 CMC, vol.56, no.2, pp.285-297, 2018

Sentiment Classification Based on Piecewise Pooling Convolutional Neural Network

Yuhong Zhang[1, *], Qinqin Wang[1], Yuling Li[1] and Xindong Wu[2]

Abstract: Recently, the effectiveness of neural networks, especially convolutional neural networks, has been validated in the field of natural language processing, in which, sentiment classification for online reviews is an important and challenging task. Existing convolutional neural networks extract important features of sentences without local features or the feature sequence. Thus, these models do not perform well, especially for transition sentences. To this end, we propose a Piecewise Pooling Convolutional Neural Network (PPCNN) for sentiment classification. Firstly, with a sentence presented by word vectors, convolution operation is introduced to obtain the convolution feature map vectors. Secondly, these vectors are segmented according to the positions of transition words in sentences. Thirdly, the most significant feature of each local segment is extracted using max pooling mechanism, and then the different aspects of features can be extracted. Specifically, the relative sequence of these features is preserved. Finally, after processed by the dropout algorithm, the softmax classifier is trained for sentiment classification. Experimental results show that the proposed method PPCNN is effective and superior to other baseline methods, especially for datasets with transition sentences.

Keywords: Sentiment classification, convolutional neural network, piecewise pooling, feature extract.

1 Introduction

Sentiment classification, also called sentiment analysis or opinion mining, is to study people's opinions, sentiments, evaluations, attitudes and sentiment from text and reviews [Liu and Zhang (2012)], which is an important task in natural language processing (NLP). With the successful application of deep learning in visual and speech recognition, some researchers have applied deep learning models such as recurrent neural networks (RNN) [Yoav (2016); Socher, Pennington, Huang et al. (2011); Sutskever, Vinyals and Le (2014); McCann, Bradbury, Xiong et al. (2017); Li, Luong, Jurafsky et al. (2015); Socher, Perelygin, Wu et al. (2013)] and convolutional neural networks (CNN) [Kim (2014); Kalchbrenner, Grefenstette and Blunsom (2014); Zeng, Liu, Lai et al. (2014); Johnson and Zhang (2015); Yin and Schütze (2016); Wang, Xu, Xu et al. (2015); Soujanya, Erik

[1] School of Computer Science and Information Engineering, Hefei University of Technology, 485 Danxia Road, Shushan District, Hefei, 230009, China.

[2] School of Computing and Informatics, University of Louisiana at Lafayette, 222 James R. Oliver Hall, Lafayette, Louisiana 70504, USA.

[*] Corresponding Author: Yuhong Zhang. Email: zhangyh@hfut.edu.cn.

and Alexander (2016)] to address the data sparseness in sentiment classification and get a better performance. Compared with RNNs, CNNs have attracted more attention because it can capture better semantics and it is easier to train with fewer tags, fewer connections and fewer parameters.

Recently, CNNs have shown to be effective in capturing syntax and semantics of words in sentences. CNNs [Kim (2014); Zeng, Liu, Lai et al. (2014); Wang, Xu, Xu et al. (2015); Hu, Lu, Li et al. (2014)] usually take a max pooling mechanism to capture the most useful feature of a sentence. Dynamic CNNs [Kalchbrenner, Grefenstette and Blunsom (2014); Yin and Schütze (2016)] use a dynamic k -max pooling operation for semantic modeling of sentences, and it can extract the k most useful features for sentiment classification. However, in practice, people are accustomed to express both positive and negative opinions connected by transitional words [Tang, Qin and Liu (2016); Vasileios and Kathleen (1997)]. In fact, statistically the number of the transition sentences takes a large proportion of about 40% in several review benchmark datasets. Therefore, the classification of the transition sentences has a great impact on the overall classification accuracy.

Most existing convolution neural networks adopt max pooling or k -max pooling to deal with transition sentences. However, this makes it difficult to capture both of the positive and negative features. For example, "beautifully filmed, talented actor and acted well, but admittedly problematic in its narrative specifics." Max pooling based CNN models only extract one feature "well" in affirmative acting, while omitting the feature "problematic" on the script which determines the sentiment orientation of this sentence. In contrast, k -max pooling based CNN models based on k -max pooling can extract three aspects features "well", "talented" and "beautifully". However, the three features extracted are the positive aspect for the filming and the performance of the actor, while the negative information on screenplay is absent.

