to sentence level: (i) simply summing up the distributional vector of each word w in the sentence, (ii) using the information content $ic(w)$ to weigh the LSA vector of each word w and summing them up. Then we used cosine similarity to measure the similarity of two sentences.

3.3 Learning Algorithms

We explored several classification algorithms to classify entailment relationships and regression algorithms to predict similarity scores using the above 72 features after performing max-min standardization procedure by scaling them to [-1,1]. Five supervised learning methods were explored: Support Vector Machine (SVM) which makes the decisions according to the hyperplanes, Random Forest (RF) which constructs a multitude of decision trees at training time and selects the mode of the classes output by individual trees, Gradient Boosting (GB) that produces a prediction model in the form of an ensemble of weak prediction models, *k*-nearest neighbors (*k*NN) that decides the class labels with the aid of the classes of *k* nearest neighbors, and Stochastic Gradient Descent (SGD) which uses SGD technique to minimize loss functions. These supervised learning methods are implemented in scikit-learn toolkit (Pedregosa et al., 2011). Besides, we also used a semi-supervised learning strategy for both tasks in order to make full use of unlabeled test data. Specifically, the co-training algorithm was used to address TE task according to (Zhao et al., 2014). Its strategy is to train two classifiers with two data views and to add the top confident predicted instances by one classifier to expand the training set of another classifier and then to re-train the two classifiers on the expanded training sets. For STS task, we utilized CoReg algorithm (Zhou and Li, 2005) which uses two *k*NN regressors to perform co-training paradigm.

3.4 Evaluation Measures

In order to evaluate the performance of different algorithms, we adopted the official evaluation measures, i.e., *Pearson* correlation coefficient for STS task and *accuracy* for TE task.

4 Experiments on Training Data

To make a reasonable comparison between different algorithms, we performed 5-fold cross validation on training data with 5000 sentence pairs. The parameters tuned in different algorithms are listed below: the trade-off parameter *c* in SVM, the number of trees *n* in RF, the number of boosting stages *n* in GB, the number of nearest neighbors *k* in *k*NN and the number of passes over the training data *n* in SGD. The rest parameters are set to be default.

Algorithm	STS task		TE task	
	Pearson	para.	*Accuracy*	para.
SVM	.807±.058	c=10	83.46±2.09	c=100
RF	.805±.052	*n*=40	83.16±2.64	*n*=30
GB	.806±.055	*n*=210	83.22±2.48	*n*=140
kNN	.797±.062	*k*=25	82.54±2.45	*k*=17
SGD	.765±.064	*n*=29	78.88±1.99	*n*=15

Table 2: The 5-fold cross validation results on training data with mean and standard deviation for each algorithm.

Table 2 reports the experimental results of 5-fold cross validation with mean and standard deviation and the optimal parameters on training data. The results of semi-supervised learning methods are not listed because only a few parameters are tried due to the limit of time. From this table we see that SVM, RF and GB perform comparable results to each other.

5 Results on Test Data

5.1 Submitted System Configurations

According to the above preliminary experimental results, we configured five final systems for each task. Table 3 presents the classification and regression algorithms with their parameters used in the five systems for each task.

System	STS task	TE task
1	SVR, *c*=10	SVC, *c*=100
2	GB, *n*=210	GB, *n*=140
3	RF, *n*=40	RF, *n*=30
4	CoReg, *k*=13	co-training, *k*=40
5	majority voting	majority voting

Table 3: Five system configurations for test data for two tasks.

Among them, System 1 acts as our primary and baseline system that employs SVM algorithm and as comparison System 2 and System 3 exploit GB and RF algorithm respectively. Unlike supervised settings in the aforementioned systems, System 4 employs a semi-supervised learning strategy to make use of unlabeled test data. For CoReg, the number of iteration and the number of nearest neighbors are set as 100 and 13 respectively, and for each iteration in co-training, the number of confident predictions is set as 40. To further improve performance, System 5 combines the results of 5 different algorithms (i.e. MaxEnt, SVM, *k*NN, GB, RF) through majority voting. We used the averaged values of the outputs from different regressors as final similarity scores for semantic similarity measurement task and chose the major class label for entailment judgement task.

