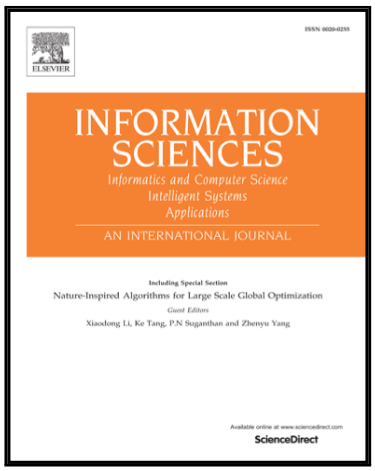


Attention Pooling-based Convolutional Neural Network for Sentence Modelling

Yong Zhang, Meng Joo Er, Ning Wang, Mahardhika Pratama

PII: S0020-0255(16)30667-3
DOI: 10.1016/j.ins.2016.08.084
Reference: INS 12487

To appear in: *Information Sciences*

Received date: 27 May 2016
Revised date: 1 August 2016
Accepted date: 26 August 2016

Please cite this article as: Yong Zhang, Meng Joo Er, Ning Wang, Mahardhika Pratama, Attention Pooling-based Convolutional Neural Network for Sentence Modelling, *Information Sciences* (2016), doi: 10.1016/j.ins.2016.08.084

Attention Pooling-based Convolutional Neural Network for Sentence Modelling

Yong Zhang[a], Meng Joo Er[a,*], Ning Wang[b], Mahardhika Pratama[c]

[a]*School of Electrical and Electronic Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798*
[b]*Marine Engineering College, Dalian Maritime University, Dalian 116026, China*
[c]*Department of Computer Science and IT, La Trobe university, Melbourne, Victoria 3086, Australia*

Abstract

Convolutional neural network has been proven to be a powerful semantic composition model for modelling sentences. A standard convolutional neural network usually consists of several convolutional and pooling layers at the bottom of a linear or non-linear classifier. In this paper, a new pooling scheme termed *Attention Pooling* is proposed to retain the most significant information at the pooling stage. An intermediate sentence representation generated by the bidirectional long short-term memory is used as a reference for local representations produced by the convolutional layer to obtain attention weights. The sentence representation is formed by combining local representations using obtained attention weights. The intermediate sentence representation is used as an input to the top classifier as well in the testing phase. The salient features of the proposed attention pooling-based convo-

*Corresponding author
Email addresses: yzhang067@e.ntu.edu.sg (Yong Zhang), emjer@ntu.edu.sg (Meng Joo Er), n.wang.dmu.cn@gmail.com (Ning Wang), m.pratama@latrobe.edu.au (Mahardhika Pratama)

lutional neural network are: (1) The model can be trained end-to-end with limited hyper-parameters; (2) Comprehensive information is extracted by the new pooling scheme and the combination of the convolutional layer and the bidirectional long-short term memory; (3) The model can implicitly separate the sentences from different classes. Experimental results demonstrate that the new model outperforms the state-of-the-art approaches on seven benchmark datasets for text classification. The learning capability of the proposed method is greatly improved and the classification accuracy is even enhanced significantly by over 2% on some datasets. The robustness of the proposed model is evidenced by some statistical tests.

Keywords: Deep Learning, Convolutional Neural Network (CNN), Long-short term memory (LSTM), Sentence Modelling, Text Classification.

1. Introduction

The sentence modelling problem is at the core of the field of natural language processing (NLP) and has received a great deal of attention recently. The main objective of sentence modelling is to learn representations of sentences which are inputs of tasks like sentiment analysis, document summarization, machine translation, discourse analysis, etc. Feature representation is a key component of many machine learning systems because the performance of a machine learner depends heavily on it [17]. Sentence features are usually extracted by performing composition over features of words or n-grams. This conforms to the principle of compositionality which claims that the meaning of a longer expression (e.g. a sentence or a document) is determined by its constituents and the combination rules [7].

With advances in deep learning techniques, distributed vectorial representation has become a common practice for word representation. Word vector representations are mostly learned through neural language models [3, 22, 23, 27, 35]. In neural language models, each word is represented by a dense vector and can be predicted by or used to predict its context representations. Word embedding[1] has drawn great attention in recent years because it can capture both semantic and syntactic information. In the vector space, words with similar semantics lie close to each other, for example, *vec(American)* is closer to *vec(France)* than to *vec(Bread)*. Vector representations of words can even preserve the semantic relationship. Word embedding turns out to be quite effective in many NLP applications such as parsing [30], tagging [12], name entity recognition [6], and machine translation [42].