In this paper, a piecewise pooling technology is introduced into CNN, and it forms our Piecewise Pooling Convolutional Neural Network (PPCNN). More specifically, with a transition word database, feature mapping vector is segmented, and then the most significant feature of each local segment is extracted using the max pooling mechanism. This not only extracts local significant features with different sentiment polarities, but also preserves the relative word sequence of these features.

Our contributions of this paper are as follows:

1. The text is represented with word embedding as the input of CNN, which does not require a complicated NLP preprocessing.

2. A piecewise pooling mechanism in CNN is proposed for sentiment classification on transition sentences, which can get multiple features with different sentiment polarities, and can also maintain the relative sequence of words in a sentence.

The remainder of this paper is organized as follows. Section 2 briefly reviews related work about RNN and CNN. Section 3 gives the details of our proposed PPCNN method. Section 4 shows the effectiveness of our proposed method experimentally. Section 5 summarizes the paper.

2 Related work

Deep learning models have been successfully applied in the fields of computer vision [Krizhevsky, Sutskever and Hinton (2012)] and speech recognition [Graves, Mohamed and Hinton (2013); Kim, Hori and Watanabe (2017)]. In the field of sentiment analysis, researchers have adopted deep learning models to learn better feature representations. These models fall into two categories: Sequence-based recursive neural network model [Socher, Pennington, Huang et al. (2011); Sutskever, Vinyals and Le (2014); McCann, Bradbury, Xiong et al. (2017); Li, Luong, Jurafsky et al. (2015)] and convolutional neural network model [Kalchbrenner, Grefenstette and Blunsom (2014); Zeng, Liu, Lai et al. (2014); Johnson and Zhang (2015); Yin and Schütze (2016); Wang, Xu, Xu et al. (2015); Soujanya, Erik and Alexander (2016)].

2.1 Recursive neural network model

Based on RNN model, Richard et al. [Socher, Pennington, Huang et al. (2011)] and Sutskever et al. [Sutskever, Vinyals and Le (2014)] proposed a semi-supervised recursive automatic encoder and a recursive neural tensor network to analyze the sentiment of sentences separately. Ramy et al. [Ramy, Hazem, Nizar et al. (2017)] created an Arabic Sentiment Treebank (ARSENTB) to explore different morphological and orthographical features at multiple levels of abstract. Kai et al. [Kai, Socher and Christopher (2015)] combined the LSTM networks with strong retention capabilities for time-series information to construct a tree-structure LSTM network model. This model outperformed other LSTM baselines on predicting the semantic relevancy between two different sentences and sentiment classification. Generally, RNNs require a lot of manual tagged words, phrases and sentences.

2.2 Convolutional neural network model

Compared with RNN, CNN is easy to be trained and requires fewer parameters and sentence-level tags. The standard CNN usually consists of an input layer, a convolution layer, a pooling layer and an output layer. In the input layer, each word is represented by a real-valued vector. Convolution layer is to learn and extract features. Pooling layer is to select features that have the strongest relevance to the task. Output layer is to classify, in which softmax classifier is usually adopted.

Kim [Kim (2014)] proposed a simple and improved CNN, whose input layer toke both task-specific and static word vectors for sentiment analysis and classification. Kalchbrenner et al. [Kalchbrenner, Grefenstette and Blunsom (2014)] introduced a dynamic convolutional neural network (DCNN), and a dynamic k -Max pooling operation was used as a nonlinear sampling function to dynamically adjust the extracted important features (k values) to accomplish sentiment classification without the requirement of parsers and other external features. Zeng et al. [Zeng, Liu, Lai et al. (2014)] established a deep convolutional neural network (DNN), which extracted the vocabulary and sentence-level features to classify. DNN also introduced the relationship label of the noun pair as a position feature into the network. Yin et al. [Yin and Schütze (2016)] proposed a multi-channel variable-size convolution neural network model (MVCNN) for sentiment classification on sentences and subjectivity classification, where

"MV" indicated that texts were initialized with five-word-vector training methods such as word2vec and Glove, and a variable-size convolution filter was applied to extract the features of sentences with various ranges.

3 Our proposed approach PPCNN

Aiming to improve the sentiment classification for a large number of transition sentences, this paper proposes a novel piecewise pooling convolution neural network, namely PPCNN. In this model, firstly, a sentence is represented with word embedding, and convolution operation is applied to obtain a feature mapping vector. Then this vector is segmented according to the positions of transition words in the sentence, so each important local feature is extracted in each fragment to capture the sentiment of the sentence. Finally, the features captured from all segments are used to train a classifier. Fig. 1 shows the architecture of our piecewise pooling neural network for text sentiment classification. Generally, the whole framework includes four parts: Data representation, convolution operation, piecewise pooling, and softmax output. We will describe these components in detail.

