ECNU at SemEval-2018 Task 10: Evaluating Simple but Effective Features on Machine Learning Methods for Semantic Difference Detection

Yunxiao Zhou[1], **Man Lan**[1,2*] **,Yuanbin Wu**[1,2]
[1]Department of Computer Science and Technology,
East China Normal University, Shanghai, P.R.China
[2]Shanghai Key Laboratory of Multidimensional Information Processing
51164500061@stu.ecnu.edu.cn, {mlan, ybwu}@cs.ecnu.edu.cn

Abstract

This paper describes the system we submitted to Task 10 (Capturing Discriminative Attributes) in SemEval 2018. Given a triple (*word*$_1$, *word*$_2$, *attribute*), this task is to predict whether it exemplifies a semantic difference or not. We design and investigate several word embedding features, PMI features and WordNet features together with supervised machine learning methods to address this task. Officially released results show that our system ranks above average.

1 Introduction

The Capturing Discriminative Attributes task (Paperno et al., 2018) in SemEval 2018 is to provide a standard testbed for semantic difference detection, which will benefit many other applications in Natural Language Processing (NLP), such as automatized lexicography and machine translation (Krebs and Paperno, 2016). The goal of this task is to predict whether a word is a discriminative attribute between two concepts. Specifically, given two concepts and an attribute, the task is to predict whether the first concept has this attribute but the second concept does not. For example, given the concepts *apple* and *pineapple*, participants are required to predict whether the attribute *seeds* characterizes the first concept but not the other. In other words, semantic difference detection is a binary classification task: given a triple (*apple*, *pineapple*, *seeds*), the task is to determine whether it exemplifies a semantic difference or not, i.e., *positive* or *negative*. Table 1 shows more data examples.

If word$_1$ has a specific attribute but word$_2$ does not, then the correlation of attribute and word$_1$ should be higher than that of attribute and word$_2$. The semantic similarity is in the same way. In view of the above considerations, to address this task, we explore supervised machine learning methods which use PMI features and WordNet features. In recent years, more and more studies have focused on word embeddings as an alternative to traditional hand-crafted features (Pennington et al., 2014; Tang et al., 2014). Therefore we use word embeddings to obtain the semantic similarity as word embedding features. Besides, we perform a series of experiments to explore the effectiveness of feature types and supervised machine learning algorithms.

word$_1$	word$_2$	attribute	label
apple	pineapple	seeds	positive
candle	chandelier	melts	positive
apple	coconut	brine	negative
apple	cucumber	seeds	negative

Table 1: Examples from the training data.

2 System Description

To perform semantic difference detection of given triples, we adopt supervised learning algorithms with several features that represent semantic similarity and correlation. In the next paragraphs, we will introduce feature engineering and learning algorithms.

2.1 Feature Engineering

In this task, we design three types of features: WordNet features, PMI features and word embedding features.

2.1.1 WordNet Features

WordNet (Miller et al., 1990) is an on-line lexical reference system, which is organized by semantic properties of words. Therefore, the WordNet features are designed to utilize WordNet to obtain the semantic information.

Each word may have a number of different semantics, corresponding to different senses in the

Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018), pages 999–1002
New Orleans, Louisiana, June 5–6, 2018.

WordNet. And the WordNet provides the definitions of the senses for each word. If a word is an attribute of the target word, the attribute word may appear in the sense definition of the target word. For example, a semantic definition of "*snow*" is "*white crystals of frozen water*" and "*white*" is the attribute of "*snow*" which appears in the above definition. Therefore, we design the following features to record the semantic information. Given the triple (*word*$_1$, *word*$_2$, *attribute*), we first load all senses definitions of *word*$_1$, *word*$_2$ and *attribute*. Then we implement four types of binary features: (1) whether *attribute* appears in the senses definitions of *word*$_1$, (2) whether *attribute* appears in the senses definitions of *word*$_2$, (3) whether *word*$_1$ appears in the senses definitions of *attribute* and (4) whether *word*$_2$ appears in the senses definitions of *attribute*. As a result, we get four features.