The simplest sentence modelling method based on word embedding may be the continuous bag-of-words model (cBoW) which employs max-pooling or average-pooling over representations of all words in a sentence to compose the representation of that sentence. The model achieves very good performance on a variety of tasks but faces the problem of losing word order which is critical for semantic analysis. A number of other neural sentence models have been proposed to capture the word order. They leverage on neural networks to construct non-linear interactions between words so as to learn sentence representations which can well capture the semantics of sentences.

The recursive neural networks of [30, 31, 33] rely on parse trees to compose word vectors into sentence vectors. The composition procedure is recursively

[1]Word embedding and word vector appear alternatively in this paper. They have the same meaning.

applied to child nodes in the parse tree in a bottom-up manner to generate hidden representations of parent nodes until reaching the root of the tree, whose representation is the sentence representation. However, the learning performance of recursive neural networks depends heavily on the construction of the textual tree where the tree construction can be very time-consuming.

The recurrent neural networks of [8, 28] are special cases of the recursive neural networks which can only compose word vectors from one end to the other and therefore can be regarded as formed through time. The recurrent neural networks can handle variable-length sequences and model contextual information dynamically. However, they face the problem of "vanishing gradient" [10] which states that the gradients tend to either vanish or explode exponentially, making the gradient-based optimization method inefficient and ineffective.

Another method termed Paragraph Vector [16] is also very powerful. The method learns representations for sentences or paragraphs in the same way as learning word vectors using CBOW or Skip-grams [22, 23]. It is an unsupervised algorithm that learns fixed-length feature representations from variable-length texts. It achieves amazingly good performance on some tasks. However, other researchers find that it performs sub-optimally on other tasks.

Convolutional neural network (CNN), first proposed for computer vision tasks [18], has been proven to be a powerful semantic composition model for modelling sentences [13, 14, 15]. A standard CNN is usually constituted by several convolutional and pooling layers at the bottom of a linear or non-linear classifier. A pooling function is applied to the feature map obtained by each convolutional filter to reduce the spatial size of the vector representation

and induce a fixed length vector. Next, the feature vectors for all the filters are concatenated to form a single feature vector which is used as an input to the classifier.

All the existing pooling strategies discard information contained in the context to some extent, which may affect the semantic extraction procedure. In this paper, a new convolutional neural network model termed *Attention Pooling-based Convolutional Neural Network (APCNN)* is developed to address the problem. A new pooling scheme termed *Attention Pooling* is proposed to retain the most significant information at the pooling stage. The bidirectional long short-term memory (BLSTM) model is employed to enhance the information extraction capability of the pooling layer. The BLSTM model is also combined with the convolutional structure to extract comprehensive information, namely historical, future and local context information, of any position in a sequence at the testing phase. Therefore, the new model retains more information contained in the sentence and is able to generate a more representative feature vector for the sentence.

Our new model can be trained end-to-end with limited hyper-parameters and it is very easy to implement. We conduct experiments on several sentence classification tasks. Experiment results demonstrate that the new model outperforms state-of-the-art approaches on seven benchmark datasets. It should be highted that the absolute classification accuracy on Stanford Treebank Datasets is even significantly improved by over two percent. The new attention pooling scheme is shown to be more effective than existing pooling strategies. The proposed sentence model can even implicitly separate sentences from different classes in the semantic space which is a very powerful

ability for classification. In summary, the main contributions of this work are as follows:

- A new CNN model is developed and a new pooling scheme termed *Attention Pooling* is proposed to retrieve the most significant information at the pooling stage. The proposed model does not need external modules and can be trained end-to-end.
- The combination of the BLSTM model and convolutional structure enables the model to extract comprehensive information in sentences. This further improves the learning capacity because comprehensive context information is very significant for extracting semantics in sentences.
- Empirical results on text classification datasets demonstrate that the proposed model can achieve higher accuracy compared with state-of-the-art approaches.

This paper is organized as follows: Section 2 gives a brief review of related works. In Section 3, the proposed model is described in details. The performance of the proposed method is compared with state-of-the-art methods in Section 4. Conclusions are drawn in Section 5.