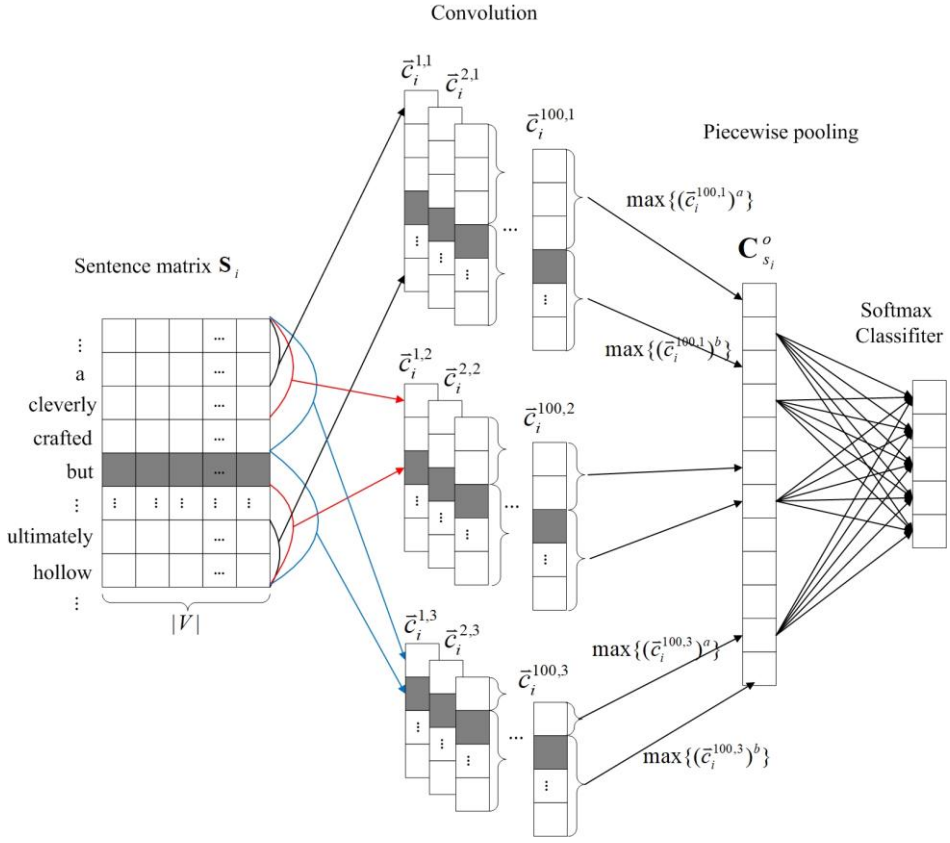

Figure 1: Frame of piecewise pooling CNN

3.1 Representation for input data

At present, there are many works about word embedding [Mikolov, Sutskever, Chen et al. (2013); Thang, Richard and Christopher (2013)]. These works point out that word vectors learned in large-scale unsupervised corpus can obtain more sematic information of words. In this paper, Google's word embedding tool, googlenews-vecctors-negative 300 billion, is adopted to train word vectors based on a corpus containing about 100 billion words. Given the training data $DS = \{s_1, s_2, \cdots s_i \cdots s_{|I|}\}$, s_i is a sample containing M words that can be represented as a two-dimensional matrix, as shown in Eq. (1).

$$S_i = \{\vec{s}_1^i, \vec{s}_2^i, \cdots, \vec{s}_m^i, \cdots, \vec{s}_M^i\} \tag{1}$$

Where $\vec{s}_m^i$ means the $|V|$-dimensional embedding word vector of the m-th word in s_i, and S_i is a $M \times |V|$ matrix.

3.2 Convolution operation

In this section, a convolution kernel and an input matrix S_i are convoluted to obtain the feature mapping vector $\vec{C}_i$ of a sample s_i. In order to capture richer features, convolution kernels of $K(K>1)$ sizes are employed, and for each size there are G convolution kernels. Then there are $K \times G$ kernels in total, and we can get $K \times G$ mapping vectors for the whole sentence, donated as $\vec{C}_i = \left\{\left[\vec{c}_i^{1,1}, \cdots, \vec{c}_i^{G,1}\right], \left[\vec{c}_i^{1,2}, \cdots, \vec{c}_i^{G,2}\right], \left[\cdots, \vec{c}_i^{g,k}, \cdots\right], \left[\vec{c}_i^{1,K}, \cdots, \vec{c}_i^{G,K}\right]\right\}$, in which $\vec{c}_i^{g,k}$ represents a mapping vector computed with the g-th kernel for the k-th size. Now, we illustrate the process of convolution in detail.

