XJSA at SemEval-2017 Task 4：A Deep System for Sentiment Classification in Twitter

Yazhou Hao
University of Xi'an Jiaotong
yazhouhao@gmail.com

Yangyang Lan
University of Xi'an Jiaotong
1564018606@qq.com

Yufei Li
University of Xi'an Jiaotong
18391819285@163.com

Chen Li
University of Xi'an Jiaotong
cli@xjtu.edu.cn

Abstract

This paper describes the XJSA System submission from XJTU. Our system was created for SemEval2017 Task 4 – subtask A which is very popular and fundamental. The system is based on convolutional neural network and word embedding. We used two pre-trained word vectors and adopt a dynamic strategy for k-max pooling.

1 Introduction

Several years ago, the typical approaches to sentiment analysis of tweets were based on classifiers trained using several hand-crafted features, in particular lexicons of words with an assigned polarity value. About since 2014 the deep neural network methods have got state-of-the-art results in many NLP tasks, especially in sentiment classification. The work of Harvard NLP group in 2014 and Kalchbrenner's work in 2014 have suggested that convolutional neural network and word embedding play important roles in this field. General word embedding has got excellent results. If we can embed sentiment information in vectors, we will get better results. There are some open word vectors on the web already such as Word2Vec (Mikolov et al., 2013), Glove (Pennington et al., 2014), SSWE (Tang et al., 2014). In our system we use Word2Vec and SSWE at the same time.

Deep learning models have achieved excellent results in computer vision and speech recognition in recent years. In the field of natural language processing, much work with deep learning methods has involved learning word vectors representations for their own task or problem (Bengio et al., 2003; Mikolov et al., 2013, Collobert C&W et al., 2011).The others exploit the open word vectors which was mentioned above.Word vectors is a transformation of the feature of letter,word,sentence and paragraph or even text. It's a lower dimensional, dense and continuous vectors. In this vector, the words have similar syntactic are close – in Euclidean or cosine distance in the vector space. So one can study and compare the syntactic functionality between different words via word vectors.

Convolutional neural network (CNN) utilize layers with convolutional filters that are applied to local features (LeCun et al., 1998). CNN originally invented for computer vision, recently CNN models have achieved remarkably results in many natural language processing problem, such as sentence modeling (Kalchbrenner et al., 2014), semantic parsing (Yih et al., 2014), sentiment classification (kim et al., 2014) and other traditional natural language processing tasks(Collobert C&W et al., 2011).

Our system was inspired by the work (kim et al., 2014) and another work (Tang et al., 2014). In the aspect of CNN, we use a simple 3 layers CNN to automatic extract features. In the aspect of pre-trained vectors, we use the Word2Vec and SSWE to filter our training set to get a proper input for CNN. The reason that we use the vectors trained by Mikolov et al. (2013) is the 100 billion words of Google News and the vectors are publicly for free. We use the SSWE vectors because the vectors was especially trained for sentiment classification by Tang et al (2014). SSWE contains sentiment information which is not in word vectors trained by Mikolov.

Proceedings of the 11th International Workshop on Semantic Evaluations (SemEval-2017), pages 728–731,
Vancouver, Canada, August 3 - 4, 2017.

2 Background

As is shown above,traditional methods typically model the syntactic context of words but ignore the sentiment information of text. As a result, words with opposite polarity are mapped into close vectors, such as good and bad, just as Word2Vec.So in our system,we use SSWE and Word2Vec at the same time for word embedding, SSWE first.

Tang et al.(2014) introduce SSWE model to learn word embedding for Twitter sentiment classification. In our task,We use the word vector trained by $SSWE_u$,which captures the sentiment information of sentences as well as the syntactic contexts of words. $SSWE_u$ is illustrated in Figure1.

Given an original(or corrupted)n-gram and the sentiment polarity of a sentence as the input, $SSWE_u$ predicts a two-dimensional vector for each input n-gram. t is the original n-gram, t^r is the corrupted n-gram.The two scalars (f_0^u, f_1^u) stand for language model score and sentiment score of the input n-gram,where f_0^u stands the positive, f_1^u the negative.

