The 2018 Conference on Computational Linguistics and Speech Processing
ROCLING 2018, pp. 214-223

An LSTM Approach to Short Text Sentiment Classification with Word Embeddings

Jenq-Haur Wang
Department of Computer Science and Information Engineering
National Taipei University of Technology, Taipei, Taiwan
jhwang@csie.ntut.edu.tw

Ting-Wei Liu
PChome eBay Co., Ltd., Taipei, Taiwan
s5548765@gmail.com

Xiong Luo, Long Wang
School of Computer and Communication Engineering
University of Science and Technology Beijing, Beijing, China
xluo@ustb.edu.cn, lwang@ustb.edu.cn

Abstract

Sentiment classification techniques have been widely used for analyzing user opinions. In conventional supervised learning methods, hand-crafted features are needed, which requires a thorough understanding of the domain. Since social media posts are usually very short, there's a lack of features for effective classification. Thus, word embedding models can be used to learn different word usages in various contexts. To detect the sentiment polarity from short texts, we need to explore deeper semantics of words using deep learning methods. In this paper, we investigate the effects of word embedding and long short-term memory (LSTM) for sentiment classification in social media. First, words in posts are converted into vectors using word embedding models. Then, the word sequence in sentences are input to LSTM to learn the long distance contextual dependency among words. The experimental results showed that deep learning methods can effectively learn the word usage in context of social media given enough training data. The quantity and quality of training data greatly affects the performance. Further investigation is needed to verify the performance in different social media sources.

Keywords: Sentiment Classification, Deep Learning, Long Short-Term Memory, Word2Vec Model.

1. Introduction

Sentiment classification has been used in analyzing user-generated contents for understanding users' intent and opinions in social media. Conventional supervised learning methods have been extensively investigated such as bag-of-words model using TF-IDF, and probabilistic model using Naïve Bayes, which usually need hand-crafted features. For social media content which are very short and diverse in topic, it's difficult to obtain useful features for classification. Thus, a more effective method for short text sentiment classification is needed.

Deep learning methods have gradually shown good performance in many applications, such as speech recognition, pattern recognition, and data classification. These methods try to learn data representation using a deeper hierarchy of structures in neural networks. Complicated concepts are possible to learn based on simpler ones. Among deep feedforward networks, Convolutional Neural Networks (CNNs) have been shown to learn local features from words or phrases [1], while Recurrent Neural Networks (RNNs) are able to learn temporal dependencies in sequential data [2]. Given the very short texts in social media, there's a lack of features. To obtain more useful features, we further utilize the idea of distributed representation of words where each input is represented by many features and each feature is involved in many possible inputs. Specifically, we use the Word2Vec word embedding model [3] for distributed representation of social posts.

In this paper, we want to investigate the effectiveness of long short-term memory (LSTM) [4] for sentiment classification of short texts with distributed representation in social media. First, a word embedding model based on Word2Vec is used to represent words in short texts as vectors. Second, LSTM is used for learning long-distance dependency between word sequence in short texts. The final output from the last point of time is used as the prediction result. In our experiments of sentiment classification on several social datasets, we compared the performance of LSTM with Naïve Bayes (NB) and Extreme Learning Machine (ELM). As the experimental results show, our proposed method can achieve better performance than conventional probabilistic model and neural networks with more training data. This shows the potential of using deep learning methods for sentiment analysis. Further investigation is needed to verify the effectiveness of the proposed approach in larger scale. The remainder of this paper is organized as follows: Sec. 2 lists the related works, and Sec. 3 describes the proposed method. The experimental results are described in Sec. 4. And some discussions of the results are summarized in Sec.5. Finally, Sec. 6 lists the conclusions.

2. Related Work

Artificial neural network is a network structure inspired by neurons in human brains. Nodes are organized into layers, and nodes in adjacent layers are connected by edges. Computations are done in a feed-forward manner, and errors can be back-propagated to previous layers to adjust the weights of corresponding edges. Extreme Learning Machines (ELMs) [5] are a special type of neural networks in which the weights are not adjusted by back propagation. The hidden nodes are randomly assigned and never updated. Thus, the weights are usually learned in one single step, which is usually much faster.