Firstly, a given size of convolution kernel $\mathbf{W}_g^k \in \mathbf{R}^{h_k \times n_k}$ is initialized randomly, in which, $h_k \times n_k$ is the size of the convolution kernel. Then, taking $h_k \times n_k$ as the size of the sliding window, convolution operation is performed on the entire sentence S_i with the narrow convolution method [Kalchbrenner, Grefenstette and Blunsom (2014)]. Therefore, a single feature mapping vector $\vec{c}^{g,k} = \left\{c_1^{g,k}, c_2^{g,k}, \cdots, c_\tau^{g,k}, \cdots c_{M-h_k+1}^{g,k}\right\}$ is obtained, and each sliding window is convoluted according to Eq. (2).

$$c_\tau^{g,k} = \sigma(\mathbf{W}_g^k \otimes \mathbf{S}_{\tau:\tau+h-1} + \mathbf{b}^c) \tag{2}$$

Where $\mathbf{S}_{\tau:\tau+h-1}^i$ refers to the segment of τ to $\tau+h-1$ in matrix S_i; $1 \le \tau \le m-h+1$; $\mathbf{b}^c \in \mathbf{R}^{M-h_k+1}$ is a bias term; $\sigma(\cdot)$ is the relu activity function; And $\otimes$ is the convolution operation.

3.3 Piecewise pooling

Traditional convolution neural network [Kim (2014); Wang, Xu, Xu et al. (2015)] takes the max pooling mechanism to map the input sentences of variable lengths into the same dimensional representation. In order to express the meanings of the text better, some

works [Kalchbrenner, Grefenstette and Blunsom (2014); Yin and Schütze (2016)] used k-max pooling operation to extract k important features from each feature map. Based on the positions of transition words, a piecewise pooling is proposed to extract the important features of each segment in this section.

Firstly, locate the position of transition word z_L in original input sentence S_i, donated as P_L^i. Secondly, according to the size of the convolution kernel $\mathbf{W}_g^k \in \mathbf{R}^{h_k \times n_k}$, we can obtain that the location of transition word in the mapping feature vector $\vec{c}^{g,k}$ is $P_L^i - h_k + 1$, which is treated as a cutting point. As shown in Fig. 1, the transition word "but" is the 5-th word in sentence. Therefore, the feature map vector $\vec{c}^{g,k}$ can be divided into two segments: $\left(\vec{c}^{g,k}\right)^a = \left\{c_1^{g,k}, c_2^{g,k}, \cdots, c_{P_L^i - h_k + 1}^{g,k}\right\}$ and $\left(\vec{c}^{g,k}\right)^b = \left\{c_{P_L^i - h_k + 2}^{g,k}, \cdots, c_{M - h_k + 1}^{g,k}\right\}$.

Then, max pooling method is applied to each segment separately. And the max value of each segment, donated as $\left(c_{\max}^{g,k}\right)^a$ and $\left(c_{\max}^{g,k}\right)^b$ respectively, represents the most important information of each segment, as shown in Eq. (3) and Eq. (4).

$$\left(c_{\max}^{g,k}\right)^a = \max\left\{c_1^{g,k}, c_2^{g,k}, \cdots, c_{P_L^i - h_k + 1}^{g,k}\right\} \tag{3}$$

$$\left(c_{\max}^{g,k}\right)^b = \max\left\{c_{P_L^i - h_k + 2}^{g,k}, \cdots, c_{M - h_k + 1}^{g,k}\right\} \tag{4}$$

It is necessary to mention that different sizes of convolution kernels indicate the differences of cutting points. In this paper, instead of selecting the most reasonable point, we use different sizes of convolution kernels to capture richer pooing features, which will benefit the classification.