The training goal of SSWE are that (1) the original n-gram should obtain a higher language model score $f_0^u(t)$ than the corrupted n-gram $f_0^u(t^r)$,and (2) the sentiment score of original n-gram $f_1^u(t)$ should be more consistent with the gold polarity annotation of sentence than corrupted n-gram $f_1^u(t^r)$ .The loss function of $SSWE_u$ is shown behind,

$$loss_u(t,t^r) = \alpha \bullet loss_{cw}(t,t^r) + (1-\alpha) \bullet loss_{us}(t,t^r) \quad (1)$$

where $loss_{cw}(t,t^r)$ is the syntactic loss as given in Equation 1, $loss_{us}(t,t^r)$ is the sentiment loss as shown in Equation 2.The hyper-parameter α weights the two parts. $\delta_s(t)$ is an indicator function reflecting the sentiment polarity of a sentence.

$$loss_{us}(t,t^r) = \max(0,1-\delta_s(t)f_1^u(t) + \delta_s(t)f_1^u(t^r)) \quad (2)$$

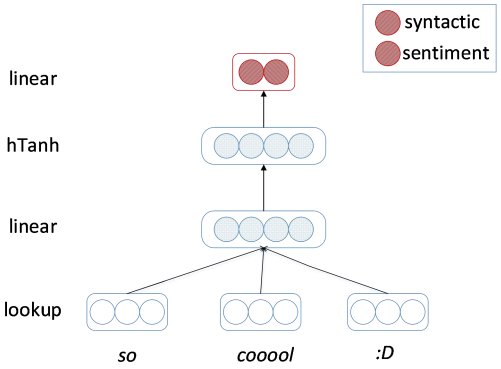

Figure 1:The $SSWE_u$ model

3 Model

The architecture of our system shown in figure 2 is a simple 3 layers CNN just like the architecture of Kim et al (2013). $x_i \in \Re^k$ is the *k*-dimensional word vector corresponding to the *i*-th word in the sentence. A sentence of length *n* is described as

$$x_{1:n} = x_1 \oplus x_2 \oplus ... \oplus x_n, \quad (3)$$

Where $\oplus$ is the concatenation operator. Then we let $x_{i:i+j}$ stand for the concatenation of words $x_i, x_{i+1}, ..., x_{i+j}$. A convolutional filter $w \in \Re^{hk}$ is applied to a window of h words to produce a new feature. For example, a feature c_i is generated from a window of words $x_{i:i+h-1}$ by

$$c_i = f(w \bullet x_{i:i+h=1} + b) \quad (4)$$

where $b \in \Re$ is a bias term and f is a non-linear function. This filter is applied to each possible window of words in the sentence $\{x_{1:h}, x_{2:h+1}, ..., x_{n-h+1:n}\}$ to produce a feature map

$$c = [c_1, c_2, ..., c_{n=h+1}], \quad (5)$$

Here $c \in \Re^{n-h+1}$. Then a max-over-time pooling operation is applied just like Collobert C&W et al. (2011).over the feature map and take the maximum value $\hat{c} = \max\{c\}$

4 Experimental Setup

We test our system on the following settings:

4.1 Hyper-parameters and Training

There are four models in Kim et al (2013):CNN-rand,CNN-static,CNN-non-static,CNN-multichannel.

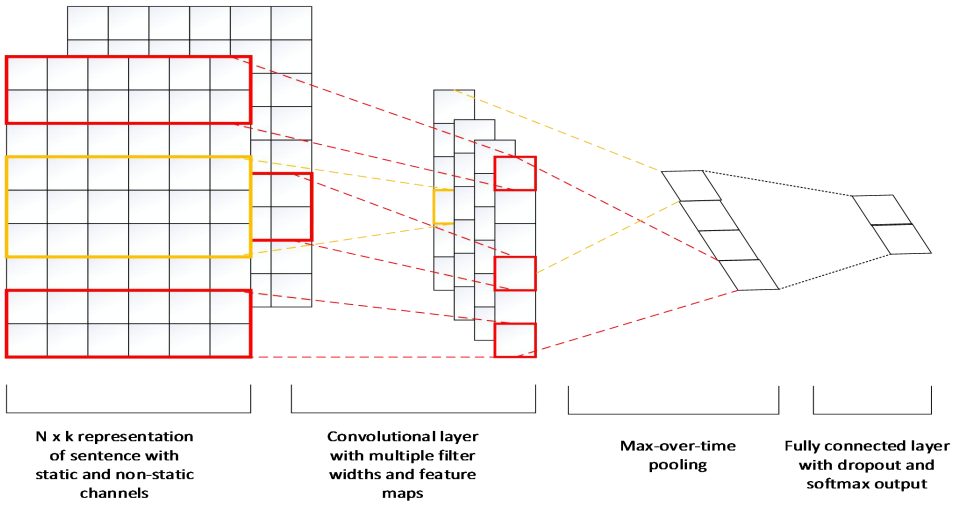

Figure 2: Architecture of XJSA

For all our experiments we use CNN-non-static: A model with pre-trained vectors from SSWE and wrd2vec.The pre-trained vectors are fine-tuned for each task.