2. Related Works

Pooling is a significant component of the CNN. The pooling function can reduce the number of parameters in the model and thus alleviate the problem of over-fitting. Max pooling is the most widely used pooling operator

Table 1: Descriptions of related pooling strategy

Strategy	methods	dimension
1-max	$max([c_1, c_2, \cdots, c_T])$	1
local-max	$[max([c_1, c_{T_1}]), max([c_{T_1+1}, c_{T_2}]), \cdots, max([c_{T_{n-1}+1}, c_{T_n}])]$	n
k-max	$k - max([c_1, c_2, \cdots, c_T])$	k
average	$average([c_1, c_2, \cdots, c_T])$	1

which returns the maximum value of a set of values. The 1-max pooling is applied over the entire feature map, inducing a feature vector of length 1 for each convolutional filter. By only keeping the biggest value, max pooling can capture the most relevant feature. However, the max pooling has some disadvantages as well. It forgets position information of the features because only the maximum value is used in the pooling stage. The intensity information of the same feature is also lost because it cannot distinguish whether a feature occurs once or in multiple times.

Local max pooling [5] is applied over small local regions of the feature map, producing a number for each local region. Next, the numbers are concatenated to form the representation vector for the feature map. In [14], Kalchbrenner *et al.* proposed an effective pooling strategy termed k-max pooling which extracts the k maximum values of the feature map and preserves the relative order of these values. Both local max pooling and k-max pooling are able to preserve some position information and intensity information of features. It can be easily seen that 1-max pooling is a special case of either local max pooling or k-max pooling. Another popular pooling function is average pooling [4] which returns the average value of the feature maps. It was demonstrated in [41] that max pooling outperforms average

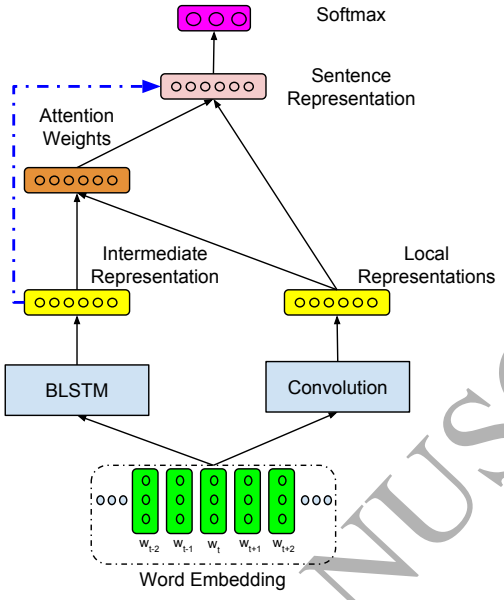


Figure 1: Architecture of APCNN. The BLSTM is incorporated in the pooling stage to obtain attention weights which are used to combine local representations (only one local representation is shown in the figure for conciseness) into the sentence representation. The dashed line indicates that the intermediate sentence representation will also be used as input to the softmax layer at the testing phase.

pooling in most applications . The descriptions of related pooling strategies are summarized in Table 1. All these pooling strategies discard information contained in the context to some extent, which may affect the semantic extraction procedure.

3. The Proposed Model

We describe the proposed model in detail in this section. Figure 1 depicts the structure of the new model. Convolutional filters perform convolutions on the input sentence matrix and generate local representations. The con-

volution operation can independently capture local information contained in every position of a sentence. An attention pooling layer is used to integrate local representations into the final sentence representation with attention weights. These weights are obtained by comparing local representations position by position with an intermediate sentence representation generated by the BLSTM [9, 10] and optimized during the training phase. At last, sentence representations of all distinct convolutional filters are concatenated into the final feature vector which is fed into a top-level softmax classifier. The intermediate sentence representation generated by the BLSTM will be also used as an input to the softmax classifier in the testing phase, which is indicated by the dashed-lines in Figure 1.

The two salient contributions, namely the new pooling scheme and the combination of the BLSTM model with convolutional structure, of the proposed model are described in detail in Section 3.3. The two novel components enable the model to extract comprehensive semantic information which is very important for good classification performance. The other necessary components of our method, namely word embedding (Section 3.1), convolution methods (Section 3.2), parallel CNNs (Section 3.4) and softmax classifier (Section 3.5) are also introduced so that the description is complete and comprehensive.

3.1. Word Embedding

The input of the algorithm is N variable-length sentences. Each sentence S is constituted by words which are represented by vectors. Traditional word representations, such as one-hot vectors, achieve good learning performance in the task of document classification [13]. However, one-hot vectors may face

the problem of curse of dimensionality when used to classify short sentences because of sparsity.

Recent research results have demonstrated that continuous word representations are more powerful. A word can be represented by a dense vector as follows:

$$x = Lw \tag{1}$$

where $w \in \mathbb{R}^V$ is a one-hot vector where the position that the word appears is one while the other positions are zeros, $L \in \mathbb{R}^{d \times V}$ is a word-representation matrix, in which the ith column is the vector representation of the ith word in the vocabulary, and V is the vocabulary size.