Finally, the feature maps of all $K \times G$ convolution kernels are respectively segmented and pooled to obtain the final output, donated as

$$\left\{\left(\left(c_{\max}^{g,k}\right)^a, \left(c_{\max}^{g,k}\right)^b\right) \middle| g = 1, 2, \cdots, G; k = 1, 2, \cdots, K\right\}.$$

In this way, we can extract the most important information in each segment based on the position of the transition word in one sample, and finally the feature vectors $\mathbf{C}_{s_i}^o \in \mathbf{R}^{2 \times K \times G}$ can be obtained. In order to avoid over-fitting and improve the prediction accuracy, the dropout algorithm [Kim (2014)] is applied to randomly set the input data to 0 according to a certain probability, and only the preserved elements are passed through the whole network to train softmax classifier.

In fact, the positions of transition words vary in different sentences, which means that the segment cannot keep balance between two parts of a whole sentence. Our proposed PPCNN method will extract one important feature from each segment regardless of the length of segment. Therefore, the position of transition words will not influence the performance of our proposed approach. Moreover, when there is no transition word, our proposed approach will perform as same as the one proposed in Richard et al. [Socher, Perelygin, Wu et al. (2013)].

4 Experiment results

4.1 Data sets

In this section, we compared our proposed PPCNN with 10 relevant algorithms on 6 benchmark datasets to prove the superiority of our proposed algorithm. The details of these 6 datasets are illustrated as follows, with the statistical summary of shown in Tab. 1.

MR: In this data set of movie reviews, there is one sentence for each review. Classification process involves detecting positive/negative reviews. There are 10662 reviews totally, in which positive/negative emotions own an equal weight. There are 4647 transition samples in this dataset, with the average length of samples 20. And the dataset is available at: https://www.cs.cornell.edu/people/pabo/movie-review-data/.

Table 1: Details of data sets

Data	#label	#AveLen	#Samples	#TransiSent	#features
MR	2	20	10662	4647	18765
SST-1	5	18	11855	4762	17836
SST-2	2	19	9613	3729	16185
Subj	2	23	10000	4411	21323
CR	2	19	3775	1489	5340
MPQA	2	3	10606	461	6246

SST-1: SST-1 is also called Stanford Sentiment Treebank, and it is an extension of MR but with five labels including very positive, positive, neutral, negative, and very negative. SST-1 can be obtained from: http://nlp.stanford.edu/sentiment/.

SST-2: SST-2 is the same as SST-1 but with neutral reviews removed and all reviews are converted to binary labels.

Subj: Subj is a dataset, and the task is to classify a sentence as being subjective or objective. There are 10000 samples in this dataset, including 4411 transition samples, and the average length of samples is 23. More details can refer to Wang et al. [Wang and Christopher (2012)].

CR: CR is a dataset about customer reviews of various products, including cameras, MP3s, etc., and the task is to predict positive/negative reviews. There are 1489 transition samples in this dataset. For more details, please refer to Wang et al. [Wang and Christopher (2012)].

MPQA: MPQA is a dataset for opinion polarity detection, and there are 10606 samples including 461 transition samples. For more details, please refer to http://www.cs.pitt.edu/mpqa/.

4.2 Baselines and parameters

To demonstrate the effectiveness of our proposed model, four categories methods, including 10 algorithms are used as the baselines, whose details are as follows.

1) **Traditional classifiers with Bag of words:** NB and SVM are the traditional classifiers using the bag of words method.

2) **Traditional classifiers wit unigram and bigram:** BiNB, NBSVM and MNB also use the traditional classifiers NB and SVM. Especially, BiNB trains NB classifier with unigram and bigram features, and NBSVM and MNB train Naive Bayes SVM and Multinomial Naive Bayes with uni-bigrams [Wang and Christopher (2012)].

3) **RNNs:** RAE, MV-RNN and RNTN are models based on the RNN and use a fully-labeled parser to parse the vector representation of tree learning phrases and complete sentences. RAE [Socher, Pennington, Huang et al. (2011)] adopts Recursive Auto Encoders with pre-trained word vectors from Wikipedia. MV-RNN [Richard, Brody, Christopher et al. (2012)] is a Matrix-Vector Recursive Neural Network with parse trees. In contrast, RNTN [Socher, Perelygin, Wu et al. (2013)] adopts a Recursive Neural Tensor Network with tensor-based feature functions and parse trees.

4) **CNNs:** Both DCNN and CNN are based on the CNN model. CNN [Kim (2014)] is a Convolutional Neural Network with max pooling, while DCNN [Kalchbrenner, Grefenstette and Blunsom (2014)] is a Dynamic Convolutional Neural Network with k -max pooling.

