Sentiment Classification via Supplementary Information Modeling

Zenan Xu[1], Yetao Fu[1], Xingming Chen[1], Yanghui Rao[1(✉)], Haoran Xie[2], Fu Lee Wang[3], and Yang Peng[2]

[1] School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China
raoyangh@mail.sysu.edu.cn
[2] Department of Mathematics and Information Technology, The Education University of Hong Kong, Tai Po, Hong Kong
[3] School of Science and Technology, The Open University of Hong Kong, Kowloon, Hong Kong

Abstract. Traditional methods of annotating the sentiment of a document are based on sentiment lexicons, which have been proven quite efficient. However, such methods ignore the effect of supplementary features (e.g., negation and intensity words), while only consider the counts of positive and negative words, the sum of strengths, or the maximum sentiment score over the whole document primarily. In this paper, we propose to use convolutional neural network (CNN) and long short-term memory network (LSTM) to model the role of negation and intensity words, so as to address the limitations of lexicon-based methods. Results show that our model can not only successfully capture the effect of negation and intensity words, but also achieve significant improvements over state-of-the-art deep neural network baselines without supplementary features.

Keywords: Negation words · Intensity words
Sentiment supplementary information

1 Introduction

Sentiment analysis is a fundamental task of classifying given instances into classes such as positive, neutral, and negative, or fine-grained classes (e.g., very positive, positive, neutral, negative, very negative) in natural language processing. The traditional way of conducting the above task is based on sentiment lexicons [2,7]. Lexicon-based methods mainly exploit features such as the counts

The research has been supported by the National Natural Science Foundation of China (61502545, U1611264, U1711262), Guangdong Science and Technology Program grant (2017A050506025), a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (UGC/FDS11/E03/16), and the Internal Research Grant (RG 92/2017-2018R) of The Education University of Hong Kong.

Y. Cai et al. (Eds.): APWeb-WAIM 2018, LNCS 10987, pp. 54–62, 2018.
https://doi.org/10.1007/978-3-319-96890-2_5

of positive/negative words, total strengths, and the maximum strength [8]. Although such methods have been shown simple and efficient, they are typically based on bag-of-words models which ignore the semantic composition problem. For sentiment classification, the problem of semantic composition can appear in different ways including negation reversing (e.g., not interesting), negation shifting (e.g., not terrific), and intensification (e.g., very good). Another stream of work focuses on employing machine learning methods, e.g., there are various deep neural networks including CNN [9], recursive autoencoders [12], and LSTM [4], being exploited into sentiment analysis. However, these models also present the above limitation despite their great success.

To address the aforementioned semantic composition problem, we here present a hybrid model for sentiment classification by modelling the supplementary information of negation and intensity words. For example, we change sentence "the movie is not good" to "the movie is bad", and sentence "the movie is very boring" to "the movie is boring + boring". Particularly, we address the issue of semantic composition based on the linguistic role of negation and intensity words. The main contribution of this study is that we develop a backward LSTM to model the reversing effect of negation words and the valence that modified by the intensity words on the following content.

2 Proposed Model

This research aims to tackle the semantic composition issues of traditional lexicon-based methods for sentiment classification. The semantic composition problem can be dealt by modeling the linguistic role of negation and intensity words through a LSTM network. We incorporate the proposed sentiment supplementary information extracted from negation and intensity words into three neural networks, CNN [8], LSTM [6], and CharSCNN [5], and denote these new models as NIS-CNN, NIS-LSTM, and NIS-CharSCNN, where "NIS" means "Negation and Intensity Supplement". In this paper, we mainly introduce the NIS-CNN model, whose architecture is shown in Fig. 1.

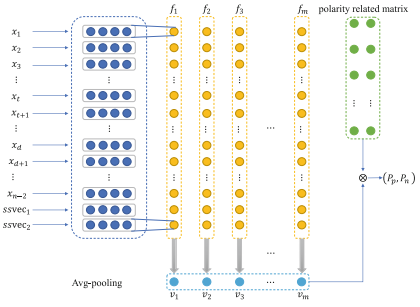

Fig. 1. The architecture of NIS-CNN

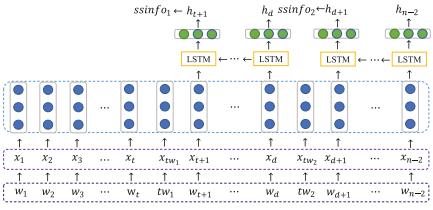

Fig. 2. Generation of $ssinfo$

2.1 Sentiment Supplementary Vector

We use LSTM to model the effect of negation and intensity words, which is called sentiment supplementary information. The generation of sentiment supplementary information ($ssinfo$) is shown in Fig. 2. A LSTM cell block consists of an input gate I_t, a memory cell C_t, a forget gate F_t, and an output gate O_t to make use of the information from the history $x_1, x_2, \ldots, x_t$ and $h_1, h_2, \ldots, h_{t-1}$ to generate O_t. Formally, O_t is computed as follows:

$$I_t = \sigma(W_i x_t + U_i h_{t-1} + V_i c_{t-1} + b_i), \tag{1}$$

$$F_t = 1.0 - I_t, \tag{2}$$

$$G_t = tanh(W_g x_t + U_g h_{t-1} + b_g), \tag{3}$$

$$C_t = f_t \odot c_{t-1} + i_t \odot g_t, \tag{4}$$

$$O_t = \sigma(W_o x_t + U_o h_{t-1} + V_o c_t + b_o), \tag{5}$$

where x_t is the word embedding of word w_t, σ denotes the sigmoid function, $\odot$ is element-wise multiplication. $\{W_i, U_i, V_i, b_i, W_q, U_q, b_q, W_o, U_o, V_o, b_o\}$ are LSTM parameters.

Now, we denote each negation word and intensity word as target word (tw). We now discuss three situations. Firstly, the model is unchanged if a sentence contains no tw. Secondly, if we have a sentence $S_t = [x_1, \ldots, x_t, tw, x_{t+1}, \ldots, x_n]$, which contains one tw, we use the backward LSTM on words $\{x_{t+1}, x_{t+2} \ldots, x_n\}$ and we get a $ssinfo$. Last but not the least, if we have another sentence $S_d = [x_1, \ldots, x_t, tw_1, x_{t+1}, \ldots, x_d, tw_2, x_{d+1} \ldots, x_n]$, which contains two tw, we use a backward LSTM on words $\{x_{t+1}, \ldots, x_d\}$ and words $\{x_{d+1} \ldots, x_n\}$ to achieve $ssinfo1$ and $ssinfo2$. To preserve the simplicity of the proposed model, we do not consider a sentence contains more than two target words.

After adding the $ssinfo$ into the original sentence and deleting tw, we get a new sentence $\{x_1, x_2, \ldots, x_{t-1}, x_t, \ldots, x_n, \lambda * ssinfo\}$. Here we call the $\lambda * ssinfo$ as a sentiment supplementary vector ($ssvec$). When it comes to intensity and negation words, the value of λ will be initially set to $+1$ and -2 respectively.

2.2 Training and Testing

This task aims to extract the feature map vector (denote as R) of every sentence through a simple CNN, and multiply it by the weight vector to calculate the relevancy between the sentence and the polarity. Finally, we choose the polarity with the largest relevancy as the label for the sentence.

A convolution operation which involves m filters $W \in R_1 * d$ is applied to the words one by one to generate the feature map of the sentence: $V_i = g(W * x_i)$, where "$*$" is a two-dimensional convolution operation and g indicates a non-linear function. The pooling layer is applied to calculate the whole representation

of the sentence from the sentiment information extracted by the filters from all words in the text. The average pooling will be used in this case, which aims to capture the average sentiment information so as to apply on the feature vector v. The average pooling is defined as:

$$r_{avg} = \frac{1}{n-h+1} \sum_{j=1}^{n-h+1} v_j. \tag{6}$$

The model uses two polarity related weight vectors (denoted as C_p and C_n) and feature map vector R obtained by pooling-layer to generate the score under different polarities (denoted as $Score_pi$, $Score_ni$) of the i-th sentence. Here we use L_i =1 and L_i =0 to indicate positive and negative sentiment. For the polarity, we use the softmax to calculate the possibility of being positive and negative as s_i^1 and s_i^0. s_i^1 and s_i^0 are estimated as

$$s_i^1 = \frac{e^{Score_{pi}}}{e^{Score_{pi}} + e^{Score_{pn}}}, \tag{7}$$

$$s_i^0 = \frac{e^{Score_{pn}}}{e^{Score_{pi}} + e^{Score_{pn}}}. \tag{8}$$

We use cross-entropy to calculate the loss of the model. Assumed that there have N training sentences, the loss function is defined as:

$$L|\theta| = -\sum_{i=1}^{N} logs_i^{L_i} + \frac{\lambda_r}{2}||\theta||^2, \tag{9}$$

where θ is the set of model parameters, λ_r is a parameter for $L2$ regularization.

3 Experiments

3.1 Dataset

We evaluate the proposed model on three datasets. The first one is Movie Review (MR) [11], in which every sentence is annotated with two classes as positive and negative. The second one is Stanford Sentiment Treebank (SST) [12], where each sentence is classified into five classes, including very negative, negative, neutral, positive, and very positive. The third one is Sentiment Labelled Sentences (SLS) [10], which is collected from reviews of products (Amazon), movies (IMDB), and restaurants (Yelp). Statistics of the three datasets are summarized in Table 1.