5.2 Results and Discussion

Table 4 lists the final results officially released by the organizers in terms of *Pearson* and *accuracy*. The best performance among these five systems is shown in bold font. All participants can submit a maximum of five runs for each task and only one primary system is involved in official ranking. The lower part of Table 4 presents the top 3 results and the results with $*$ are achieved by our systems.

System	STS task	TE task(%)
1	0.8279	83.641
2	0.8389	**84.128**
3	**0.8414**	83.945
4	0.8210	81.165
5	0.8349	83.986
rank 1st	0.8279*	84.575
rank 2nd	0.8272	83.641*
rank 3rd	0.8268	83.053

Table 4: The results of our five systems for two tasks and the officially top-ranked systems.

From this table, we found that (1) System 3 (using GB algorithm) and System 2 (using RF algorithm) achieve the best performance among three supervised systems in STS and TE task respectively. However, there is no significant difference among these systems. (2) Surprisingly, the semi-supervised system (i.e., System 4) that employs the co-training strategy to make use of test data performs the worst, which is beyond our expectation. Based on our further observation in TE task, the possible reason is that a lot of misclassified examples are added into the training pool in the initial iteration, which results in worse models built in the subsequent iterations. And we speculate that the weak learner *k*NN employed in CoReg may lead to poor performance as well. (3) The majority voting strategy fails to boost the performance since GB and RF algorithm obtain the best performance among these algorithms. (4) Our systems obtain very good results on both STS and TE task, i.e., we rank 1st out of 17 participants in STS task and rank 2nd out of 18 participants in TE task according to the results of primary systems and as shown in Table 4 our primary system (i.e., System 1) do not achieve the best performance.

In a nutshell, our systems rank first and second in STS and TE task respectively. Therefore the answer to the first question raised in Section 1 is *yes*. For two tasks, i.e., STS and TE, which are very closely related but slightly different, we can use the same features to solve them together.

5.3 Feature Combination Experiments

To answer the second question and explore the influences of different feature types, we performed a series of experiments under the best system setting. Table 5 shows the results of different feature combinations where for each time we selected and added one best feature type. From this table, we find that for STS the most effective feature is *cps* and for TE task is *td*. Almost all feature types have positive effects on performance. Specifically, *td* alone achieves 81.063% in TE task which is quite close to the best performance (84.128%) and *cps* alone achieves 0.7544 in STS task. Moreover, the *td* feature proposed for TE task is quite effective in STS task as well, which suggests that text semantic difference measures are also crucial when measuring sentence similarity.

Therefore the answer to the second question is *yes*. It is clear that the features proposed for TE are also effective for STS and heterogenous features yield better performance than a single feature type.

len	st	ss	gr	td	str	cps	result
						+	0.7544 (STS)
				+		+	0.8057(+5.13)
		+		+		+	0.8280(+2.23)
		+	+	+		+	0.8365(+0.85)
	+	+	+	+		+	0.8426(+0.61)
	+	+	+	+	+	+	**0.8432**(+0.06)
+	+	+	+	+	+	+	0.8429(-0.03)
				+			81.063 (TE)
	+			+			82.484(+1.421)
	+			+	+		82.992(+0.508)
	+			+	+	+	83.844(+0.852)
	+		+	+	+	+	83.925(+0.081)
	+	+	+	+	+	+	84.067(+0.142)
+	+	+	+	+	+	+	**84.128**(+0.061)

Table 5: Results of feature combinations, the numbers in the brackets are the performance increments compared with the previous results.

6 Conclusion

We set up five state-of-the-art systems and each system employs different classifiers or regressors using the same feature set. Our submitted systems rank the 1st out of 17 teams in STS task with the best performance of 0.8414 in terms of *Pearson* coefficient and rank the 2nd out of 18 teams in TE task with 84.128% in terms of *accuracy*. This result indicates that (1) we can use the same feature set to solve these two tasks together, (2) the features proposed for TE task are also effective for STS task and (3) heterogenous features outperform a single feature. For future work, we may explore the underlying relationships between these two tasks to boost their performance by each other.

Acknowledgments

This research is supported by grants from National Natural Science Foundation of China (No.60903093) and Shanghai Knowledge Service Platform Project (No. ZF1213).