Since the advantages of multiple sizes of convolution kernels have been demonstrated in existing works, we adopt three sizes of filter windows of $3\times|V|$, $4\times|V|$, and $5\times|V|$ ($|V|=300$), and each size includes 100 kernels as the settings in these works [Kim (2014); Kalchbrenner, Grefenstette and Blunsom (2014)]. Meanwhile, other parameters are kept the same as those in Kim et al. [Kim (2014)], such as: A dropout rate is set to 0.5, an L2 constraint is set to 3, and a mini-batch size is set to 50. The classification accuracy averaged over 10 cross-validations will be reported in the following subsection.

The word corpus trained on the news corpus of Google is utilized to initialize the experimental data in this paper.

We incorporate Smart Words (http://www.smart-words.org/linking-words/transition-words.html) and MSU (https://msu.edu/user/jdowell/135/transw.html) to get a transition word corpus, which includes 179 transition words in total. This transition word corpus is utilized to locate probable transitions in each sentence.

4.3 Classification performance

We compare our proposed PPCNN with the baselines, and the classification accuracies of all methods are shown in Tab. 2. We have notice that there are some missing values in Tab. 2. On the one hand, for Subj and CR data sets, the accuracies of RNN models are not included because the data sets have not the phrase tag information. On the other hand, as for other missing values in Tab. 2, the results are not included because these data sets cannot run in the open source code.

Compared with traditional methods (such as NB, SVM, etc.), the classification

performance based on neural networks (including RAE, MV-RNN, RNTN, DCNN, CNN, and PPCNN) have an improvement by a range [3.4%, 6.7%]. It indicates that neural network models can obtain more valuable context information, relieve the data sparseness and explore the semantic information of texts more effectively.

Table 2: Classification accuracy of all algorithms (%)

Algorithms	MR	SST-1	SST-2	MPQA	Subj	CR
NB	78.6	41	81.8	86.9	92.3	81
SVM	77.7	40.7	79.4	86.7	91.7	80.8
BINB	78.9	41.9	83.1	86.5	92.8	81.4
NBSVM	79.4	42.1	83.5	86.3	93.2	81.8
MNB	79	42.3	83	86.3	**93.6**	80
RAE	77.7	43.2	82.4	86.4	-	-
MV-RNN	79	44.4	82.9	-	-	-
RNTN	80.6	45.7	84.4	-	-	-
DCNN	80.3	46.4	84.1	-	-	-
CNN	80.2	45.8	84.2	**89.5**	93	83.9
PPCNN	**81.1**	**47.4**	**85.2**	**89.5**	93.4	**84.5**

Compared with the RNNs (including RAE, MV-RNN, and RNTN), convolution-based models (including CNN, DCNN and PPCNN) are more suitable for representing the semantic of texts. And classification accuracies of CNNs are improved by [2.8%, 4.2%] on 6 datasets. This is due to the fact that CNNs can select richer and more important features in pooling layer and capture the contextual information in convolution layer. In contrast, RNNs can only capture contextual information using semantic combinations of constructed text trees, which heavily depends on the performance of tree construction. In addition, RNNs cost $O(n^2)$ time to represent the sentence, whereas CNNs only cost $O(n)$, where n means the length of the text.

Compared with other CNNS, our proposed CNN based PPCNN improves the classification accuracies by [0.6%, 1.6%] on MR, SST-1, SST-2 and CR datasets, because both positive and negative sentiments exist in these four kinds of comment texts, which are interrupted by transition words. Traditional CNN and DCNN ignore local important features, which may lead to an incorrect final sentiment label. However, our PPCNN algorithm extracts multiple features with different sentiment from multiple segments, and all these features are useful in sentiment classification.

As for MPQA and Subj datasets, our proposed PPCNN method is superior to RAE and traditional bag-of-words based methods on accuracy performance, but it is not prior to other baselines. This is due to the fact that MPQA dataset is very short (an average length of sentence is 3), and the advantage of dividing the sentence according to transition word cannot be reflected. As a result, our method performs similarly to other CNN methods. In addition, Subj dataset is to determine the text whether subjective evaluation or objective factual summary, so the results have nothing to do with transition words.

