Sentiment Classification Using Negative and Intensive Sentiment Supplement Information

Xingming Chen¹ · Yanghui Rao^{1,5} · Haoran Xie² · Fu Lee Wang³ · Yingchao Zhao⁴ · Jian Yin^{1,5}

Received: 13 December 2018 / Revised: 23 March 2019 / Accepted: 6 June 2019 / Published online: 15 June 2019
© The Author(s) 2019

Abstract

Traditional methods of annotating the sentiment of an unlabeled document are based on sentiment lexicons or machine learning algorithms, which have shown low computational cost or competitive performance. However, these methods ignore the semantic composition problem displaying in several ways such as negative reversing and intensification. In this paper, we propose a new method for sentiment classification using negative and intensive sentiment supplementary information, so as to exploit the linguistic feature of negative and intensive words in conjunction with the context information. Particularly, our method can solve the domain-specific problem without relying on the external sentiment lexicons. Experimental results on two real-world datasets demonstrate the effectiveness of our proposed method.

Keywords Negative words · Intensive words · Sentiment supplementary information

Extended from their APWEB-WAIM 2018 papers

✉ Yanghui Rao
raoyangh@mail.sysu.edu.cn
Xingming Chen
chenxm47@mail2.sysu.edu.cn
Haoran Xie
hrxie2@gmail.com
Fu Lee Wang
pwang@ouhk.edu.hk
Yingchao Zhao
yczhao@cihe.edu.hk
Jian Yin
issjyin@mail.sysu.edu.cn

¹ School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China

² Department of Mathematics and Information Technology, The Education University of Hong Kong, Tai Po, New Territories, Hong Kong

³ School of Science and Technology, The Open University of Hong Kong, Ho Man Tin, Kowloon, Hong Kong

⁴ School of Computing and Information Sciences, Caritas Institute of Higher Education, Tseung Kwan O, Hong Kong

⁵ Guangdong Key Laboratory of Big Data Analysis and Processing, Sun Yat-sen University, Guangzhou, China

1 Introduction

Sentiment analysis is a fundamental task of classifying given instances into coarse-grained 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 [3, 10, 31]. Sentiment lexicons can serve as a word-level basis to help analyze the sentiment of unlabeled documents for the discrete information such as polarities and strengths they contain. Lexicon-based methods mainly exploited features such as the counts, the total strengths, and the maximum strengths of positive and negative words [2, 12, 31]. For example, a straightforward method was proposed in [10, 29] by counting the aggregated value of sentiment strengths of all words that exist in a sentiment lexicon to analyze the sentiment of each document. Although such methods have been shown simple and efficient, they suffer from the imitation of existing sentiment lexicons. Particularly, a fixed polarity or strength is assigned to each word in a sentiment lexicon, but the same word under different domains may have different polarities or strengths. Take “hot” as an example, it expresses positive in comments of a popular song while negative in comments of a restaurant. Another stream of work focused on employing machine learning methods. Unlike lexicon-based models which leverage lexicon features to predict sentiments,

machine learning-based methods are data dependent because most of them are trained on a specific corpus. Benefit from the growing of user generated messages, there are various deep neural networks including CNN [13, 15], recursive autoencoders [24, 25], and LSTM [6, 17, 27, 34], being exploited into sentiment analysis. However, these models also present some drawbacks despite their great success. For example, most of these methods ignore the semantic composition problem. Semantic composition can display in several ways such as negative reversing (e.g., not interesting), negative shifting (e.g., not terrific), and intensification (e.g., very good). Although the above issue can be alleviated by tree-structured models like recursive autoencoders and Tree-LSTM [27, 34], it is quite time-consuming to parse tree structures and annotate phrase-level features.

A method called supplementary information modeling [32] was recently proposed for sentiment classification by taking the role of negative and intensive words into consideration. In this method, two sentences “the movie is not good” and “the movie is very boring” approximate to “the movie is bad” and “the movie is boring + boring,” by generating new representation for “not good” and “very boring,” respectively. However, it may drop some context information by considering the sentiment supplementary information only. For the sake of overcoming the above shortcoming, we propose a new model for sentiment classification using negative and intensive sentiment supplementary information. In our model, we keep all information of the corpus in conjunction with the negative and intensive supplementary information which can exploit the linguistic feature of negative and intensive words. When a negative or intensive word exists in a sentence, we use a backward LSTM to encode the semantics after it and generate a new word embedding vector to emphasize the effect of the negative or intensive word. Furthermore, local sentiment attributes of all words through several deep learning networks are generated to help predict the sentiment of a given document. Our contributions are three-folds as follows:

- This paper proposes the concept of semantic composition based on the linguistic role of negative and intensive words. Without dropping any context information, we develop a backward LSTM to model the reversing effect of negative words and the valence that modified by the intensive words on the following content.
- Unlike previous lexicon-based methods which directly employ external sentiment lexicons, we generate the sentiment strength under different sentiment polarities of each word in the corpus, which can solve the domain-specific problem for methods based on external sentiment lexicons.
- Our method can adapt to the situation that two negative or intensive words exist in a sentence, and can be

employed to several deep neural network models to improve the performance on sentiment classification.

The remainder of this paper is organized as follows. We describe related work in Sect. 2. We present the method for sentiment classification in Sect. 3. We detail the dataset, results, and discussions in Sect. 4. Finally, we present conclusions in Sect. 5.

2 Related Work

2.1 Lexicon-Based Sentiment Classification

Sentiment lexicons [10, 31] usually define the prior sentiment attributes (i.e., polarity or strengths) of a collection of words and phrases, which is useful for lexicon-based methods [2, 4, 7, 8, 13, 19, 26, 30] in sentiment classification. There are two mainstreams of lexicon-based methods, one is under the bag-of-words framework and the other is rule based. In the bag-of-words framework, lexicon features like the counts of sentiment words, their total strengths, and the maximum strength are leveraged to predict the sentiment of a given document. Beyond bag-of-words models that exploit sentiment lexicons, rule-based methods [20, 26] introduce semantic composition [21] into sentiment operations. However, the above methods are domain-specific primarily.

2.2 Deep Neural Networks for Sentiment Classification

Unlike lexicon-based methods that predict sentiments based on the bag-of-words assumption or straightforward rules, deep neural networks predict sentence-level sentiments of given documents by training various neural networks, such as convolutional [11, 13, 16, 23], on a large amount of labeled data. A noticeable work can be seen in Teng’s research [28], in which the sentiment of each sentence is determined by the weighted sum score of negative words and sentiment words, and the weights are learned by a neural network. These models all achieved competitive accuracies, and their main characteristics are as follows. Compared with recursive models, which usually require fine-grained annotations and tree-structured data, convolutional neural network does not need these features. Long short-term memory models [9], which are usually used to model the prefix or suffix context, can be also applied to sequential data and tree-structured data [27, 34]. A method called supplementary information modeling [32] is integrated to several deep neural networks to model the role of negative and intensive words for sentiment classification, and it is validated to be effective.

3 Proposed Model

This research aims to tackle the semantic composition issue of existing deep neural networks for sentiment classification. Particularly, we model the distinct effects of negative and intensive words through a LSTM network, as follows: (1) *Negative Expression Modeling*. Negative words, such as “not” and “never,” mainly have a sentiment reversing effect on the following expression. In most cases, the sentiment polarity of the content following a negative word is reversed. To this end, we convert a sentence such as “the movie is not good” to “the movie is bad” approximately, by employing a backward LSTM to model each negative word in conjunction with all following words. (2) *Intensive Expression Modeling*. Intensive words, such as “very” and “so,” mainly change the sentiment strength of the following expression (e.g., from

the negative side to the very negative side). Take the sentence “the movie is so boring” as an example, by employing a backward LSTM on the expression of “so boring,” we can convert the original sentence into a new sentence like “the movie is boring + boring.” For the convenience of describing our method, frequently used notations are summarized in Table 1.

To validate the effectiveness of the above operations, we incorporate them into three deep neural networks, CNN [12], LSTM [9], and CharSCNN [8], and denote these new models as NIM-CNN, NIM-LSTM, and NIM-CharSCNN, where “NIM” means “Negative and Intensive Modeling.” For example, the architecture of NIM-CNN is shown in Fig. 1.

In the following, we first describe how to generate sentiment supplementary information. Then, we detail the sentence encoding using the above information. Finally, we

Table 1 Frequently used notations

Notation	Description
tw	The target word, i.e., a negative or intensive word
ntw	The number of target words
$S = [x_1, x_2, \dots, x_n]$	A sentence with n words
$x_i \in R^d$	A d -dimensional word embedding of the i th word
$V = [v_1, v_2, \dots, v_{n+ntw}]$	The vector representations generated by our method
$F = [f_1, f_2, \dots, f_m]$	A hidden vector containing sentiment feature of each sentence
r_{avg}	A vector containing the average information
f_i	The sentiment feature of the i th word
$W \in ws * d$	A convolutional filter applied to continuous word embeddings
$ssinfo$	Sentiment supplement information extract by the backwards LSTM
$ssvec$	A sentiment supplement vector generate by λ times the $ssinfo$
s_i^1	The positive polarity strength of the i th sentence
s_i^0	The negative polarity strength of the i th sentence
C_p	The predicted value of the positive label
C_n	The predicted value of the negative label

Fig. 1 The architecture of NIM-CNN

show methods of training the proposed model and predicting the sentiment of a new document.