For more complex relations, deep learning methods are adopted, which utilize multiple hidden layers. With deeper network structures, it usually takes more computing time. These methods were made feasible thanks to the recent advances of computing powers in hardware, and the GPU processing in software technologies. Depending on the different ways of structuring multiple layers, several types of deep neural networks were proposed, where CNNs and RNNs are among the most popular ones. CNNs are usually used in computer vision since convolution operations can be naturally applied in edge detection and image sharpening. They are also useful in calculating weighted moving averages, and calculating impulse response from signals. RNNs are a type of neural networks where the inputs of hidden layers in the current point of time depend on the previous outputs of hidden layer. This makes them possible to deal with a time sequence with temporal relations such as speech recognition. According to previous comparative study of RNN and CNN in natural language processing [6], RNNs are found to be more effective in sentiment analysis than CNNs. Thus, we focus on RNNs in this paper.

As the time sequence grows in RNNs, it's possible for weights to grow beyond control or to vanish. To deal with the vanishing gradient problem [7] in training conventional RNNs, Long Short-Term Memory (LSTM) [4] was proposed to learn long-term dependency among longer time period. In addition to input and output gates, forget gates are added in LSTM. They are often used for time series prediction, and hand-writing recognition. In this paper, we utilize LSTM in learning sentiment classifiers of short texts.

For natural language processing, it's useful to analyze the distributional relations of word occurrences in documents. The simplest way is to use one-hot encoding to represent the occurrence of each word in documents as a binary vector. In distributional semantics, word embedding models are used to map from the one-hot vector space to a continuous vector

space in a much lower dimension than conventional bag-of-words model. Among various word embedding models, the most popular ones are distributed representation of words such as Word2Vec [3] and GloVe [8], where neural networks are used to train the occurrence relations between words and documents in the contexts of training data. In this paper, we adopt the Word2Vec word embedding model to represent words in short texts. Then, LSTM classifiers are trained to capture the long-term dependency among words in short texts. The sentiment of each text can then be classified as positive or negative.

3. The Proposed Method

In this paper, the overall architecture include three major components: pre-processing & feature extraction, word embedding, and LSTM classification, as shown in Fig. 1.

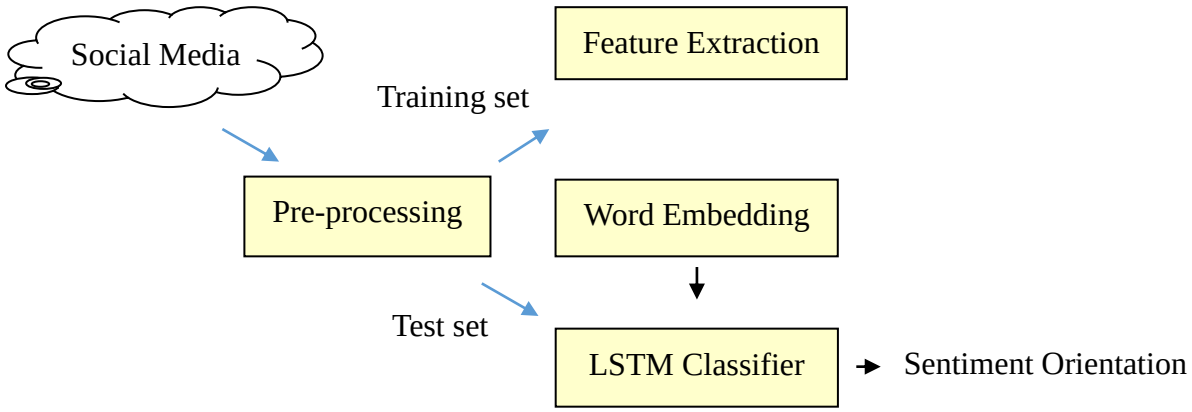

Figure 1. The system architecture of the proposed approach

As shown in Fig.1, short texts are first preprocessed and word features are extracted. Second, the Word2Vec word embedding model [3] is used to learn word representations as vectors. Third, LSTM [4] is adopted for sequence prediction among words in a sentence. The details of each component are described in the following subsections.

3.1. Preprocessing and Feature Extraction

First, short texts are collected with custom-made crawlers for different social media. Then, preprocessing tasks are needed to cleanup post contents. For example, URL links, hashtags, and emoticons are filtered. Also, stopword removal is performed to focus on content words. For Chinese posts, we used Jieba for word segmentation. Then, we extract metadata such as poster ID, posting time, and the number of retweets and likes. These will be used as additional features for classification.

3.2. Word Embedding

In bag-of-words model, it's very high dimensional, and there's a lack of contextual relations between words. To better represent the limited content in short texts, we use Word2Vec word embedding model [3] to learn the contextual relations among words in training data. Also, the fixed number of dimensions in word embedding model can facilitate more efficient computations. There are two general models in Word2Vec, Continuous Bag-of-Words (CBOW) and Skip-gram. Since much better performance for skip-gram model in semantic analysis can be obtained [3], we use word vectors trained via Word2Vec Skip-gram model as the inputs to the following stage of classification.