4.4 Effectiveness of piecewise pooling

Table 3: Examples of captured features by CNN, DCNN and PPCNN on MR dataset

Label	SENTENCES	CNN	DCNN	PPCNN
Negative	It is Tommy's job to clean the peep booths...a cleverly crafted **but** ultimately hollow mockumentary.	cleverly	cleverly, clean, peep	cleverly, hollow
	Like its title character, Esther kahn is unusual **unfortunately** also irritating	unusual	unusual, Like, irritating	unusual, irritating
	This likable movie is not successful, **although** the actors are appealing and hard **but** too amateurish and awkward.	appealing	appealing, hard, amateurish	is not successful, appealing, awkward
Positive	It has charm to spare, unlike many romantic comedies cliche, it does not alienate either gender in the audience.	cliche	cliche, romantic, not	charm, cliche
	There is not a fresh idea at core of tale, the version' no classic, **but** its pleasures are still plentiful.	no classic	no classic, not a fresh, tale	no classic, pleasures
	The film is funny, **despite** you are depressed and angry, **but** you will be entertained as well.	Angry	angry, depressed, funny	funny, angry, entertained

In this subsection, we validate the effectiveness of piecewise pooling from two aspects: The quality of extracted features and the classification accuracy. Firstly, we examine the completeness of the captured features based on piecewise pooling. Tab. 3 compares the features extracted by CNN, DCNN and PPCNN on MR dataset.

In sentences with the transition word, the sentiment polarity will turn from the positive (negative) to negative (positive). However, CNN can extract only one feature, while DCNN can extract k features according to frequency or position. Taking the first review in Tab. 3 as an example, CNN extracts only one feature "cleverly", and DCNN extracts two positive features "cleverly", "clean" and a negative feature "peep". Both may predict the label incorrectly. And our proposed PPCNN can extract "cleverly" from one segment and "hollow" from another segment according to the transition word, and it has larger probability to predict the label correctly.

It can also be seen that the positions of transition words vary, for example, some are balance (such as example 1, 3 and 6 in Tab. 3) and some are not (such as example 2, 4 and 5 in Tab. 3). It can be concluded that the positions of transition words will not influence the performance of our proposed PPCNN. Especially, when the transition words are not centered, baselines may neglect the smaller part, while our algorithm will not. As a result, our PPCNN will extract richer sentimental features.

Table 4: Comparison results on a Transition subset and a non-Transition subset (%)

Models	SST-1（11855）		SST-2（9613）		CR（3775）	
	Transi（4762）	NoTransi（7093）	Transi（3729）	NoTransi（5884）	Transi（1489）	NoTransi（2286）
DCNN	41.1	48.8	79.6	85.2	74.6	85
CNN	40.9	48.2	79.3	85.7	73.7	85.2
PPCNN	42.3	48.6	80.2	85.9	76.8	85.1

In addition, we show the effectiveness of the piecewise pooling in term of the accuracy. With the dataset divided into two subsets according to the fact whether one sample contains transition words, we have a transition subset (i.e. Transi in Tab. 4) and a non-transition subset (i.e. NoTransi in Tab. 4). In Tab. 4, the SST-1 dataset is divided into a transition subset (with the size of 4762) and a non-transition subset (with the size of 7093). In Tab. 4, 41.1% is the accuracy of DCNN trained and tested on the transition subsets using 10-fold cross-validation.

As shown in Tab. 4, the classification accuracies of three methods on Transi subsets are significantly lower than those on NoTransi subsets, which reveals that the classification for transition sentence is challenging. In the three subsets with transition words, our proposed PPCNN performs best with an improvement by [0.6%, 3.1%] compared with DCNN and CNN. It shows that piecewise pooling can capture more representative features for transition sentences. Additionally, when there is no transition word in sentence, our PPCNN will not divide the sentence, therefore, PPCNN performs as same as CNN and DCNN, as shown in Tab. 4 on NoTransi subsets.

5 Conclusion

Transition sentences in the real application make sentiment classification a challenging and attractive task. This paper focuses on the transition sentences and proposes a piecewise pooling convolution neural network (PPCNN). For common texts with transitional semantics, we can capture important local features from multiple segments of sentences. Experimental results show that the proposed model is superior to the current convolutional neural network models on four public customer comment datasets. In the near future, we tend to represent the input data in chunk vector [Yan, Zheng, Zhang et al. (2017)] to address the sentiment classification to improve the efficiency.

Acknowledgement: This work is supported in part by the Natural Science Foundation of China under grants (61503112, 61673152 and 61503116).

References

Graves, A.; Mohamed, A.; Hinton, G. (2013): Speech recognition with deep recurrent neural networks. *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6645-6649.

Hu, B.; Lu, Z.; Li, H.; Chen, Q. (2014): Convolutional neural network architectures for matching natural language sentences. *Advances in Neural Information Processing Systems*, pp. 2042-2050.