3.1 Sentiment Supplementary Information Generation

We use LSTM to model the effect of negative and intensive words, which is called sentiment supplementary information. The generation of sentiment supplementary information (*ssinfo*) is shown in Fig. 2. A LSTM cell block employs 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 previous inputs. Formally, given the input x_t at time step t , O_t is computed as follows:

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

$$F_t = 1.0 - I_t, \quad (2)$$

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

$$C_t = F_t \odot C_{t-1} + I_t \odot G_t, \quad (4)$$

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

where h_t and h_{t-1} denote the current and previous hidden state, respectively. σ denotes to the sigmoid function, $\odot$ refers to element-wise multiplication, and $\{W_i, U_i, V_i, b_i, W_g, U_g, b_g, W_o, U_o, V_o, b_o\}$ are LSTM parameters.

A backward LSTM is used to encode the content following each target word (*tw*) and generate *ssinfo*, where *tw* denotes a negative or intensive word. We now discuss three situations. Firstly, given a sentence $S = [x_1, x_2, \dots, x_i, \dots, x_n]$, where x_i denotes the word embedding of the i th word in the sentence. If the sentence does not contain *tw*, the *ssinfo* of S is empty. Secondly, given a sentence $S = [x_1, x_2, \dots, x_{t-1}, tw, x_{t+1}, \dots, x_n]$, which contains one *tw*, we use the backward LSTM on word embeddings

$\{x_{t+1}, x_{t+2}, \dots, x_n\}$ and get the *ssinfo* of S . Finally, given a sentence $S = [x_1, \dots, x_{t-1}, tw_1, x_{t+1}, \dots, x_{d-1}, tw_2, x_{d+1}, \dots, x_n]$, which contains two instances of *tw*, we use a backward LSTM on word embeddings $\{x_{t+1}, \dots, x_{d-1}\}$ and word embeddings $\{x_{d+1}, \dots, x_n\}$ to generate *ssinfo*₁ and *ssinfo*₂, respectively. Considering the efficiency of the proposed model, we do not consider the situation that a sentence contains more than two target words.

3.2 Sentence Encoding

After the generation of *ssinfo*, we encode the sentence to a new vector representation V . Given the sentence $S = [x_1, x_2, \dots, x_{t-1}, tw, x_{t+1}, \dots, x_n]$, we get a new sentence representation $\{x_1, x_2, \dots, x_{t-1}, tw, x_{t+1}, \dots, x_n, \lambda * ssinfo\}$ after applying the backward LSTM on the sequence $\{x_{t+1}, \dots, x_n\}$. Here, we call $\lambda * ssinfo$ as a sentiment supplement vector (*ssvec*). For negative words, the value of λ will be initially set to -2 . For intensive words, the value of λ will be initially set to $+1$. After adding the above *ssvec* to each sentence, the model generates a new vector representation of the sentence which could be described as $V = [v_1, v_2, \dots, v_i, \dots, v_{n+ntw}]$, where v_i denotes the i th vector representation in V , and ntw denotes the number of target words.

3.3 Model Training and Sentiment Prediction

After sentence encoding, we can employ V to a deep neural network for model training and sentiment prediction. Take CNN as an instance. In order to extract the sentiment information of each element in V , a convolution operation which involves many filters $W \in ws * d$ is applied to V to generate the feature map of V by: $F = g(W * V)$, where “ $*$ ” is a two-dimensional convolution operation and g indicates a nonlinear function. Then, the average pooling operation

Fig. 2 Sentiment supplement information

is employed to capture the average sentiment information, which is defined as:

$$r_{\text{avg}} = \frac{1}{n + ntw - ws + 1} \sum_{j=1}^{n+ntw-ws+1} f_j, \quad (6)$$

where f_j is the j th element of the feature map F .