References

Ion Androutsopoulos and Prodromos Malakasiotis. 2009. A survey of paraphrasing and textual entailment methods. *arXiv preprint arXiv:0912.3747*.

Carmen Banea, Samer Hassan, Michael Mohler, and Rada Mihalcea. 2012. Unt:a supervised synergistic approach to semantictext similarity. In *First Joint Conference on Lexical and Computational Semantics (*SEM*.

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*, pages 435–440. Association for Computational Linguistics.

Johan Bos and Katja Markert. 2005. Recognising textual entailment with logical inference. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 628–635. Association for Computational Linguistics.

Danilo Croce, Valerio Storch, and Roberto Basili. 2013. Unitor-core typed: Combining text similarity and semantic filters through sv regression. In *Proceedings of the 2nd Joint Conference on Lexical and Computational Semantics*, page 59.

Wael H Gomaa and Aly A Fahmy. 2013. A survey of text similarity approaches. *International Journal of Computer Applications*, 68(13):13–18.

Edward Grefenstette and Mehrnoosh Sadrzadeh. 2011. Experimental support for a categorical compositional distributional model of meaning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1394–1404. Association for Computational Linguistics.

Weiwei Guo and Mona Diab. 2012. Modeling sentences in the latent space. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*.

Zellig S Harris. 1954. Distributional structure. *The Philosophy of Linguistics,*.

Yuhua Li, David McLean, Zuhair A Bandar, James D O'shea, and Keeley Crockett. 2006. Sentence similarity based on semantic nets and corpus statistics. *Knowledge and Data Engineering, IEEE Transactions on*, 18(8):1138–1150.

Mihai C. Lintean and Vasile Rus. 2012. Measuring semantic similarity in short texts through greedy pairing and word semantics. In *FLAIRS Conference*. AAAI Press.

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*, pages 42–47. Association for Computational Linguistics.

M. Marelli, L. Bentivogli, M. Baroni, R. Bernardi, S. Menini, and R. Zamparelli. 2014. Semeval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. In *Proceedings of SemEval 2014: International Workshop on Semantic Evaluation*.

Rada Mihalcea, Courtney Corley, and Carlo Strapparava. 2006. Corpus-based and knowledge-based measures of text semantic similarity. In *AAAI*, volume 6, pages 775–780.

Fabian Pedregosa, Gaël. Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Evan Sandhaus. 2008. The new york times annotated corpus ldc2008t19. *Philadelphia: Linguistic Data Consortium*.

Socher, Richard, Huval Brody, Manning Christopher, and Ng Andrew. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of EMNLP*, Jeju Island, Korea.

Peter D Turney, Patrick Pantel, et al. 2010. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37(1):141–188.

Frane Šarić, Goran Glavaš, Mladen Karan, Jan Šnajder, and Bojana Dalbelo Bašić. 2012. Takelab: Systems for measuring semantic text similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics*, pages 441–448, Stroudsburg, PA, USA. Association for Computational Linguistics.

Fabio Massimo Zanzotto, Ioannis Korkontzelos, Francesca Fallucchi, and Suresh Manandhar. 2010. Estimating linear models for compositional distributional semantics. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1263–1271. Association for Computational Linguistics.

Jiang Zhao, Man Lan, and Zheng-Yu Niu. 2013. Ecnucs: Recognizing cross-lingual textual entailment using multiple text similarity and text difference measures. In *Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 118–123, Atlanta, Georgia, USA, June. Association for Computational Linguistics.

Jiang Zhao, Man Lan, Zheng-Yu Niu, and Donghong Ji. 2014. Recognizing cross-lingual textual entailment with co-training using similarity and difference views. In *The 2014 International Joint Conference on Neural Networks (IJCNN2014)*. IEEE.

Zhi-Hua Zhou and Ming Li. 2005. Semi-supervised regression with co-training. In *IJCAI*, pages 908–916.

Tian Tian Zhu and Man Lan. 2013. Ecnucs: Measuring short text semantic equivalence using multiple similarity measurements. In *Proceedings of the 2nd Joint Conference on Lexical and Computational Semantics*, page 124.