Negation and intensity words are derived from Linguistic Inquiry and Word Count (LIWC2007), in which a certain word is labelled according to its characteristic or property. We use all negation words from the Negate part of LIWC2007 and the intensity words manually from the Adverb part by removing some words that are obviously not intensity words.

Table 1. Dataset statistics. S: Number of sentences. L: Average sentence length. V: Vocabulary size. $|N|$: Percentage of documents with negation words. $|I|$: Percentage of documents with intensity words.

Dataset	S	L	V	$\|N\|$	$\|I\|$
MR	10662	20	18376	33.7%	53.2%
SST	9613	17	17439	25.8%	49.8%
SLS	3000	12	5170	27.8%	39.0%

3.2 Experiment Design

To evaluate the performance of the proposed NIS-CNN, NIS-LSTM, and NIS-CharSCNN, we implement the following baselines for comparison:

- CNN: which generates sentence representation by a convolutional layer with multiple kernels (i.e., kernels' size of 3, 4, 5 with 100 feature maps each) and pooling operations. Note that dropout operations are added to prevent over-fitting [8].
- LSTM: The whole corpus is processed as a single sequence, and LSTM generates the sentence representation by calculating the means of the whole hidden states of all words. The hidden state size is empirically set to 128 [6].
- CharSCNN: which employs two convolutional layers to extract features from characters to sentences. Following the convolutional layers are two fully-connected layers, the output of the second convolutional layer is passed to them to calculate the sentiment score. Empirically, the context windows of words and characters are set to 1. The convolution state size of the character-level layer and that of the word-level layer are respectively set to 20 and 150 [5].

Our experiments are implemented using the TensorFlow [1] and Keras [3] Python libraries. We use Stochastic Gradient Descent with Adadelta [13] for training. We set the batch size at each iteration to 32 and the size of word embeddings to 300 for all datasets and models. All other parameters are initialized to their default values as specified in the TensorFlow and Keras library. For all datasets, we randomly select 80% samples as the training set, 10% as validation samples, and the remaining 10% for testing.

In our negation and intensity supplement method, LSTM's hidden state sizes d and the dropout rate p are tuned on the validation set for each dataset. The values of d in MR, SST and SLS are 128, 256 and 128 respectively, and the values of p in MR, SST and SLS are 0.5, 0.3 and 0.2 respectively.

3.3 Evaluation Metrics

We use *Accuracy* to evaluate the model performance, as follows:

$$Accuracy = \frac{\sum_{i=1}^{N} tp_i + tn_i}{\sum_{i=1}^{N} tp_i + fp_i + tn_i + fn_i}, \tag{10}$$

where tp_i is 1 if the i-th sentence is positive and the prediction is positive, otherwise, it is 0. tn_i is 1 if the i-th sentence is negative and the prediction is negative, otherwise, it is 0. fp_i is 1 if the i-th sentence is negative and the prediction is positive, otherwise, it is 0. fn_i is 1 if the i-th sentence is positive and the prediction is negative, otherwise, it is 0. N is the number of sentences.

3.4 Results and Analysis

As shown in Table 2, in all datasets, the experimental results of NIS-CNN are superior to those baselines (e.g., CNN, LSTM, and CharSCNN) that do not consider negation and intensity words. We can conclude that the linguistic role of negation and intensity words that our model captured is effective.

Table 2. Accuracy (%) of all models on MR, SST and SLS datasets.

Model	MR	SST	SLS
CNN	75.8	80.2	85.6
IS-CNN	76.7	80.6	85.5
NS-CNN	78.6	82.1	87.4
NIS-CNN	**78.9**	**82.3**	**88.2**
LSTM	75.9	75.8	85.3
IS-LSTM	76.0	76.3	85.0
NS-LSTM	76.7	77.0	86.1
NIS-LSTM	**77.2**	**77.6**	**86.8**
CharSCNN	73.5	81.7	82.0
IS-CharSCNN	73.1	81.2	82.4
NS-CharSCNN	**74.6**	**82.9**	82.8
NIS-CharSCNN	74.5	82.7	**83.3**

We also conduct ablation experiments to evaluate the functional performance of negation words and intensity words respectively, these experiments are conducted on the entire dataset. First of all, we conduct the experiment with no negation and intensity words. Then we remove either negation words or intensity words each time on the basis of our model and execute the NS-CNN and the IS-CNN on the whole dataset respectively. In Table 2, significant improvement can be observed between CNN and NIS-CNN on MR (the accuracy rises from 75.8% to 78.9%), SST (the accuracy rises from 80.2% to 82.3%), SLS (the accuracy rises from 85.6% to 88.2%), which validates the effectiveness of NIS-CNN on modelling the linguistic role of negation and intensity words.