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 to indicate the actual sentiment of the i th sentence. For the polarity, we use the softmax to calculate the possibility of being positive and negative as s_i^1 and s_i^0 . In particular, s_i^1 and s_i^0 are estimated as:

$$s_i^1 = \frac{e^{\text{Score}_{pi}}}{e^{\text{Score}_{pi}} + e^{\text{Score}_{ni}}}, \quad (7)$$

$$s_i^0 = \frac{e^{\text{Score}_{ni}}}{e^{\text{Score}_{pi}} + e^{\text{Score}_{ni}}}. \quad (8)$$

Note that the whole model is trained end-to-end and *ssinfo* is also updated along with the other components. We use cross-entropy to calculate the loss of the model. Assume that there are N training sentences; the loss function is defined as:

$$\text{Loss}|\theta| = - \sum_{i=1}^N L_i \log s_i^{L_i} + \frac{\lambda_r}{2} \|\theta\|^2, \quad (9)$$

where θ is the set of model parameters, λ_r is a parameter for L_2 regularization.

4 Experiments

4.1 Datasets

We evaluate the proposed model on three datasets. The first one is Movie Review (MR) [22], in which every sentence is annotated with two classes as positive and negative. The second one is Stanford Sentiment Treebank (SST) [16, 25], where each sentence is classified into five classes, including very negative, negative, neutral, positive, and very positive. The third one is Sentiment Labeled Sentences (SLS) [18], which is collected from reviews of products (Amazon), movies (IMDB), and restaurants (Yelp). Statistics of the three datasets are summarized in Table 2.

Negative and intensive words are derived from Linguistic Inquiry and Word Count (LIWC2007), in which a certain word is labeled according to its characteristic or property. We use all negative words from the “Negate” part of LIWC2007 and select intensive words manually from the

Table 2 Dataset statistics

Dataset	N_s	L_s	V	N (%)	I (%)
MR	10,662	20	18,376	33.7	53.2
SST	9613	17	17,439	25.8	49.8
SLS	3000	12	5170	27.8	39.0

N_s number of sentences, L_s average sentence length, |V| vocabulary size, |N| percentage of documents with negative words, |I|, percentage of documents with intensive words

Table 3 Examples of negative and intensive words

Negative words	Intensive words
Cannot	Cannot
Negate	Absolutely
Neither	Completely
Never	Even
No	Just
Nobody	Mostly
...	...

“Adverb” part by removing some words that are obviously not intensive words. Some of the negative and intensive words are shown in Table 3.

4.2 Experiment Design

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

- Sentiment lexicon-based methods. Such methods annotate the sentiment of each unlabeled document by summing sentiment strengths of all words that exist in the sentiment lexicon [10, 29]. Here, we use three sentiment lexicons, which are SentiWordNet [3], SCL-NMA [14], and Opinion Lexicon [10].
- CNN [12]. It 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 the dropout operation is added to prevent overfitting.
- LSTM [9]. The whole corpus is process as a single sequence, and LSTM generates the sentence representation by calculating the mean of the whole hidden states of all words. The hidden state size was empirically set to 128.

- CharSCNN [8]. It employs two convolutional layers to extract features from character and sentence levels, and the output of the second convolutional layer is passed to two fully connected layer is passed to two fully connected layers to calculate the sentiment score. Empirically, the context windows of word and character were set to 1. The convolution kernel size of the character-level layer and that of the sentence-level layer were, respectively, set to 20 and 150.
- Supplementary information modeling-based methods [32]. Such methods incorporate a kind of sentiment supplementary information into three neural networks, i.e., CNN, LSTM, and CharSCNN. These new models are denoted as NIS-CNN, NIS-LSTM, and NIS-CharSCNN, where “NIS” means “Negative and Intensive Supplement.”

Our experiments were implemented using the TensorFlow [1] and Keras [5] Python libraries. We used Stochastic Gradient Descent with Adadelta [33] for training, which can adjust the learning rate adaptively rather than rely on a global variable. We set the batch size at each iteration to 32 and the size of word embeddings to 300 for all datasets and models. Furthermore, word embeddings were obtained by the word2vec tool. All other parameters were initialized to their default values as specified in the TensorFlow and Keras libraries. For all datasets, we randomly selected 80% samples as the training set, 10% as validation samples and the remaining 10% for testing.

In our negative and intensive supplement method, LSTM’s hidden state sizes d and the dropout rate p were tuned on the validation set for each dataset. Particularly, the optimal values of these parameters for each dataset are shown in Table 4.