We can easily adopt off-the-shelf word embedding matrices as initial matrices to make better use of semantic and grammatical associations of words. Word2vec [22] and GloVe [27] are two most widely used pre-trained word embedding matrices. Previous research results demonstrate that different performance may result from using either matrix on different tasks, but with slight differences. In this paper, we use *word2vec*[2] embedding for all the datasets. The model is trained on 100 billion words from the Google News by using the Skip-gram method and maximizing the average log probability of all the words [22] as follows:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0} logp(x_{t+j}|x_t) \tag{2}$$

$$p(y|x) = \frac{exp(x_i^T y)}{\sum_{j=1}^{V} exp(x_j^T y)} \tag{3}$$

[2]https://code.google.com/p/word2vec

where c is the context window size. The values of word vectors are included in the parameters which are optimized during the training procedure.

3.2. Convolution Layer

Convolutional layers play critical roles in the success of the CNN because they can encode significant information contained in input data with significantly fewer parameters than other deep learning architectures. Empirical experiences in the area of computer vision suggest that deep architectures with multiple convolutional layers are necessary to achieve good performance [17]. However, only one convolutional layer was used to achieve state-of-the-art or comparable performance on several datasets of the sentence classification task in [15]. In some cases, the performance only increases marginally or even decreases because of over-fitting with increasing number of convolutional layers. Furthermore, the computation complexity increases quickly if more layers are used. In this paper, we also use only one convolution layer and carry out experiments on the same datasets as those of [15].

The convolution operation in our model is conducted in one dimension between k filters $W_c \in \mathbb{R}^{md \times k}$ and a concatenation vector $x_{i:i+m-1}$ which represents a window of m words starting from the ith word, obtaining features for the window of words in the corresponding feature maps. The term d is the dimension of word embedding. The parameters of each filter are shared across all the windows. Multiple filters with differently initialized weights are used to improve the model's learning capability. The number of filters k is determined using cross-validation and the convolution operation is governed by

$$c_i = g(W_c^T x_{i:i+m-1} + b_c) \in \mathbb{R}^k \tag{4}$$

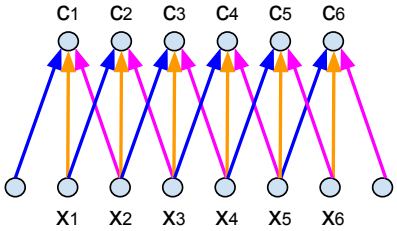


Figure 2: Illustration of 'same' convolution graph. The window size of the filter is 3 and weights are shared across different windows. The arrows with the same colors represent the same weight values. (Readers are referred to the web version of the article for a better interpretation)

where $x_i \in \mathbb{R}^d$, the term b_c is a bias vector and $g(\cdot)$ is a nonlinear activation function. The ReLU has become a standard nonlinear activation function of CNN recently because it can improve the learning dynamics of the network and significantly reduce the number of iterations required for convergence in deep learning networks. We employ a special version of the ReLU called LeakyReLU [20] that allows a small gradient when the unit is not active. It helps further improve the learning efficiency compared with ReLU.

Suppose the length of a sentence is T. We set the border mode of convolution as 'same' so that the output length of the convolution layer is the same as that of the input. This kind of convolution is shown in Figure 2. The border needs zero-paddings to guarantee the same length. As the word window slides, the feature maps of the convolutional layer can be represented as follows:

$$c = [c_1, c_2, \cdots, c_T] \in \mathbb{R}^{k \times T} \tag{5}$$

The output of the convolutional layer represents local representations of the sentence and each element c_i is a local representation of the corresponding

position.

3.3. Attention Pooling

Most existing convolutional neural networks take advantage of max pooling to reduce parameters and induce fixed length vector. However, max pooling loses position and intensity information of features which may deteriorate the learning performance. In this paper, an innovative pooling strategy termed *Attention Pooling* is proposed.

First, as shown in Figure 1, an intermediate sentence representation is needed. The intermediate representation is generated by the BLSTM [9, 10]. The BLSTM is a variant of the recurrent neural network which can learn both historical and future information contained in a sequence. It is also able to address the problem of "vanishing gradient" by replacing the hidden state of the recurrent neural network with a gated memory unit. With regard to the LSTM unit, we follow the implementation described in [39]. We denote the intermediate sentence representation as $\widetilde{s}$.