2.1.2 PMI Features

Pointwise mutual information (PMI) (Church and Hanks, 1990) is a measure of association between two things used in information theory and statistics. And in NLP, this metric can be used to measure the correlation between two words. The higher the PMI, the stronger the correlation between the two words. So we obtain the PMI features of the given triple (*word*$_1$, *word*$_2$, *attribute*). We record the PMI value of *word*$_1$ and *attribute* as well as the PMI value of *word*$_2$ and *attribute* as PMI features. The PMI values we used are calculated using Wikimedia dumps[1] and directly obtained from SEMILAR (Rus et al., 2013). As a result, we get four PMI features.

2.1.3 Word Embedding Features

Word embedding is a continuous-valued vector representation for each word, which usually carries syntactic and semantic information. In this work, we employ two types of word embeddings which are pre-trained word vectors downloaded from Internet with dimensionality of 300: *GoogleW2V* (Mikolov et al., 2013) and *GloVe* (Pennington et al., 2014). The former is pre-trained on News domain, available in Google[2]. And the latter is pre-trained on tweets, available in GloVe[3].

- ***WE_similarity:*** Given the triple (*word*$_1$, *word*$_2$, *attribute*), if *attribute* characterizes *word*$_1$ rather than *word*$_2$, the semantic similarity score of *attribute* and *word*$_1$ should be higher than that of *attribute* and *word*$_2$. After acquiring the vectors of three words in the triple, we calculate the similarity scores of *attribute* and *word*$_1$ as well as *attribute* and *word*$_2$ using *cosine similarity* and *pearson coefficient*. Finally, we got four word embedding similarity features.

- ***WE_operation:*** In addition to the above similarity functions, we also explore two different ways of interaction between word vectors in order to capture the semantic information as much as possible. The operations between three vectors include *concatenation* and *subtraction*. Specifically, given the vectors V_1, V_2 and V_a of the triple (*word*$_1$, *word*$_2$, *attribute*), the concatenation operation is to concatenate three vectors as $[V_1 \oplus V_2 \oplus V_a]$, and the subtraction operation is the element-wise subtraction of *attribute* from *word*$_1$ and *word*$_2$ respectively, i.e., $[V_1 - V_a]$ and $[V_2 - V_a]$. Since we employ two types of word embeddings, finally, we get $3,000$ dimensional vectors as word embedding operation features.

2.2 Learning Algorithm

We grant this task as a binary classification task and explore six supervised machine learning algorithms: Logistic Regression (LR) and Support Vector Machine (SVM) both implemented in Liblinear toolkit (Fan et al., 2008), Stochastic Gradient Descent (SGD), RandomForest and AdaBoost all implemented in scikit-learn tools (Pedregosa et al., 2011), and XGBoost[4] provided in (Chen and Guestrin, 2016). All these algorithms are used with default parameters.

3 Experiments

3.1 Datasets

Table 2 shows the statistics and distributions of training, development, test data sets of this task provided by task organizers.

3.2 Evaluation Metric

To evaluate the system performance, the official evaluation criterion is *macro-averaged F1-score*,

[1] https://dumps.wikimedia.org/enwiki/20170724/
[2] https://code.google.com/archive/p/word2vec
[3] http://nlp.stanford.edu/projects/glove
[4] https://github.com/dmlc/xgboost

Dataset	Positive	Negative	Total
train	6,330 (36%)	11,171 (64%)	17,501
dev	1,364 (50%)	1,358 (50%)	2,722
test	1,293 (55%)	1,047 (45%)	2,340

Table 2: The statistics of data sets in training, development and test data. The numbers in brackets are the percentages of different classes in each data set.

which is calculated among two classes (positive and negative) as follows:

$$F_{macro} = \frac{F_{Pos} + F_{Neg}}{2}$$

3.3 Experiments on Training Data

Firstly, in order to explore the effectiveness of each feature type, we perform a series of experiments. Table 3 lists the comparison of different contributions made by different features on development data with *Logistic Regression* algorithm. We observe the following findings.

(1) The simple PMI features and word embedding similarity features are effective for semantic difference detection and it shows the effectiveness of semantic similarity and correlation for semantic difference detection.

(2) The combination of the first three features not only achieves the best performance for the overall classification but also for each class. These three types of features make contributions to semantic difference detection task. Therefore we use these features in following experiments.