4.3 Evaluation Metrics

We use Accuracy, Precision, Recall and F-measure to evaluate the model performance, as follows:

$$\begin{aligned}\text{Accuracy} &= \frac{\sum_{i=1}^S tp_i + tn_i}{\sum_{i=1}^S tp_i + fp_i + tn_i + fn_i}, \\ \text{Precision} &= \frac{\sum_{i=1}^S tp_i}{\sum_{i=1}^S tp_i + fp_i}, \\ \text{Recall} &= \frac{\sum_{i=1}^S tp_i}{\sum_{i=1}^S tp_i + fn_i}, \\ \text{F-measure} &= \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}},\end{aligned}$$

Table 4 Optimal hyperparameters for each dataset

Dataset	d	p
MR	128	0.5
SST	256	0.3
SLS	128	0.2

where tp_i is 1 if the i th sentence is positive actually and the predicted label is positive, otherwise, it is 0. tn_i is 1 if the i th sentence is negative actually and the predicted label is negative, otherwise, it is 0. fp_i is 1 if the i th sentence is negative and the predicted label is positive, otherwise, it is 0. fn_i is 1 if the i th sentence is positive and the predicted label is negative, otherwise, it is 0. S is the number of sentences.

4.4 Results and Analysis

As shown in Table 5, the proposed NIM-CNN performed better than those baselines for all datasets. Compared with deep neural network (DNN)-based methods, we can observe that sentiment lexicon-based methods have a relatively poor performance. On the one hand, sentiment lexicons are domain specific, but the same word under different domains may have different polarities and different strengths. On the other hand, sentiment lexicon-based methods are typically based on bag-of-words models which ignore the semantic composition problem. In contrast, DNN-based methods are data dependent and can learn high-level interactions among deep latent features which contribute a lot to predict the sentiment polarity.

We also conducted ablation experiments to evaluate the functional performance of negative words and intensive words, respectively; these experiments were conducted on the three datasets mentioned above. First of all, we conducted the experiment with no negative and intensive words. Then, we removed either negative words or intensive words each time on the basis of our model and executed the NM-CNN and the IM-CNN on the whole dataset, respectively. In Table 5, significant improvement could be observed among CNN, NIS-CNN and NIM-CNN on MR (the accuracy increases from 78.9% to 79.8%, and then to 80.1%), SST (the accuracy increases from 81.6% to 82.1%, and then to 82.3%), SLS (the accuracy increases from 87.8% to 88.3%, and then to 88.6%), which validated the effectiveness of NIM-CNN on modeling the linguistic role of negative and intensive words. To further validate the effectiveness of the supplement information provided by negative and intensive words, we conducted similar ablation experiments on LSTM and CharSCNN. Improvements could also be observed among LSTM, NIS-LSTM, and NIM-LSTM on MR (the accuracy increases from 75.9 to 76.2%, and then to 76.3%), SST (the accuracy increases from 75.8 to 76.3%, and then to 76.4%), and SLS (the accuracy increases from 85.8 to 86.1%, and then to 86.4%), as well as among CharSCNN, NIS-CharSCNN, and NIM-CharSCNN on MR (the accuracy increases from 74.0 to 74.4%, and then to 74.6%), SST (the accuracy increases from 81.7 to 82.0%, and then to 82.3%), and SLS (the accuracy increases from 86.4 to 86.9%, and then to 87.3%). Although supplementary information modeling methods performed better than conventional DNN-based