Once the intermediate sentence representation is generated, we can compare local representations generated by the convolutional layer with it to calculate the attention weights. In order to compare the two representations, we should map both local representation and intermediate sentence representation to the space of the same dimension. This can be achieved by controlling the output dimension of the BLSTM same as the number of convolutional filters k. The higher the similarity between the intermediate sentence representation and each local representation, the bigger attention weight is assigned to that local representation. The attention weights are

calculated as follows:

$$\alpha_i = \frac{exp(e_i)}{\sum_{i=1}^{T} exp(e_i)} \tag{6}$$

where

$$e_i = sim(c_i, \widetilde{s}) \tag{7}$$

The term α_i is a scalar and the function $sim(\cdot)$ is used to measure the similarity between its two inputs. Cosine similarity is used in our model. After the attention weights are obtained, the final sentence representation is given by:

$$s = \sum_{i=1}^{T} \alpha_i c_i \in \mathbb{R}^k \tag{8}$$

The BLSTM model is jointly trained with all the other components of the model. The gradients of the cost function back-propagate through the intermediate sentence representation so that it is optimized during the training phase. As such, no external modules are needed in our model and the model can be trained end-to-end. The intermediate sentence representation obtained by the BLSTM model will be concatenated with the sentence representation obtained by the convolutional structure to form the input of the top classifier in the testing phase.

The attention pooling can be regarded as taking a weighted sum of all the word annotations to compute the sentence annotation. The weight of each word measures how much the word contributes to the meaning of the entire sentence. This method borrows the idea of one very effective mechanism namely 'attention' used in recent years in various tasks, like machine translation [2], object recognition [1], and image captioning [38], etc. Intuitively, the model can decide which features to pay attention to. Compared with the

max pooling methods, the attention pooling method is able to reserve more information contained in the sentence. Compared with the average pooling method, the new pooling strategy has an obvious advantage by assigning bigger weights to more significant features.

The other import component of the new model is the combination of the BLSTM and the convolutional structure. The intermediate sentence representation generated by the BLSTM should already be a sufficiently satisfactory input representation for the top classifier. By comparing local representations with this intermediate sentence representation, the obtained attention weights encode richer information of the sentence. On the other hand, the local context extraction capacity of convolutional structure improves the information retrieval ability compared with just using the BLSTM. The proposed model is able to access comprehensive information, namely historical, future and local context of any position in a sequence. In summary, the position and intensity information of features are completely preserved with the help of the proposed innovative attention mechanism. This is what we aim to achieve at the pooling stage so as to overcome disadvantages of existing pooling strategies.

3.4. Parallel CNNs

The CNN architecture described so far is very simple with only one convolutional layer and one attention pooling layer. Following the approach in [13, 15], we also use filters with varying convolution window sizes to form parallel CNNs so that they can learn multiple types of embedding of local regions so as to complement each other to improve model accuracy. The convolutional layers with distinct window sizes are depicted in Figure 3. Sen-

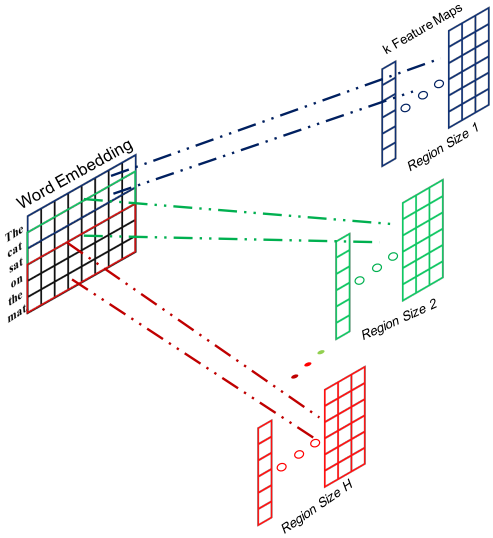


Figure 3: Illustration of parallel convolution procedure. It involves H parallel convolution layers with varying convolution window sizes and each layer has k filters. Each convolution layer is followed by an attention pooling layer to generate a distinct sentence representation. The H distinct representations are then concatenated to form the final representation vector.

tence representations produced by all the distinct CNNs are concatenated to form the final feature vector as an input to the top softmax classifier.

3.5. Softmax Classifier

The sentence representation s is naturally regarded as an input to the top classifier during the training phase while $[s, \widetilde{s}]$ is used at the testing phase. A linear transformation layer and a softmax layer are added at the top of the model to produce conditional probabilities over the class space. To avoid overfitting, dropout with a masking probability p is applied to the penultimate layer. The key idea of dropout is to randomly drop units (along with their connections) from the neural network during the training phase

[34]. This output layer is calculated as follows:

$$y = \begin{cases} W_s(s \odot q) + b_s & \text{training phase} \\ W_s([s, \widetilde{s}]) + b_s & \text{testing phase} \end{cases} \tag{9}$$

$$P_c = \frac{exp(y_c)}{\sum_{c' \in C} exp(y'_c)} \tag{10}$$

where $\odot$ is an element-wise multiplication operator, q is the masking vector with dropout rate p which is the probability of dropping a unit during training, and C is the class number. In addition, a l_2 norm constraint of the output weights W_s is imposed during training as well.