To further validate the effectiveness of the supplementary information, we conduct similar ablation experiments on LSTM and CharSCNN. Improvements

can also be seen between LSTM and NIS-LSTM on MR, SST, SLS, as well as between CharSCNN and NIS-CharSCNN on MR, SST, and SLS.

However, we find that methods with negation words only show significant improvement on the accuracy of binary classification compared with methods without negation and intensity words, while methods with intensity words only show a slight improvement and even a little descend. To explore the reason behind such phenomenon, we conduct detailed experiments as follows.

Table 3. Examples about the effect of negation words on MR dataset. *NW*: Negation word. *C*: Content. *Pos*: The probability of predicted Positive (%). *Neg*: The probability of predicted Negative (%).

Sentence	*NW*	*C*	CNN		NS-CNN	
			Pos	*Neg*	*Pos*	*Neg*
You cannot help but get caught up	cannot	Positive	38.1	61.9	56.1	43.9
Hollywood wouldn't have the guts to make	not	Positive	41.7	58.3	59.6	40.4
The story is nowhere near gripping enough	nowhere	Negative	69.1	30.9	36.4	63.6

For negation words, we extract all the sentences with negation words in MR dataset and compare the probability under different polarity predicted by CNN and NS-CNN. We can see in Table 3, for those sentences with negation words that were annotated with the false label by CNN, NS-CNN could correct such faults and consequently improved the accuracy. Therefore when we modeled the sentiment reversing effect of negation words and introduce it into CNN, we could correct those sentences that are classified into wrong classes by CNN.

Table 4. Examples about the effect of intensity words on MR dataset. *IW*: Intensity word. *C*: Content. *Pos*: The probability of predicted Positive (%). *Neg*: The probability of predicted Negative (%).

Sentence	*IW*	*C*	CNN		IS-CNN	
			Pos	*Neg*	*Pos*	*Neg*
An extremely unpleasant film	extremely	Negative	22.8	77.2	6.60	93.4
Really quite funny	really	Positive	73.5	26.5	85.7	14.3
Too silly to take seriously	too	Negative	19.8	80.2	16.3	83.7
The tenderness of the piece is still intact	still	Positive	52.8	47.2	48.7	51.3

For intensity words, we observe that intensity words just change the sentiment level of the sentence with intensity words but do not change the sentiment

polarity. For example, in Table 4, the sentence "An extremely unpleasant film" with the intensity word "extremely" is labelled correctly by CNN. When considering the sentiment shifting effect of intensity words, the probability of negative predicted by IS-CNN is still higher than the probability of positive, while the label keeps negative too. In summarize, when a sentence is annotated with a false label, considering intensity words will not help to correct it. Intensity words should play a more significant role in fine-grained sentiment classification tasks.

4 Conclusion

In this work, we proposed an effective model for sentiment classification. The proposed model addressed the sentiment reversing effect of negation words and the sentiment shifting effect of intensity words. Experimental results validate the effectiveness of our model. In the future, we plan to introduce the attention mechanism to model the valence of every word in the sentence, including the negation and intensity words that change the sentiment of the sentence. Furthermore, we will apply the similar process on negation and intensity words to conjunctions, which may shift the sentiment level of a sentence to some extent.

References

1. Abadi, M., Barham, P., Chen, J.M., Chen, Z.F., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D.G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., Zheng, X.Q.: Tensorflow: a system for large-scale machine learning. In: OSDI, pp. 265–283 (2016)
2. Baccianella, S., Esuli, A., Sebastiani, F.: SentiWordNet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining. In: LREC, pp. 2200–2204 (2010)
3. Choi, K., Joo, D., Kim, J.: Kapre: On-GPU audio preprocessing layers for a quick implementation of deep neural network models with Keras. CoRR, abs/1706.05781 (2017)
4. Chung, J.Y., Gulcehre, C., Cho, K.H., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. CoRR, abs/1412.3555 (2014)
5. Guerini, M., Gatti, L., Turchi, M.: Sentiment analysis: How to derive prior polarities from SentiWordNet. In: EMNLP, pp. 1259–1269 (2013)
6. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
7. Hu, M.Q., Liu, B.: Mining and summarizing customer reviews. In: KDD, pp. 168–177 (2004)
8. Kim, S.M., Hovy, E.: Determining the sentiment of opinions. In: COLING, Article no. 1367 (2004)
9. Kim, Y.: Convolutional neural networks for sentence classification. CoRR, abs/1408.5882 (2014)
10. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. CoRR, abs/1301.3781 (2013)

11. Pang, B., Lee, L.: Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating sales. In: ACL, pp. 115–124 (2005)
12. Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C.D., Ng, A., Potts, C.: Recursive deep models for semantic compositionality over a sentiment treebank. In: EMNLP, pp. 1631–1642 (2013)
13. Zeiler, M.D.: ADADELTA: an adaptive learning rate method. CoRR, abs/1212.5701 (2012)