Table 5 Accuracy (%) of all models on MR, SST, and SLS datasets

Model	Dataset	Accuracy	Precision	Recall	F1-measure
SentiWordNet [3]	MR	58.3	56.3	77.8	65.3
	SLS	64.9	59.7	86.1	70.5
	SST	60.1	59.0	78.4	67.3
SCL-NMA [14]	MR	60.9	59.2	82.0	68.7
	SLS	69.5	66.4	88.8	76.0
	SST	65.0	63.4	84.7	72.5
Opinion Lexicon [10]	MR	69.0	68.8	73.9	71.3
	SLS	80.6	76.5	94.9	84.7
	SST	73.7	74.6	78.4	76.5
CNN [12]	MR	78.9	79.5	77.9	78.7
	SLS	87.8	89.3	85.9	87.6
	SST	81.6	82.5	80.2	81.3
NIS-CNN [32]	MR	79.8	79.7	80.0	79.8
	SLS	88.3	89.7	86.5	88.1
	SST	82.1	82.6	81.3	82.0
IM-CNN	MR	79.1	79.2	78.9	79.1
	SLS	88.0	89.0	86.7	87.8
	SST	81.7	82.3	80.1	81.5
NM-CNN	MR	79.7	79.9	79.4	79.6
	SLS	88.4	89.5	87.0	88.2
	SST	82.2	82.8	81.3	82.0
NIM-CNN	MR	80.1	79.8	80.6	80.2
	SLS	88.6	89.9	87.0	88.4
	SST	82.3	82.9	81.4	82.1
LSTM [9]	MR	75.9	75.4	76.9	76.1
	SLS	85.8	86.0	85.5	85.8
	SST	75.8	77.6	72.3	75.0
NIS-LSTM [32]	MR	76.2	76.0	76.6	76.3
	SLS	86.1	86.3	85.8	86.1
	SST	76.3	77.5	74.1	75.8
IM-LSTM	MR	76.0	75.3	77.4	76.3
	SLS	85.9	85.8	86.0	85.9
	SST	75.8	77.9	72.0	74.9
NM-LSTM	MR	76.2	75.8	77.0	76.4
	SLS	86.2	86.3	86.1	86.2
	SST	76.2	78.4	72.3	75.2
NIM-LSTM	MR	76.3	76.2	76.5	76.3
	SLS	86.4	86.5	86.3	86.4
	SST	76.4	78.7	72.4	75.4
CharSCNN [8]	MR	74.0	75.1	71.8	73.4
	SLS	86.4	88.6	85.3	86.9
	SST	81.7	83.1	79.6	81.3
NIS-CharSCNN [32]	MR	74.4	75.5	72.2	73.8
	SLS	86.9	88.4	85.9	87.3
	SST	82.0	83.5	79.8	81.6
IM-CharSCNN	MR	74.1	75.1	72.1	73.6
	SLS	86.6	88.4	85.2	86.7
	SST	81.9	83.3	79.8	81.5
NM-CharSCNN	MR	74.4	75.7	71.9	73.7
	SLS	86.9	88.9	85.7	87.3
	SST	82.2	83.4	80.4	81.8

Table 5 (continued)

Model	Dataset	Accuracy	Precision	Recall	F1-measure
NIM-CharSCNN	MR	74.6	75.2	73.4	74.3
	SLS	87.3	88.8	85.9	87.3
	SST	82.3	83.6	80.4	82.0

methods by taking the role of negative and intensive words into consideration, it may drop some salient context information since it removes the target words and the sequences of words following them. By keeping the context information of the whole sentence, our *ssinfo* has a positive impact on the performance.

The performance improvement of our model over baselines on the MR dataset is larger than that on SST when considering negative and intensive words. The reason may be that although the total number of sentences in MR is similar to that in SST, the percentage of sentences with negative words in MR is larger than that in SST. To sum up, modeling the sentiment reversing effect of negative words can significantly improve the accuracy of sentiment prediction by correcting the labels of sentences with negative words that are annotated with wrong labels.

However, we also observe that methods with negative words showed significant improvement on the accuracy of sentiment classification when compared with methods without negative and intensive words, while methods with intensive words only showed a slight improvement and even a little descend. To explore the reason behind such phenomenon, we conducted detail experiments as follows.

For negative words, we extracted all the sentences with negative words in the MR dataset and compared the

probability under different polarities predicted by CNN and NM-CNN. We could observe in Table 6, for those sentences with negative words that were annotated with the false label by CNN, NM-CNN can correct such faults and consequently improve the accuracy. This is because not only CNN, but also LSTM and CharSCNN, do not take the sentiment reversing effect of negative words into account and do not process the negative words as special words. Therefore, when we model the sentiment reversing effect of negative words and introduce it into CNN, the probability under different polarities will reverse too. Then, we can correct those sentences that are classified into wrong classes by CNN.

For intensive words, we conducted similar experiments as on negative words. From the experimental results, we can draw the reason of why the modeling of intensive words cannot improve the accuracy as much as negative words, that is, intensive words just change the sentiment level of the sentence with intensive words but do not change the sentiment polarity. Therefore, even though we can successfully model the sentiment shifting effect of intensive words and incorporate it into basic methods, the new methods still annotate the sentence with the same label as the original method has labeled. For example, in Table 7, the sentence “An extremely unpleasant film” with the intensive word “extremely” is labeled correctly

Table 6 Examples about the effect of negative words on MR dataset

Sentence	NW	C	CNN		NM-CNN	
			Pos	Neg	Pos	Neg
You cannot help but get caught up	Cannot	Positive	38.1	61.9	58.3	41.7
Hollywood wouldn't have the guts to make	Not	Positive	41.7	58.3	61.4	38.6
The story is nowhere near gripping enough	Nowhere	Negative	69.1	30.9	32.9	67.1