3.3. Long Short-Term Memory (LSTM)

After representing each word by its corresponding vector trained by Word2Vec model, the sequence of words $\{T_1, \ldots, T_n\}$ are input to LSTM one by one in a sequence, as shown in Fig.2.

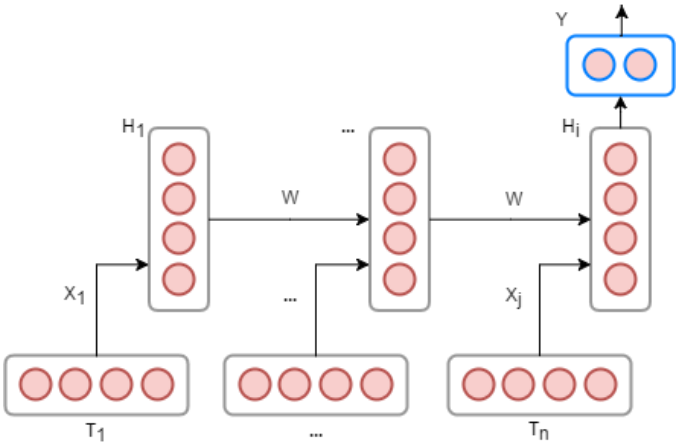

Figure 2. The idea of LSTM

In Fig.2, each term T_i is first converted to the corresponding input vector x_i using Word2Vec model and input into LSTM one by one. At each time j, the output W of the hidden layer H_j will be propagated back to the hidden layer together with the next input x_{j+1} at the next point of time $j+1$. Finally, the last output W_n will be fed to the output layer.

To comply with the sequential input of LSTM, we first convert posts into three-dimensional matrix M(X, Y, Z), where X is the dimension of Word2Vec word embedding model, Y is the number of words in the post, and Z is the number of posts. To avoid the very long training time, we adopt a single hidden-layer neural network. The number of neurons in input layer is

the same as the dimension of Word2Vec model, and the number of neurons in output layer is the number of classes, which is 2 in this case. By gradient-based back propagation through time, we can adjust the weight of edges in hidden layer at each point of time. After several epochs of training, we can obtain the sentiment classification model.

3.4. Sentiment Classification

After sentiment classification model is trained using LSTM, all posts in test set are preprocessed with the same procedure as the training set, and represented using the same word embedding model. Then, for testing step, the same processes of LSTM in Fig. 2 are followed, except for the weights update. The output of LSTM model will then be evaluated with the labels of each post in test data in the experiments.

4. Experiments

In order to evaluate the performance of our proposed approach, we conducted three different experiments. First, we used English movie reviews from IMDB. Second, we tested Chinese movie review comments from Douban. Finally, we evaluated the performance for Chinese posts in the PTT discussion forum.

In the first dataset, there are 50,000 review comments in IMDB Large Movie Review Dataset [9], where 25,000 were used as training and 25,000 as test data. To avoid influence from previous comments of the same movie, we collected no more than 30 reviews for each movie. The rating of each movie was used as the ground truth, where a rating above 7 as positive, and a rating below 4 as negative.

In the second dataset, we collected top 200 movies for each of the 10 categories in Douban Movies. The top 40 to 60 review comments were extracted from each movie according to their popularity. The ground truth of this dataset was set as follows: a rating of 1-2 as negative, and a rating of 4-5 as positive. The comments with a rating of 3 or no ratings are ignored. After removing these comments, there are 12,000 comments where 6,000 were used as training and 6,000 as test data.

In the third dataset for the most popular social media platform PTT in Taiwan, we collected 3,500 posts during Aug. 31 and Sep. 1, 2015 and the corresponding 34,488 comments as the training data, and 1,000 posts during Sep. 2 and Sep. 3, 2015 and the 6,825 comments as the test data. The user ratings of like/dislike are used as the ground truth of this dataset.

To evaluate the classification performance, standard evaluation metrics of precision, recall, F-measure, and accuracy were used to compare three classifiers: Naïve Bayes (NB), Extreme Learning Machine (ELM), and Long Short-Term Memory (LSTM). Word2Vec model is applied for all three classifiers.

For the first dataset of IMDB movie reviews, the performance comparison among three classifiers are shown in Fig.3.

