Splusplus: A Feature-Rich Two-stage Classifier for Sentiment Analysis of Tweets

Li Dong[†,*] **Furu Wei**[‡], **Yichun Yin**[§,*], **Ming Zhou**[‡], **and Ke Xu**[†]
[†]Beihang University, Beijing, 100191, China
[‡]Microsoft Research, Beijing, 100080, China
[§]Peking University, Beijing, 100871, China
dl@cse.buaa.edu.cn {fuwei,mingzhou}@microsoft.com
yichunyin@pku.edu.cn kexu@nlsde.buaa.edu.cn

Abstract

This paper describes our sentiment classification system submitted to SemEval-2015 Task 10. In the message-level polarity classification subtask, we obtain the highest macro-averaged F1-scores on three out of six testing sets. Specifically, we build a two-stage classifier to predict the sentiment labels for tweets, which enables us to design different features for subjective/objective classification and positive/negative classification. In addition to n-grams, lexicons, word clusters, and twitter-specific features, we develop several deep learning methods to automatically extract features for the message-level sentiment classification task. Moreover, we propose a polarity boosting trick which improves the performance of our system.

1 Introduction

In the task 10 of SemEval-2015, submitted systems are required to categorize tweets to positive, negative, and neutral classes (Rosenthal et al., 2015). There are six testing sets in SemEval-2015. Four of them are tweets: Twitter13, Twitter14, Twitter14Sarcasm, and Twitter15. The TwitterSarcasm14 consists of the tweets which express sarcasm. In order to evaluate the performance on out-of-domain data, the other two datasets are LiveJournal14 and SMS13 that are from web blogs and SMS messages respectively. The details of these datasets are described in (Nakov et al., 2013; Rosenthal et al., 2014; Rosenthal et al., 2015).

We utilize both basic features and deep learning features in our system. Deep learning is used to automatically learn representations, which has achieved some promising results on sentiment analysis (Kim, 2014; Socher et al., 2013; Dong et al., 2014). In order to design more flexible features, we use a two-stage classification framework which conducts subjective/objective (sub/obj) classification and positive/negative (pos/neg) classification. In addition, we introduce a polarity boosting trick that can utilize pos/neg training data to improve classifying tweets to sub/obj. With the help of these features and methods, our system achieves the best results on three out of six datasets among 40 teams in SemEval-2015. We describe the basic features and deep learning features used in our system, and compare their contributions. Moreover, we make the word2vec clustering results on Twitter data publicly available for research purpose.

2 System Description

2.1 Overview

As shown in Figure 1, our sentiment analysis system is a two-stage sentiment classifier which consists of a subjective/objective (sub/obj) classifier and a positive/negative (pos/neg) classifier. By using this artitacture, we can design different feature sets for the two classification steps. Notably, the predicted values of pos/neg classifier is employed to help classify tweets to sub/obj classes. We employ the LIBLINEAR (Fan et al., 2008) with option "*-s 1*" as our classifier. All the input tweets are normalized by replacing the @ mentions and URLs. Moreover, the

[*]Contribution during internship at Microsoft Research.

Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), pages 515–519,
Denver, Colorado, June 4-5, 2015.

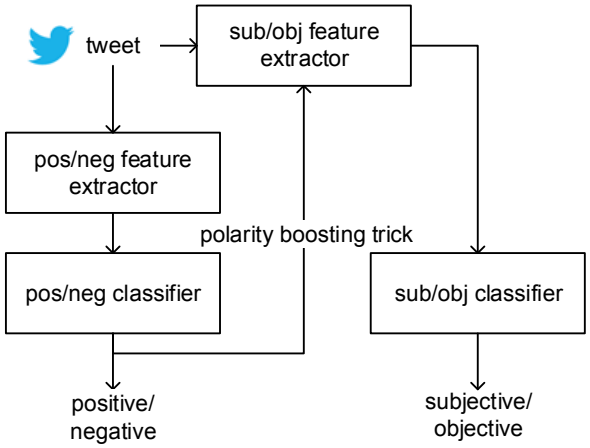

Figure 1: The overview of our two-stage sentiment analysis system. We use two classifiers to predict labels for tweets. Different features are extracted for sub/obj and pos/neg classification steps. The predicted value of pos/neg classifier is used to extract features for sub/obj step, which is called as polarity boosting trick.

elongated words are normalized by shortening them to three contiguous letters.