NW negative word, C category. Pos the predicted probability of being positive (%). Neg the predicted probability of being negative (%)

Table 7 Examples about the effect of intensive words on MR dataset

Sentence	IW	C	CNN		IM-CNN	
			Pos	Neg	Pos	Neg
An extremely unpleasant film	Extremely	Negative	22.8	77.2	5.3	94.7
Really quite funny	Really	Positive	73.5	26.5	88.0	12.0
Too silly to take seriously	Too	Negative	19.8	80.2	11.5	88.5
The tenderness of the piece is still intact	Still	Positive	52.8	47.2	42.7	57.3

IW intensive word, C category, Pos the predicted probability of being positive (%), Neg the predicted probability of being negative (%)

by CNN. When considering the sentiment shifting effect of intensive words, the probability of negative predicted by IM-CNN is higher than the probability of being positive, while the label keeps negative too. In summary, when a sentence is annotated with a false label, considering intensive words will not help to correct it. Intensive words may play a more significant role in fine-grained sentiment classification tasks.

5 Conclusions

In this work, we proposed an effective model for sentiment classification. Without drop any context information, the proposed model addressed the sentiment reversing effect of negative words and the sentiment shifting effect of intensive words. Experimental results validated the effectiveness of our model. In the future, we plan to introduce attention mechanisms to model the valence of every word in the sentence, including the negative and intensive words that change the sentiment of the sentence. Furthermore, we will apply the similar process on negative and intensive words to conjunctions, which may shift the sentiment level of a sentence to some extent.

Acknowledgements We are grateful to the anonymous reviewers for their valuable comments on this article. This research has been supported in part by the Research Grants Council of the Hong Kong Special Administrative Region, China (UGC/FDS11/E03/16), the Interdisciplinary Research Scheme of the Dean's Research Fund 2018–19 (FLASS/DRF/IDS-3), Top-Up Fund (TFG-04) for General Research Fund/Early Career Scheme of the Dean's Research Fund (DRF) 2018–19 and the Internal Research Grant (RG 90/2018–2019R) of The Education University of Hong Kong, the National Key R&D Program of China (2018YFB1004404), Key R&D Program of Guangdong Province (2018B010107005), and National Natural Science Foundation of China (U1711262, U1401256, U1501252, U1611264, U1711261). The preliminary version of this article has been published in APWeb-WAIM 2018 [32].

Author's Contributions XC (Experimental design and evaluations), YR (Model development), HX (Manuscript refinement), FLW (Manuscript refinement), YZ (Manuscript refinement), JY (Manuscript refinement).

Funding The National Natural Science Foundation of China (61502545, U1711262, U1611264), a grant from Research Grants Council of Hong Kong Special Administrative Region, China (UGC/FDS11/E03/16), and the Innovation and Technology Fund (Project No. GHP/022/17GD) from the Innovation and Technology Commission of the Government of the Hong Kong Special Administrative Region.

Compliance with Ethical Standards

Consent for Publication Yes.

Conflict of interest All authors declare that they have no conflict of interest.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

1. Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, Devin M, Ghemawat S, Irving G, Isard M et al (2016) Tensorflow: a system for large-scale machine learning. In: OSDI volume 16, pp 265–283
2. Agarwal A, Xie B, Vovsha I, Rambow O, Passonneau R (2011) Sentiment analysis of twitter data. In: Proceedings of the workshop on languages in social media. Association for Computational Linguistics, pp 30–38
3. Baccianella S, Esuli A, Sebastiani F (2010) Sentiwordnet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining. In: LREC, vol 10, pp 2200–2204
4. Cambria E (2016) Affective computing and sentiment analysis. *IEEE Intell Syst* 31(2):102–107
5. Choi K, Joo D, Kim J (2017) Kapre: On-gpu audio preprocessing layers for a quick implementation of deep neural network models with keras. [arXiv:1706.05781](https://arxiv.org/abs/1706.05781)
6. Chung J, Gulcehre C, Cho KH, Bengio Y (2014) Empirical evaluation of gated recurrent neural networks on sequence modeling. [arXiv:1412.3555](https://arxiv.org/abs/1412.3555)
7. Dong L, Wei F, Tan C, Tang D, Zhou M, Ke X (2014) Adaptive recursive neural network for target-dependent twitter sentiment classification. In: ACL, vol 2, pp 49–54
8. Guerini M, Gatti L, Turchi M (2013) Sentiment analysis: how to derive prior polarities from sentiwordnet. [arXiv:1309.5843](https://arxiv.org/abs/1309.5843)
9. Hochreiter S, Schmidhuber J (1997) Long short-term memory. *Neural Comput* 9(8):1735–1780
10. Hu M, Liu B (2004) Mining and summarizing customer reviews. In: SIGKDD, pp 168–177
11. Kalchbrenner N, Grefenstette E, Blunsom P (2014) A convolutional neural network for modelling sentences. [arXiv:1404.2188](https://arxiv.org/abs/1404.2188)
12. Kim S-M, Hovy E (2004) Determining the sentiment of opinions. In: COLING, pp 1367–1373
13. Kim Y (2014) Convolutional neural networks for sentence classification. *arXiv preprint* [arXiv:1408.5882](https://arxiv.org/abs/1408.5882)
14. Kiritchenko S, Mohammad S (2016) The effect of negators, modals, and degree adverbs on sentiment composition. In: Proceedings of the 7th workshop on computational approaches to subjectivity, sentiment and social media analysis. Association for Computational Linguistics, pp 43–52
15. Lei T, Barzilay R, Jaakkola T (2015) Molding CNNs for text: non-linear, non-consecutive convolutions. [arXiv:1508.04112](https://arxiv.org/abs/1508.04112)
16. Li J, Luong M-T, Jurafsky D, Hovy E (2015) When are tree structures necessary for deep learning of representations? *arXiv preprint* [arXiv:1503.00185](https://arxiv.org/abs/1503.00185)
17. Mikolov T (2012) Statistical language models based on neural networks. Presentation at Google, Mountain View, 2nd April
18. Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. [arXiv:1301.3781](https://arxiv.org/abs/1301.3781)
19. Mohammad SM, Kiritchenko S, Zhu X (2013) Nrc-Canada: building the state-of-the-art in sentiment analysis of tweets. [arXiv:1308.6242](https://arxiv.org/abs/1308.6242)
20. Moilanen K, Pulman S (2007) Sentiment composition. In: RANLP, vol 7, pp 378–382

21. Montague R (1974) Formal philosophy: selected papers of Richard Montague. Ed. and with an introd. by Richmond H. Thomason. Yale University Press, New Haven
22. Pang B, Lee L (2005) Exploiting class relationships for sentiment categorization with respect rating sales. *ACL*, pp 115–124
23. Ren Y, Zhang Y, Zhang M, Ji D (2016) Context-sensitive twitter sentiment classification using neural network. In: *AAAI*, pp 215–221
24. Socher R, Pennington J, Huang EH, Ng AY, Manning CD (2011) Semi-supervised recursive autoencoders for predicting sentiment distributions. In: *EMNLP*, pp 151–161
25. Socher R, Perelygin A, Wu J, Chuang J, Manning CD, Ng A, Potts C (2013) Recursive deep models for semantic compositionality over a sentiment treebank. In: *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp 1631–1642
26. Taboada M, Brooke J, Tofiloski M, Voll K, Stede M (2011) Lexicon-based methods for sentiment analysis. *Comput linguist* 37(2):267–307
27. Tai KS, Socher R, Manning CD (2015) Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint [arXiv:1503.00075](https://arxiv.org/abs/1503.00075)*
28. Teng Z, Vo DT, Zhang Y (2016) Context-sensitive lexicon features for neural sentiment analysis. In: *Proceedings of the 2016 conference on empirical methods in natural language processing*, pp 1629–1638
29. Turney PD (2002) Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In: *ACL*, pp 417–424
30. Vo D-T, Zhang Y (2015) Target-dependent twitter sentiment classification with rich automatic features. In: *IJCAI*, pp 1347–1353
31. Wilson T, Wiebe J, Hoffmann P (2005) Recognizing contextual polarity in phrase-level sentiment analysis. In: *EMNLP*, pp 347–354
32. Xu Z, Fu Y, Chen X, Rao Y, Xie H, Wang FL, Peng Y (2018) Sentiment classification via supplementary information modeling. In: *APWeb-WAIM*, pp 54–62
33. Zeiler MD (2012) Adadelata: an adaptive learning rate method. [arXiv:1212.5701](https://arxiv.org/abs/1212.5701)
34. Zhu X, Sobihani P, Guo H (2015) Long short-term memory over recursive structures. In: *ICML*, pp 1604–1612