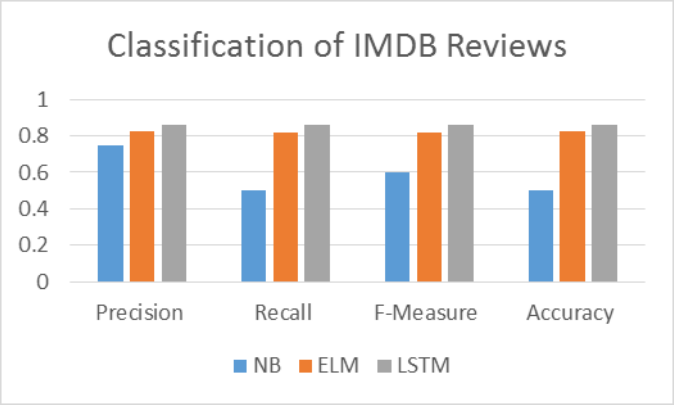

Figure 3. The performance comparison of three classifiers for IMDB reviews

As shown in Fig.3, the best performance can be achieved for LSTM with a F-measure of 0.859. We can see the consistently better performance for LSTM than NB and ELM. Naïve Bayes is the worst due to its high false positive rates. This shows the better performance for neural network methods, especially for deep learning methods. Next, the performance for more casual comments in Douban Movies is shown in Fig.4.

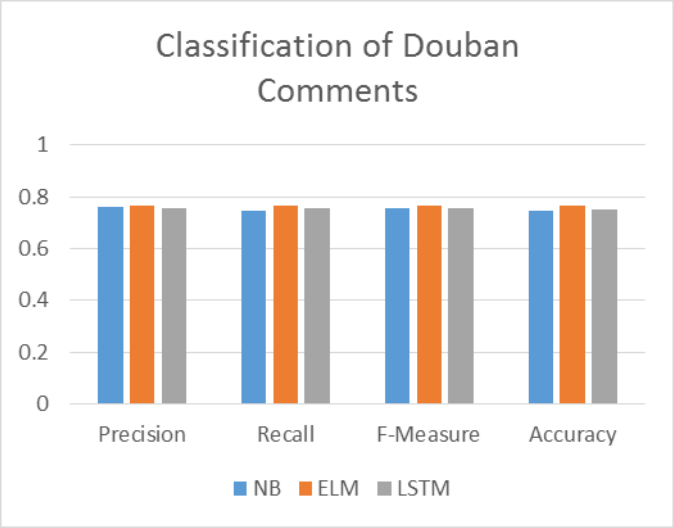

Figure 4. The performance comparison of classification for Douban comments

As shown in Fig.4, the performance of all three classifiers are comparable with slight differences. The best performance can be achieved for ELM with a F-measure of 0.765, while LSTM obtained a comparable F-measure of 0.754. Since the comments are less formal and shorter in lengths, they are more difficult to classify than longer reviews in IMDB. To further evaluate the effects of training data size on the performance, we include more training data from 6,000 reviews to 10,000 and 20,000 in the next experiment. The test data size remains unchanged. The results are shown in Fig.5.

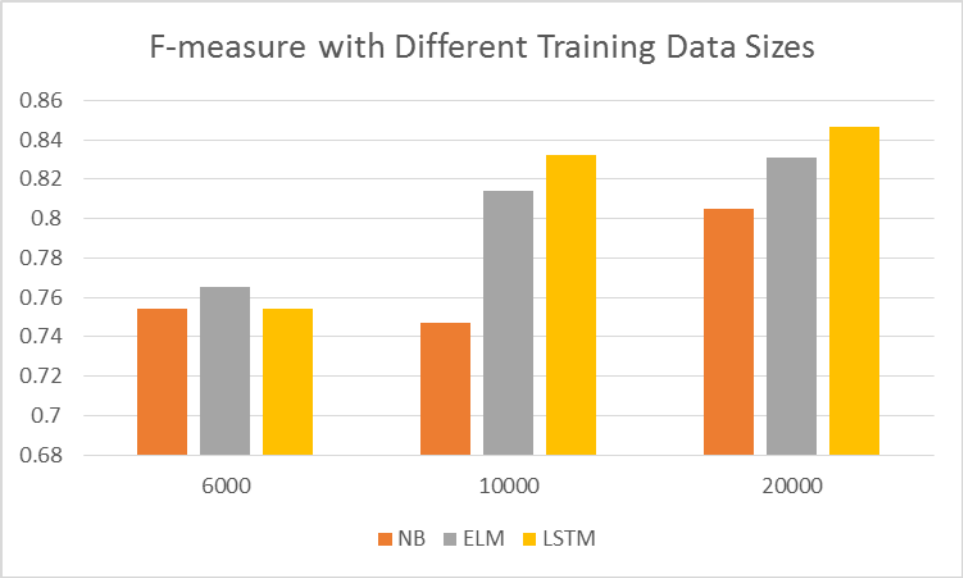

Figure 5. The effects of training data size on classification performance

As shown in Fig.5, as the training data size grows, the classification performance improves for all classifiers except for Naïve Bayes at 10,000. The best performance can be achieved for LSTM with a F-measure of 0.847 when training data size reaches 20,000. Next, the performance of three classifiers on PTT posts are shown in Fig.6.