As our model is a supervised method, each sentence S has its golden label P_c^g. The following objective function in terms of minimizing the categorical cross-entropy is used:

$$\mathbb{L} = -\sum_{i=1}^{N}\sum_{c=1}^{C} P_c^g(S_i) log(P_c(S_i)) \tag{11}$$

where P_c^g has a 1-of-K coding scheme whose dimension corresponding to the true class is 1 while all others being 0. The parameters to be determined by the model include all the weights and bias terms in the convolutional filters, the BLSTM and the softmax classifier. The attention weights will be updated during the training phase. Word embeddings are fine-tuned as well. Optimization is performed using the Adadelta update rule of [40], which has been shown as an effective and efficient back-propagation algorithm.

The entire learning algorithm of APCNN is summarized as Algorithm 1.

4. Experimental Results and Discussions

In this section, we evaluate the performance of the proposed model on seven benchmark datasets for text classification and compare it with state-of-the-art approaches. The performance of the proposed model is evaluated by comparing it with other pooling strategies in Section 4.4. In Section 4.5, statistical tests are carried to demonstrate that the improvement of our method over other approaches is statistically significant. A sensitivity analysis of four key parameters of the model is done in Section 4.6. The effectiveness of the *Attention Pooling* mechanism is verified in Section 4.7. We also visualize the sentence representation space in Section 4.8 in order to have a better understanding of the sentence distribution produced by our model.

We test our model on seven benchmark datasets for sentence classification tasks which are the same as those used in [15]. Statistics of the datasets are listed in Table 2. To facilitate the following discussion, the datasets are now briefly described:

4.1. Datasets

- **MR**[25]: This is the dataset for movie reviews with one sentence per review. The objective is to classify each review into either positive or negative by its overall sentiment polarity. The class distribution of this dataset is 5331/5331.[4]

- **SUBJ** [24]: This is the subjectivity dataset where the goal is to classify a sentence as beiing subjective or objective. The class distribution is

[3]CV means there is no standard train/test split and thus 10-fold CV is used

[4]https://www.cs.cornell.edu/people/pabo/movie-review-data/

Table 2: Characteristics of datasets

Dataset	Number of classes	Average sentence length	Maximum sentence length	Dataset size	Test-set size[3]
MR	2	20	56	10662	CV
SUBJ	2	23	120	10000	CV
CR	2	19	105	3784	CV
MPQA	2	3	36	10606	CV
SST-1	5	18	53	11855	2210
SST-2	2	19	53	9618	1821
TREC	6	10	37	5952	500

5000/5000.

- **CR** [11]: This dataset gives customer review of various products (MP3s, cameras, etc.) where the objective is to classify each review into positive or negative class. The class distribution is 2411/1373.[5]

- **MPQA** [37]: This is the opinion polarity detection subtask of the MPQA dataset. The class distribution is 3310/7296.[6]

- **SST-1** [33]: This is the Stanford Sentiment Treebank dataset, an extension of MR dataset but with train/dev/test splits provided and fine-grained labels (very positive, positive, neutral, negative, very negative). The class distribution is 1837/3118/2237/3147/1516.[7]

[5] http://www.cs.uic.edu/liub/FBS/sentiment-analysis.html

[6] http://www.cs.pitt.edu/mpqa/

[7] http://nlp.stanford.edu/sentiment/ We train the model on both phrases and sentences but only score on sentences at test time, as in [14, 15, 16, 33]. Thus the training set is an order of magnitude larger than listed in Table 2

- **SST-2**: This dataset is derived from SST-1, which only has binary labels by excluding neutral reviews. The class distribution is 4955/4663.

- **TREC** [19]: This is the question classification dataset where the objective is to classify a question into 6 question types (whether the question is about person, location, numeric information, etc.). The class distribution is 1288/916/1009/1344/95/1300.[8]

4.2. Comparison Systems

The performance of the proposed method is compared with different basic baseline and state-of-the-art neural sentence models reviewed in the introduction section. Two Naive Bayes-based models are also included so as to compare with methods which do not use word embedding. The comparison systems are described as follows:

- **NB-SVM** and **MNB** (Naive Bayes SVM and Multinomial Naive Bayes): They are developed by taking Naive Bayes log-count ratios of uni- and bi-gram features as input to the SVM classifier and Naive Bayes classifier respectively [36]. The word embedding technique is not used.