We use rectified linear units, filter windows (h) of 3, 4, 5 with 100 feature maps each, dropout rate (p) of 0.5, l_2 constraint (s) of 3, and mini-batch size of 50. These values were chosen via a grid search on the dev sets.

4.2 Pre-trained Word Vectors

Initializing word vectors with those obtained from an unsupervised neural language model is a popular method to improve performance in the absence of a large supervised training set (Collobert C&W et al., 2011; Socher et al., 2011; Iyyer et al., 2014). First we use SSWE (Tang et al. 2014) which is a word vector contains sentiment information.We also use the publicly available word2vec vectors which were trained on 100 billion words from Google News. The vectors have dimensionality of 300 and were trained using the CBOW architecture.Because the dimensionality of vectors in SWEE is 50, so we extended it to 300 dimension by padding the 250 dimension randomly.

4.3 Environment of experiment

The experiments were run on a linux server with an nVIDIA GTX 1080 accelerated GPU.

5 Results

In order to compare the results of our system with other better system's results, here we show enough results generated by our system and the top one. The official submission achieved results presented in Table 1, compared to the top scoring system. We also list our detailed scores in Table 2

	P	**R**	**F1**
Positive	0.5791	0.6748	0.5423
Negative	0.5655	0.5592	0.5065
Neutral	0.5264	0.4340	0.4962
AvgP = 0.557, **AvgR** = 0.556, **AvgF1**= 0.519			
Overall score: 0.556			

Table 1: Official results of our submission compared to the top one.

system	**AvegF1**	**AvegR**	**Acc**
BB_twtr	0.685	0.681	0.658
XJSA	0.519	0.556	0.575

Table 2: Detailed scores of XJSA official submission.

6 Conclusion

Our work based on the method with deep learning neural network built on the top of word2vec and SSWE. We can find if we exploit the sentiment information in the pre-trained word vector we would get better result. Our work and some previous work mentioned in this paper show that unsupervised pre-training of word vectors plays an important role in deep learning for sentiment analysis.

References

Alfred. V. Aho and Jeffrey D. Ullman. 1972. *The Theory of Parsing, Translation and Compiling, volume 1*. Prentice-Hall, Englewood Cliffs, NJ.

American Psychological Association. 1983. *Publications Manual.* American Psychological Association, Washington, DC

Collobert, Ronan, and Jason Weston. "A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning." In Proceedings of the 25th International Conference on Machine Learning, 160–167. ACM, 2008. http://dl.acm.org/citation.cfm?id=1390177.

Collobert, J Weston, L.Bottou, M.Karlen, K.Kavukcugla, P.Kuksa. 2011"Natural Language Processing(Almost)from Scratch." Journal of Machine Learning Research 12:2493-2537

Kalchbrenner, Nal, Edward Grefenstette, and Phil Blunsom. "A Convolutional Neural Network for Modelling Sentences." arXiv Preprint arXiv:1404.2188, 2014. http://arxiv.org/abs/1404.2188.

Kim, Yoon. "Convolutional Neural Networks for Sentence Classification." arXiv Preprint arXiv:1408.5882, 2014. http://arxiv.org/abs/1408.5882.

Tang, Duyu, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. "Learning Sentiment-Specific Word Embedding for Twitter Sentiment Classification." In ACL (1), 1555–1565, 2014. http://anthology.aclweb.org/P/P14/P14-1146.pdf.

Gao J, He X, Yih W, et al. Learning continuous phrase representations for translation modeling//In ACL. 2014.

Pennington J, Socher R, Manning C D. Glove: Global Vectors for Word Representation//EMNLP. 2014, 14: 1532-1543.

Socher R, Perelygin A, Wu J Y, et al. Recursive deep models for semantic compositionality over a sentiment treebank//Proceedings of the conference on empirical methods in natural language processing (EMNLP). 2013, 1631: 1642.

Iyyer M, Boyd-Graber J L, Claudino L M B, et al. A Neural Network for Factoid Question Answering over Paragraphs//EMNLP. 2014: 633-644.