Sentiment Classification Using Word Embedding

Yongjie Kang

Abstract

We present a deep neural network on top of word embedding for sentiment classification task. The neural network contains convolutional layer and LSTM layer which are used to extract context and sequence information from texts. We show that this multiple layers neural network model outperforms logistic regression based bag-of-words models and almost same as the benchmark created by linear SVM[1] . Further discuss also indicates that it can be easily extended to become a more robust model.

1. Introduction

Sentiment analysis aims to extract affective states and subjective information from texts [1] and is a well-known NLP problem. The big raise of social media such as Facebook and Twitter brings tons of text data which can be used for analyzing sentiment. There are many state-of-the-art algorithms available, for example, logistic regression, support vector machine. and they do outperform human beings on text categorization when the size of data is large. However, these models fail in considering the order and semantics of words[7] and are diffuclt to be extented since the whole input has to be recomputed . But natural language is complex and context and semantics matter. Word embedding is an available approach that cares both cases above.

Word embedding has been successful in many NLT tasks. It encodes sequences of texts into word vectors with real number values, and the closeness of the meanings between two words can be represented as the similarity between two word vectors, for example, cosine similarity is one of the similarity measurements. In such way, the value vector is determined by context and can be seen as the meaning of the word in higher dimension space.

In this report, we present a deep neural network sentiment classifier which takes the word vectors as the input. The model takes the context of text into account and has produced remarkable results.

We use the IMDB movie reviews dataset as the training data. The comparison between the performance of our model and other bag-of-words based classification models indicates that our model is well performed and robust. In the end, we also discuss the opportunities of improving the model to obtain a better performance.

2. Related Works

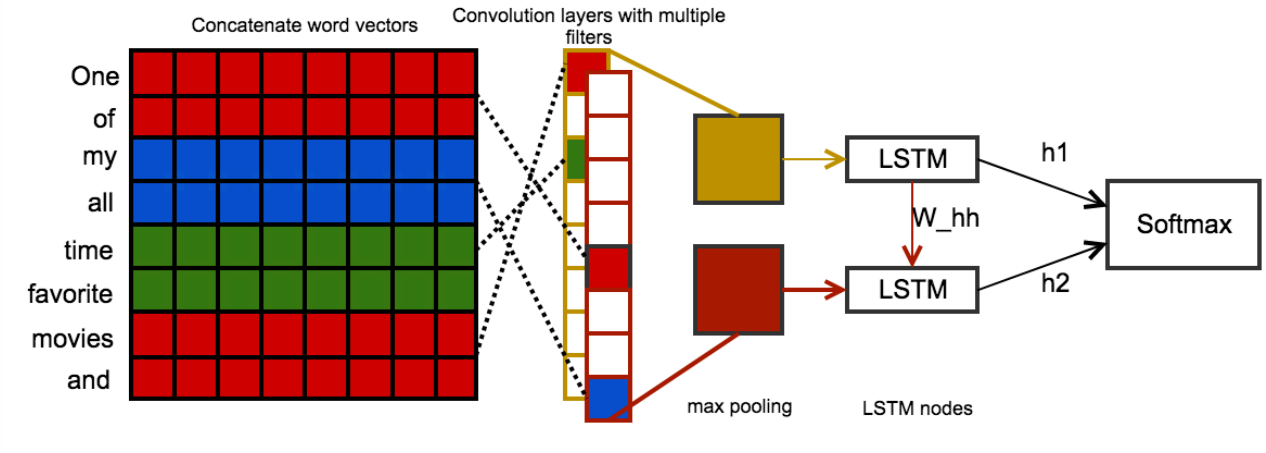
A broad overview of existing related works, the range varies from phrase-level sentiment analysis to document-level sentiment analysis. On document level sentiment analysis, the authors of (Bo Pang, 2008) [1] demonstrated different machine learning techniques and the background knowledge of each algorithm. It's a general view of all state-of-the-art approaches dealing with sentiment and opinion analysis. The results are used as benchmarks of this report. Regarding feature engineering, in (Efthymios Kouloumpis, 2011), they considered features including n-gram features, part-of-speech features and lexicon features to train the models. Although they made a conclusion that the part-of-speech features are not useful in microblogging domain, but considering them in movie reviews domain could be an interesting trial. Pre-processing has an significant role in opinion mining task for the specific domain of reviews (Fernando Leandro dos Santos, 2004). In (Ryan Coughlin, 2017) Labeled Latent Dirichlet Allocation was applied to recognize the topic words (keywords). In this way, those most import words would have higher weights. the authors of (Changhua Yang, 2007) applied SVM and CRF classifiers on web blog opinion classification, they first categorized sentiment on sentence level and took the whole document into account by combining the results of sentence level analysis. They also concluded that considering the last sentence of a document would generate the best classifier.