2.2 Basic Features

We briefly describe the basic features used in our system as follows. The features are used in both pos/neg and sub/obj classifiers unless noted otherwise. The features which appear less than two times are pruned to reduce the model size.

Word ngrams We use unigrams and bigrams for words.

Character ngrams For each word, character ngrams are extracted. We use four-grams and five-grams in our system.

Word skip-grams For all the trigrams and four-grams, one of the words is replaced by $*$ to indicate the presence of non-contiguous words. This feature template is used in sub/obj classification.

Brown cluster ngrams We use Brown clusters[1] to represent words, and extract unigrams and bigrams as features.

POS The presence or absence of part-of-speech tags are used as binary features. We use the CMU ARK Twitter Part-of-Speech Tagger (Owoputi et al., 2013) in our implementation.

Lexicons The NRC Hashtag Sentiment Lexicon and Sentiment140 Lexicon[2] are used. These two lexicons are automatically generated by calculating pointwise mutual information (PMI) scores between the words and positive or negative labels (Kiritchenko et al., 2014). The hashtags and emoticons are used to assign noisy polarity labels for tweets. For both positive and negative lexicons, we extract the following features: (1) the number of occurrences; (2) the maximal PMI score; (3) the score of last term; (4) the total PMI score of terms.

Twitter-specific features The number of hashtags, emoticons, elongated words, and puncuations are used as features.

2.3 Deep Learning Features

In order to automatically extract features, we explore using some deep learning techniques in our system. These features and the basic features described in Section 2.2 are used together to learn classifiers.

Word2vec cluster ngrams We use the word2vec tool (Mikolov et al., 2013) to learn 40-dimensional word embeddings from a twitter dataset. Then, we employ K-means algorithm and L2 distance of word vectors to cluster the $255,657$ words to 4960 classes. The clusters are used to represent words. We extract unigrams and bigrams as features, and use them in sub/obj classifier. The word2vec clustering results are publicly available[3] for research purposes. As shown in Table 1, similar words are clustered into the same clusters. This feature template is used in sub/obj classification.

CNN predicted distribution The convolutional neural networks (dos Santos, 2014) are used to predict the probabilities of three sentiment classes, and the predicted distribution is used as a three-dimension feature template. As illustrated in Figure 2, we use the network architecture proposed by Collobert et al. (2011). The dimension of word vectors is 50, and the window size is 5. Then the concatenated word vectors are fed into a convolutional layer. The vector representation of a sentence is obtained by a max pooling layer, and is used to predict the probabilities of three classes by the softmax layer. We employ stochastic gradient descent to minimize the cross-entropy loss. In order to pre-

[1] http://www.ark.cs.cmu.edu/TweetNLP/clusters/50mpaths2

[2] http://goo.gl/ee2CVo

[3] http://goo.gl/forms/8pLMMClzxB

Cluster	Words
4493	good, hope, great, nice, lovely, special, gr8, enjoying, goood, enjoyed, fabulous, magical, beaut, fab, g8, spectacular, pleasant, spoilt, swell, brill, greaaat, amazin, terrific, kickass, gr9, grrreat, greatt, fabbb, lush, marvellous, frantastic, greeeat, amzing, badasss, greaat, beauitful, pawsome
2123	love, miss, luv, loveee, looove, luh, lovee, misss, ilove, luvvv, lub, wuv, luhhh, luhh, imiss, thnk, loove, loooveee, iove, luuuv, luvv, lovvve, looovvveee, luff, mish, lobe, lovveee, wuvvv, lurv, mith, lovve, love/miss, luuuvvv, lubb, lurve

Table 1: Examples of word2vec clusters. Similar words are clustered to the same cluster.