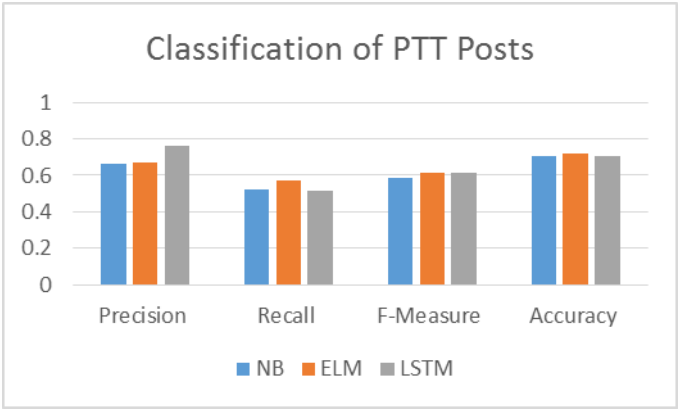

Figure 6. The performance comparison of classification for PTT posts

As shown in Fig.6, the best performance can be achieved for ELM with a F-measure of 0.615, and LSTM with a comparable F-measure of 0.613. LSTM got a higher precision but lower recall for negative posts. The reason is due to the mismatches between user ratings and the sentiment polarity of the corresponding user comments. Users often marks their ratings as "likes" for the posts they agree with, but express their strong negative opinions in their comments. This disagreement is more common in the online forum PTT due to the special characteristics in the community. The impact of this special behavior is larger for LSTM than the other two classifiers.

5. Discussions

From the experimental results, some observations about the proposed approach are shown as follows. First, using word embedding models, the sentiments of short texts can be effectively classified. Second, depending on different types of social media, the performance might vary. The classification performance is better for movie reviews than casual comments and posts in online forums. But the performance of LSTM is still comparable to ELM and NB. This shows the feasibility of an LSTM-based approach to short-text sentiment classification. Third, data size can also affect the classification performance. More training data can lead to better performance. Finally, special characteristics in certain online forums might lead to inferior classification performance. This behavior mismatch between user opinions in comments and user ratings reflects the sarcastic language used among the community in PTT online forum. To deal with this special characteristic, we need more training data to train "community"-specific sentiment lexicon to reflect the online behaviors of social media community.

6. Conclusion

In this paper, we have proposed a sentiment classification approach based on LSTM for short texts in social media. Using word embeddings such as Word2Vec model, it's feasible to train the contextual semantics of words in short texts. Also, deep learning methods such as LSTM show better performance of sentiment classification when there are more amounts of training data. For special community behaviors, further experiments using "community"-specific sentiment lexicon and larger data sizes are needed in future.

References

[1] Y. Kim, “Convolutional Neural Networks for Sentence Classification,” *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pp. 1746–1751, 2014.

[2] J. L. Elman, “Finding Structure in Time,” *Cognitive Science*, Vol. 14, No.2, pp. 179-211, 1990.

[3] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient Estimation of Word Representations in Vector Space,” *Proceedings of the International Conference on Learning Representations 2013 Workshop*.

[4] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Journal of Neural Computation*, Vol. 9 No. 8, pp. 1735-1780, 1997.

[5] G. B. Huang, Q. Y. Zhu, and C. K. Siew, "Extreme Learning Machine: Theory and Applications," *Neurocomputing*, 70 (1): 489–501, 2006.

[6] W. Yin, K. Kann, M. Yu, and H. Schütze, “Comparative Study of CNN and RNN for Natural Language Processing,” *Computing Research Repository (CoRR)*, vol. abs/1702.01923, 2017.

[7] S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber, “Gradient Flow in Recurrent Nets: the Difficulty of Learning Long-Term Dependencies,” In S. C. Kremer and J. F. Kolen, editors, *A Field Guide to Dynamical Recurrent Neural Networks*. IEEE Press, 2001.

[8] J. Pennington, R. Socher, and C. D. Manning, “GloVe: Global Vectors for Word Representation,” *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pp. 1532–1543, 2014.

[9] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning Word Vectors for Sentiment Analysis," *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (HLT 2011)*, pp. 142-150, 2011.