- **cBoW** (Continuous Bag-of-Words): This model uses average or max pooling to compose a set of word vectors into a sentence representation.

- **RAE**, **MV-RNN** and **RNTN** (Recursive Auto-encoder [32], Matrix-vector Recursive Neural Network [29] and Recursive Neural Tensor Net-

[8] http://cogcomp.cs.illinois.edu/Data/QA/QC/

[9] RecursiveNN, RecurrentNN, and CNN stand for recursive neural networks, recurrent neural networks, and convolutional neural networks respectively.

Table 3: Characteristics of the comparison systems

Comparison systems	Word embedding	Category of neural systems[9]
NB-SVM and MNB	×	–
cBoW	✓	average/max
RAE	✓	RecuriveNN
MV-RNN	✓	RecuriveNN
RNTN	✓	RecuriveNN
RNN	✓	RecurrentNN
BRNN	✓	RecurrentNN
CNN	✓	CNN
one-hot CNN	×	CNN
DCNN	✓	CNN
P.V.	✓	CBOW/skip-gram

work [33]): These three models belong to recursive neural networks and recursively compose word vectors into sentence vector along a parse tree. Every word in the parse tree is represented by a vector, a vector and a matrix and a tensor-based feature function in RAE, MV-RNN, and RNTN respectively.

- **RNN** and **BRNN** (Recurrent Neural Network [8] and Bidirectional Recurrent Neural Network [28]): The RNN composes words in a sequence from the beginning to the end into a final sentence vector while the BRNN does the composition from both the beginning to the end and the end to the beginning.

- **CNN**, **one-hot CNN** and **DCNN** (Standard Convolutional Neural Network [15], One-hot vector Convolutional Neural Network [13] and Dynamic Convolutional Neural Network [14]): The CNN and the DCNN use pre-trained word vectors while the one-hot CNN employs high dimensional 'one-hot' vector representation of words as input. The CNN and the one-hot CNN employ max pooling and the DCNN uses k-max pooling at the pooling stage.

- **P.V.** (Paragraph Vector [16]): It learns representations for sentences or paragraphs in the same way as learning word vectors using CBOW and skip-grams.

The characteristics of the comparison systems are summarized in Table 3.

4.3. Parameter Settings

The authors of [41] provide a guide regarding CNN architecture and hyperparameters for practitioners who deploy CNNs for sentence classification tasks. We follow their suggestions to select parameters for our model. The paper shows that using word2vec or GloVec pre-trained word vectors may result in different performances, but with slight differences. We choose word2vec for all the datasets in this paper. The dimension of each word vector is 300. For all tasks, the word vectors are fine-tuned during the training phase [15].

It is claimed in [41] that the filter window size and the number of feature maps may have large effects on the performance while regularization

Table 4: Parameter settings of different datasets

Dataset	Region size	Feature maps	dropout rate	$l2$ norm constraint
MR	(4,5,6)	200	0.5	4
SUBJ	(6,7,8)	200	0.4	3
CR	(6,7,8)	200	0.3	4
MPQA	(4,5,6)	100	0.4	5
SST-1	(2,3,4)	100	0.5	3
SST-2	(2,3,4)	100	0.5	3
TREC	(2,3,4)	100	0.4	5

may have relatively small effects. We employ coarse grid search and cross-validation to determine these parameter values. As suggested by authors of [41], it is appropriate to set the region sizes for parallel CNNs near the single best size. We first determine the best single filter window size (odd number) and then use two adjacent region sizes. Three parallel CNNs are used in order to compare fairly with models in [13] and [15]. Settings for the four parameters for the datasets are listed in Table 4. However, sensitivity analysis over the four parameters shows that their choices do not affect the final performance too much as long as they are in an appropriate range.

The output dimension of the BLSTM for each dataset is set the same as the number of feature maps in order to compare local representations with intermediate sentence representation. The training batch size for SST-1/SST-2 is set as 100 while that for the other datasets as 50 because the number of training dataset size of SST-1/SST-2 is very large. Every experiment is conducted with 30 epochs. The learning rate, decay factor, and fuzzy factor of

Adadelta are set as 1.0, 0.95, and $1e-6$ respectively. Our model is developed based on keras[10]. All simulation studies are conducted using a GeForce 510 GPU on a Windows PC with 2.0 GHZ CPU and 4 GB RAM.