3. Approach

Our model is a 5-layer deep neural network which contains input layer, convolution layer, max pooling layer, LSTM layer and output layer. It encodes words into dense vectors and concatenates word vectors to generate document-level vector space. Then it uses the concatenated vectors to train the sentiment classifier. Keras provides convenient APIs to generate deep neural networks. "Embedding" layer is the first layer used for converting texts to word vectors. In optimization process, the model minimizes cross-entropy defined as follow:

$$C = y_{true} ln(y) + (1 - y_{true}) ln(1 - y)$$

The architecture of our model is abstracted as the diagram below:



The initialization step includes fixing the length of each review and randomly initializing the values of each word vector. The values of word vector will be updated using backpropgataion after each iteration.

At the training step, firstly, word vectors are mapped to convolution layer using convolutional kernel

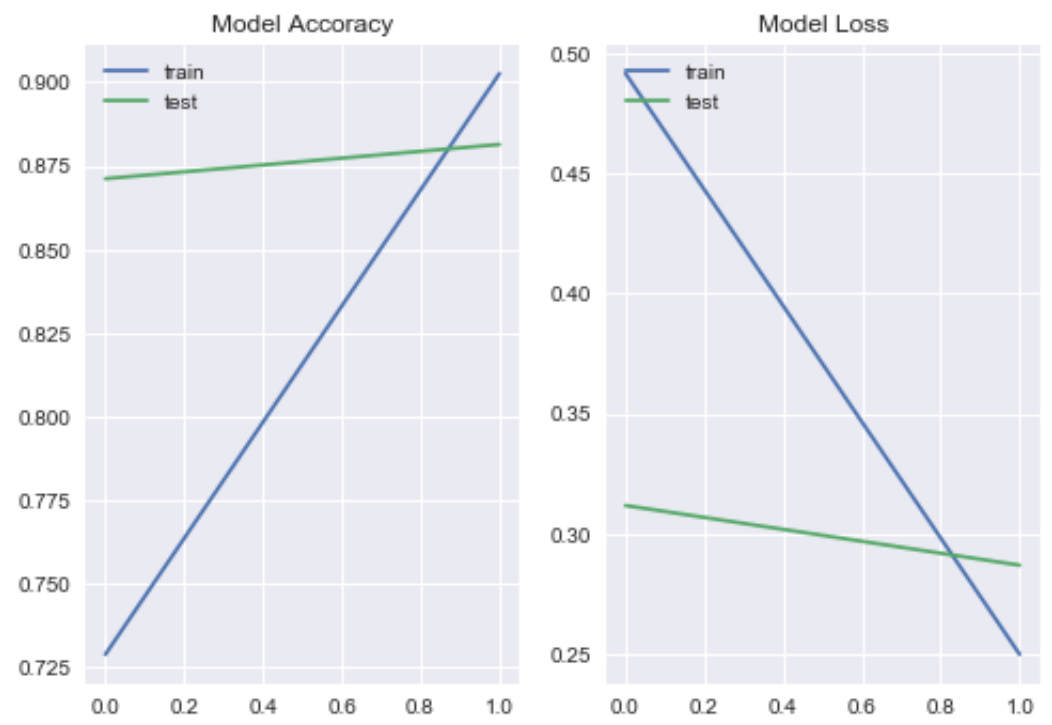
$$X_i = f(W x_{i:i+h} + b), [6]$$

where x is a window of words. Then the max pooling process creates an abstracted form of representation from the output of the previous step. After that, we apply a LSTM network to extract the sequence information from outputs of layers below. Finally, since it's binary classification model (the label only contains positive and negative), we simple use logistic regression to predict the sentiment label.