(3) The result of merging the *WE_operation* features is not as good as we expected and the possible reason is that the dimensionality of *WE_operation* features is quite huger than the other three features($3,000$ Vs. 16), which dominates the performance of classification rather than other low dimension features. And the operations of word vectors are too simple to detect the semantic difference.

(4) The WordNet features are not as effective as expected, and the reason maybe that in many cases the attribute words do not appear in the sense definitions of concepts, so we can not get nonzero features.

Secondly, we also explore the performance of different supervised machine learning algorithms. Table 4 lists the comparison of different learning algorithms with *WordNet*, *PMI* and *WE_similarity* features. We find:

Features	$\mathbf{F_{Pos}}$	$\mathbf{F_{Neg}}$	$\mathbf{F_{macro}}$
WordNet	0.683	0.237	0.459
.+PMI	0.653	0.598	0.625 (+0.166)
.+WE_similarity	**0.684**	**0.64**	**0.662** (+0.037)
.+WE_operation	0.561	0.616	0.588 (-0.074)

Table 3: Performance of different features on development data in terms of F1-score. ".+" means to add current features to the previous feature set. The numbers in the brackets are the performance increments compared with the previous results.

(1) LR and SVM achive better results than the other supervised machine learning algorithms and Logistic Regression algorithm achieves the best performance when considering single classification algorithm.

(2) The ensemble of the top 3 machine learning algorithms (LR + SVM + XGBoost) achives higher performance than any single learning algorithm, i.e., 0.663.

Algorithms	$\mathbf{F_{Pos}}$	$\mathbf{F_{Neg}}$	$\mathbf{F_{macro}}$
LR	**0.684**	0.64	0.662
SVM	0.681	0.64	0.661
XGBoost	0.598	0.613	0.606
SGD	0.623	0.559	0.591
AdaBoost	0.631	0.551	0.591
RandomForest	0.565	0.607	0.586
Ensemble	0.683	**0.643**	**0.663**

Table 4: Performance of different learning algorithms on development data in terms of F1-score.

Based on the above results, the system configuration of our final submission is ensemble of LR, SVM and XGBoost algorithms with WordNet, PMI and WE_similarity features. The models are trained on both training and development data sets.

3.4 Results on Test Data

Table 5 shows the results of our system and the top-ranked systems provided by organizers for this semantic difference detection task. Compared with the top ranked systems, there is much room for improvement in our work. There are several possible reasons for this performance lag. First, the features we used are simple. We only record some semantic similarity information and correlations between words. More complex interactions of word vectors could be tried. Second, we only extract features from three words that need to

be classified and have not used some extended resources like the sentences returned from search engines when retrieving these words.

Team ID	$\mathbf{F_{macro}}$
ECNU	0.67 (8)
SUNNYNLP	0.75 (1)
BomJi	0.73 (2)
NTU NLP Lab	0.73 (2)

Table 5: Performance of our system and the top-ranked systems. The numbers in the brackets are the official rankings.

4 Conclusion

In this paper, we extract WordNet features, PMI features and word embedding features from triples and adopt supervised machine learning algorithms to perform semantic difference detection. The system performance ranks above average. In future work, we consider to try more complex interactions of word vectors and use more web resources to capture semantic information.

Acknowledgements

This work is is supported by the Science and Technology Commission of Shanghai Municipality Grant (No. 15ZR1410700) and the open project of Shanghai Key Laboratory of Trustworthy Computing (No.07dz22304201604).

References

Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM.

Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational linguistics*, 16(1):22–29.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. Liblinear: A library for large linear classification. *Journal of machine learning research*, 9(Aug):1871–1874.

Alicia Krebs and Denis Paperno. 2016. Capturing discriminative attributes in a distributional space: Task proposal. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 51–54.

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

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.

Denis Paperno, Alessandro Lenci, and Alicia Krebs. 2018. Semeval-2018 Task 10: Capturing discriminative attributes. In *Proceedings of International Workshop on Semantic Evaluation (SemEval-2018)*, New Orleans, LA, USA.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Vasile Rus, Mihai Lintean, Rajendra Banjade, Nobal Niraula, and Dan Stefanescu. 2013. Semilar: The semantic similarity toolkit. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 163–168.

Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014. Learning sentiment-specific word embedding for twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1555–1565.