Johnson, R.; Zhang, T. (2015): Effective use of word sequence for text categorization with convolutional neural networks.

http://www.oalib.com/paper/4067541#.WusLlOQh02w.

Kai, S. T.; Richard, S.; Christopher, D. M. (2015): Improved semantic Representations from tree-structured long short-term memory networks. *Proceedings of the 53nd Annual Meeting of the Association for Computational Linguistics*, pp. 1556-1566.

Kalchbrenner, N.; Grefenstette, E.; Blunsom, P. (2014): A convolutional neural network for modelling sentences. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, vol. 1, pp. 655-665.

Kim, Y. (2014): Convolutional neural networks for sentence classification.

http://aclweb.org/anthology/D/D14/D14-1181.pdf.

Kim, S.; Hori, T.; Watanabe, S. (2017): Joint CTC-attention based end-to-end speech recognition using multi-task learning. *IEEE International Conference on Acoustics Speech and Signal Processing*, pp. 4835-4839.

Krizhevsky, A.; Sutskever, I.; Hinton, G. E. (2012): ImageNet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, pp. 1106-1114.

Li, J.; Luong, M. T.; Jurafsky, D.; Hovy, E. (2015): When are tree structures necessary for deep learning of representations? *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 2304-2314.

Liu, B.; Zhang, L. (2012): *A Survey of Opinion Mining and Sentiment Analysis*. Springer, US.

McCann, B.; Bradbury, J.; Xiong, C.; Socher, R. (2017): Learned in translation: Contextualized word vectors. *Advances in Neural Information Processing Systems*, pp. 6297-6308.

Mikolov, T.; Sutskever, I.; Chen, K.; Gregory, S. C.; Jeffrey, D. (2013): Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, pp. 3111-3119.

Ramy, B.; Hazem, M. H.; Nizar, H.; Khaled, B. S.; Wassim, E. (2017): A sentiment treebank and morphologically enriched recursive deep models for effective sentiment analysis in arabic. *ACM Transactions on Asian and Low-Resource Language Information Processing*, vol. 16, no. 4, pp. 1-21.

Socher, R.; Huval, B.; Manning, C. D.; Ng, A. Y. (2012): Semantic compositionality through recursive matrix-vector spaces. *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 1201-1211.

Socher, R.; Pennington, J.; Huang, E. H.; Ng, A. Y.; Manning, C. D. (2011): Semi-supervised recursive autoencoders for predicting sentiment distributions. *EMNLP'11*

Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 151-161.

Socher, R.; Perelygin, A.; Wu, J. Y.; Chuang, J.; Manning, C. D. (2013): Recursive deep models for semantic compositionality over a sentiment treebank. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631-1642.

Soujanya, P.; Erik, C.; Alexander, F. G. (2016): Aspect extraction for opinion mining with a deep convolutional neural network. *Knowledge-Based Systems*, vol. 108, pp. 42-49.

Sutskever, I.; Vinyals, O.; Le, Q. V. (2014): Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*, pp. 3104-3112.

Tang, D.; Qin, B.; Liu, T. (2016): Aspect level sentiment classification with deep memory network. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 214-224.

Thang, L.; Richard, S.; Christopher, D. M. (2013): Better word representations with recursive neural networks for morphology. *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pp. 104-113.

Vasileios, H.; Kathleen, M. (1997): Predicting the semantic orientation of adjectives. *Proceedings of the Eighth Conference on European Chapter of the Association for Computational Linguistics*, pp. 174-181.

Wang, P.; Xu, J.; Xu, B.; Liu, C.; Zhang, H. et al. (2015): Semantic clustering and convolutional neural network for short text categorization. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, vol. 2, pp. 352-357.

Wang, S.; Christopher, D. M. (2012): Baselines and bigrams: simple, good sentiment and topic classification. *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, vol. 2, pp. 90-94.

Yan, L.; Zheng, W.; Zhang, H.; Tao, H.; He, M. (2017): Learning discriminative sentiment chunk vectors for twitter sentiment analysis. *Journal of Internet Technology*, vol. 18, no. 7, pp. 1605-1613.

Yin, W.; Schütze, H. (2016): Multichannel variable-size convolution for sentence classification. https://arxiv.org/abs/1603.04513.

Yoav, G. (2016): A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, vol. 57, pp. 345-420.

Zeng, D.; Liu, K.; Lai, S.; Zhou, G.; Zhao, J. (2014): Relation classification via convolutional deep neural network. *25th International Conference on Computational Linguistics*, pp. 2335-2344.