4. Experiment

In experiment, we both implement linear SVM presented in (Bo Pang, 2008) and logistic regression as benchmarks. The data are collected and served in *Large Moive Reivew Dataset*. There are 25000 movie reviews including 12500 positive reviews and 12500 negative reviews. We use the prediction accuracy of test data as the measurement of model performance. Since reviews do not have to be same length, in order to feed our model with same dimension of input, we truncate the long sentences and add padding to the short ones. The average length of all sentences is used as the dimension of sentence space. The number of filters in convolution layer is 32, which is selected from grid search and cross validation. Same process for selecting number of unites in LSTM node.

During the training process, batch size and epoch is determined by cross validation accuracy. If the epoch value is too large, it will cause overfitting. After parameter tuning, we get our best model and the changes of model accuracy and loss with the increasing of epoch are shown as follow:



The outputs are shown as table below.

Models	Training Accuracy	Cross-Validation Accuracy
Linear SVM	0.93792	0.88348

Logistic Regression	0.93292	0.88292
Word Embedding NN	**0.9054**	**0.8872**

Results of our model against both models including linear SVM presented in (Bo Pang, 2008). We had expected better performance since the model is much complex and considers the context of texts instead of just takes the term-document (weighed) matrix as input directly. But our model is more flexible and there are opportunities for further improvement, such as adding a dropout layer to prevent model from overfitting, using pre-trained word vectors (google word2vec model)[7] rather than updating vectors after each iteration, and even generating a deeper convolution network.

5. Conclusion

In the report we have described details about training deep multiple layers neural network and compared the cross-validation accuracy with the accuracies generated from two different bag-of-words based models, The results indicate that our model is well performed and can be easily extended to become a more robust model. The experiment also demonstrates that word embedding is able to learn context from texts and assign meaningful vector to each word. Furthermore, word embedding base model avoids generating term-document matrix every time and is highly efficient in online training process. We believe our word embedding based neural network will be imporved to solve more complex NLP tasks.

6. Reference

[1] Pang, B., and Lee, L. 2008. Opinion mining and sentiment analysis. Foundations and Trends inInformation Retrieval 2(1-2):1–135.

[2] E. Haddi, X.Liu, and Y. Shi, ‘‘The role of text pre-processing in sentiment analysis,’’Procedia Comput. Sci., vol. 17, pp. 26–32, Sep. 2014.

[3] ZHAO JIANQIANGand GUI XIAOLIN, “Comparison Research on Text Pre-processing Methods on TwitterSentiment Analysis” IEEE., vol. 5, Feb. 2017.

[4] Kouloumpis, E., Wilson, T., & Moore, J. D.(2011). Twitter sentiment analysis: The good the bad and the omg!. Icwsm,11(538-541), 164.

[5] Yang, Changhua, Kevin Hsin-Yih Lin, and Hsin-Hsi Chen. "Emotion classification using web blog corpora." In Web Intelligence, IEEE/WIC/ACM International Conference on, pp. 275-278. IEEE, 2007.

[6] Kim, Yoon. "Convolutional neural networks for sentence classification." arXiv preprint arXiv:1408.5882 (2014).

[7] Le, Quoc, and Tomas Mikolov. "Distributed representations of sentences and documents." In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pp. 1188-1196. 2014.

[8] Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. "Efficient estimation of word representations in vector space." *arXiv preprint arXiv:1301.3781* (2013).

[9] Maas, Andrew L., Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. "Learning word vectors for sentiment analysis." In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pp. 142-150. Association for Computational Linguistics, 2011.