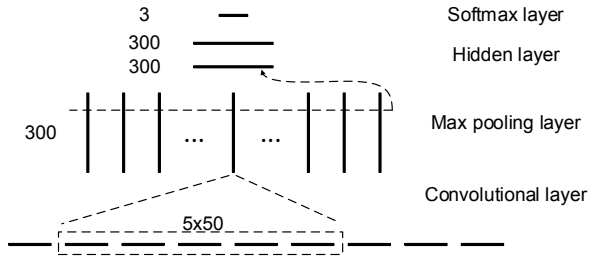

Figure 2: Architecture of convolutional neural network used in our system. The lines represent vectors, and the numbers indicate the vector dimensions.

vent overfitting, a L2-norm constraint for the column vectors of weight matrices is used. The back-propagation algorithm (Rumelhart et al., 1986) is employed to compute the gradients for parameters. The word vectors provided by Tang et al. (2014) are used for initialization.

Sentiment-specific embedding Tang et al. (2014) improve the word2vec model to learn sentiment-specific word embeddings from tweets annotated by emoticons. We use element-wise $\max$, $\min$, and avg operations for the word vectors to extract features.

2.4 Polarity Boosting Trick

Predicted scores indicate the confidence of classifier. If the pos/neg classifier has a high confidence to classify a tweet to positive or negative, it is less likely that this tweet is objective. Consequently, the absolute value of output of pos/neg classifier is used as a feature in sub/obj classification step, which is called as *polarity boosting trick*. This method better utilizes the pos/neg training data to help sub/obj step instead of only using the sub/obj training data. Moreover, this approach is based on the fact that classifying pos/neg is much easier than categorizing sub/obj (Pang and Lee, 2008).

Unlike most of previous work, we perform the pos/neg classification for every message to extract the polarity boosting feature, even if it is classified as an objective message.

3 Experimental Results

The macro-averaged F1-score of positive and negative classes is used as the evaluation metric (Rosenthal et al., 2015). Notably, this evaluation metric also takes the neutral class into consideration. We train the model on TRAIN/DEV (7,072/1,120) provided in SemEval-2013.

3.1 Overall Results

As shown in Table 2, we compare our system with the best results of other teams on six datasets. Our system ranks first on three out of six datasets, namely, Twitter13 (Twt13), Twitter14 (Twt14), and LiveJournal14 (LvJn14). The results indicate that our system performs well for short texts in online social networks. Furthermore, we find that the performance drops for the tweets which are sarcastic. Another model is needed to better address the sarcasm problem in Twitter. In addition, the performance on SMS13 is worse than on Twitter data. This suggests that the mismatch of domains between training data and testing data harms the results.

3.2 Contribution of Features

We conduct ablation experiments on six testing sets to show effectiveness of features. As presented in Table 3, the overall conclusion is that both basic features and deep learning features contribute to the performance. In addition, the polarity boosting trick improves the performance.

Specifically, after removing the ngrams features, our system still performs well, and the results on

Feature	Twt13	Twt14	Twt15	LvJn14	SMS13	Sarc14
all	72.80	74.42	63.73	75.34	67.16	42.86
- basic features	69.80	70.35	59.48	72.74	63.32	47.90
- word/char ngrams & skip-grams	72.70	73.14	62.99	75.43	66.32	44.41
- Brown cluster ngrams	72.03	73.62	63.85	74.75	67.75	42.75
- lexicons	72.48	72.40	62.84	74.78	66.76	44.18
- deep learning features	70.13	70.46	62.23	72.25	66.91	51.47
- word2vec cluster ngrams	72.71	74.14	62.66	74.99	67.11	43.35
- CNN predicted distribution	71.83	70.60	62.81	74.81	68.08	45.87
- sentiment-specific embedding	72.78	74.29	63.69	74.70	67.31	44.10
- polarity boosting trick	72.42	72.20	62.91	75.10	65.74	41.46

Table 3: Results of ablation experiments.

Dataset	Best of Others	Spp (Ours)
Twt13	72.79	**72.80**
Twt14	73.60	**74.42**
Twt15	64.84	63.73
LvJn14	74.52	**75.34**
SMS13	68.37	67.16
Sarc14	59.11	42.86

Table 2: We compare the macro-averaged F1-scores of our system (Spp) with the best results of other teams in SemEval-2015. Our system achieves the highest F1-scores on three out of six datasets.

LvJn14 and Sarc14 become better. Moreover, the automatically learned lexicons play a positive role in our system. We also try some manually annotated lexicons (such as MPQA Lexicon (Wilson et al., 2005), and Bing Liu Lexicon (Hu and Liu, 2004)), but the performance drops on the dev data. It illustrates the coverage of lexicons is important for the informal text data. The cluster features are also useful in this task, because the clusters reduce the feature sparsity and have the ability to deal with out-of-vocabulary words.

The deep learning significantly improves test results on all the datasets except on the sarcastic tweets. Using the clustering results of word2vec performs better and more stable than directly using the vectors as features. This feature template contributes more than other features on Twitter-15 (Twt15). The CNN predicted probabilities also increase the F1-scores. It is the most useful feature template on Twitter-13 (Twt13) and Twitter-14 (Twt14). Additionally, the sentiment-specific embeddings which is learned on emoticon annotated tweets contributes to the performances. It provides more explicit sentiment information than word2vec vectors.

As shown in Table 2, the polarity boosting trick also contributes to the performance of our system on all the six datasets.

4 Conclusions

We describe our message-level sentiment classification system submitted in SemEval-2015. Our system ranks first on three out of six testing sets in the message-level polarity classification task. It employs various basic features and modern deep learning techniques. The deep learning methods help us get rid of feature engineering and improve the results significantly. Furthermore, the polarity boosting trick which is easy to implement is a good way to utilize positive/negative data to improve the subjective/objective classification. There are several interesting directions to further improve the results. First, more recently proposed deep learning models can be used to automatically learn features. Second, we can utilize the noisy data annotated by hashtags or emoticons to learn lexicons of higher quality. Third, making the classifier robust for out-of-domain test data is crucial in practice.

Acknowledgments

We thank Dr. Nan Yang for sharing his K-means clustering code. This research was partly supported by NSFC (Grant No. 61421003).

References

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537, November.

Li Dong, Furu Wei, Ming Zhou, and Ke Xu. 2014. Adaptive multi-compositionality for recursive neural models with applications to sentiment analysis. In *AAAI Conference on Artificial Intelligence*, pages 1537–1543.

Cicero dos Santos. 2014. Think positive: Towards Twitter sentiment analysis from scratch. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 647–651, Dublin, Ireland, August.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. Liblinear: A library for large linear classification. *JMLR*, 9:1871–1874, June.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1746–1751.

Svetlana Kiritchenko, Xiaodan Zhu, and Saif M Mohammad. 2014. Sentiment analysis of short informal texts. *Journal of Artificial Intelligence Research*, pages 723–762.

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.

Preslav Nakov, Sara Rosenthal, Zornitsa Kozareva, Veselin Stoyanov, Alan Ritter, and Theresa Wilson. 2013. Semeval-2013 task 2: Sentiment analysis in Twitter. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 312–320, Atlanta, Georgia, USA, June.

Olutobi Owoputi, Brendan OConnor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *Proceedings of NAACL-HLT*, pages 380–390.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, 2(1-2):1–135.

Sara Rosenthal, Alan Ritter, Preslav Nakov, and Veselin Stoyanov. 2014. Semeval-2014 task 9: Sentiment analysis in Twitter. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 73–80, Dublin, Ireland, August.

Sara Rosenthal, Preslav Nakov, Svetlana Kiritchenko, Saif M Mohammad, Alan Ritter, and Veselin Stoyanov. 2015. Semeval-2015 task 10: Sentiment analysis in Twitter. In *Proceedings of the 9th International Workshop on Semantic Evaluation*, SemEval '2015, Denver, Colorado, June.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, volume 1631, page 1642.

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*, pages 1555–1565.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*, pages 347–354.