4.4. Comparison of Classification Accuracy

The classification accuracy of APCNN compared with other approaches is given in Table 5. The results of NB-SVM, MNB, RAE, MV-RNN, RNTN, CNN and DCNN are extracted from their original papers. The results of one-hot CNN are extracted from [41]. Public implementation of the P.V. is used and the logistic regression is used on top of the pre-trained paragraph vectors for prediction. For cBoW, RNN, and BRNN, they were implemented by ourselves and the best results are reported.

We can see from the table that the two Bayesian models achieve very good performance on datasets with long sentences but perform not well on datasets with short sentences. This results from the sparsity of n-gram encoding for short sentences. The cBoW is not performing very well which should result from losing word order information in the sentence. It is surprising that the three recursive neural network structures, namely RAE, MV-RNN and RNTN do not achieve very satisfactory performances. Their performances largely rely on the construction of the parse trees. The models may be so complicated that the problem of over-fitting affects classification accuracy. Compared with RNN, the bidirectional structure helps BRNN extract more information and enhance the learning performance.

When we compare the two one-layer CNN structures with word vectors

[10]http://keras.io

Table 5: Classification accuracy results of APCNN against other approaches on benchmark datasets

System	MR	SUBJ	CR	MPQA	SST-1	SST-2	TREC
NB-SVM	79.4	93.2	81.8	86.3	–	–	–
MNB	79.0	93.6	80.0	86.3	–	–	–
cBoW	77.2	91.3	79.9	86.4	42.8	81.5	87.3
RAE	77.7	–	–	86.4	43.2	82.4	–
MV-RNN	79.0	–	–	–	44.4	82.9	–
RNTN	–	–	–	–	45.7	85.4	–
RNN	77.2	92.7	82.3	90.1	47.2	85.8	90.2
BRNN	81.6	93.2	82.6	90.3	48.1	86.5	91.0
CNN	81.5	93.4	84.3	89.5	48.0	87.2	93.6
one-hot CNN	77.8	91.1	78.2	83.9	42.0	79.8	88.3
DCNN	–	–	–	–	48.5	86.8	93.0
P.V.	74.8	90.5	78.1	74.2	48.7	87.8	91.8
APCNN	**82.5**	**94.3**	**85.8**	**90.7**	**50.1**	**89.9**	**93.9**

and one-hot vectors as input respectively, we find that the one-hot approach performs much worse than the word-embedding approach. This shows that one-hot CNN may not be suitable for sentence classification although it achieves good performance on document classification tasks in [13]. The reason may be that the sentences are too short to provide enough information for high-dimensional encoding, resulting in extreme sparsity. The classification accuracy of DCNN with multiple layers and k-max pooling does not improve too much compared with the one-layer CNN model, proving that simple models can already be very effective in achieving competitive perfor-

mance. Another surprising result is that P.V. yields very bad performance on several datasets while it demonstrates state-of-the-art performance on almost all the tasks in [16]. This indicates that P.V. may only work for certain datasets.

We can conclude from the table that the APCNN consistently outperforms the other systems in all the tasks. We believe that it is the attention pooling strategy that helps the new model outperform the max pooling-based CNN approach because it can extract the most significant information contained in the sentence. Furthermore, the combination of the BLSTM model and convolutional structure enables the model to extract comprehensive information, namely historical, future and local context of any position in a sequence. The effectiveness of the new model has been verified by experiments.

4.5. Statistical Test

In order to verify that the performance improvement of the APCNN over other approaches is statistically significant, we perform some statistical tests. We use paired comparison t-tests because we believe the results of distinct approaches on the same dataset have correlations to some extent.

Paired t-test is a statistical technique that is used to compare population means of two approaches. It can be used for comparison of two different methods of measurement when the measurements are applied to the same subject. For each approach, we conducted the experiment ten times for statistical comparison (the accuracy is supposed to be the same for ten times if the results are extracted from the original papers).

Firstly, we set up the null hypothesis as the mean difference of the two

paired methods is zero. Next, we calculate the differences of each time for each pair of the methods. Next, the mean difference $\overline{d}$ and standard deviation of differences s_d are calculated in order to obtain the t-statistic as follows:

$$t = \frac{\overline{d}}{s_d/\sqrt{n}} \tag{12}$$

where n is the number of pairs of observation, which is 10 in our case. Under the null hypothesis, the t-statistic follows a t-distribution with $n-1$ degrees of freedom. Comparing the t-statistic with the t-distribution table gives the p-value for the paired t-test. The p-value is used to indicate whether the differences of measurements of the two methods are statistically significant.

We find the associated p-values for each pair of approaches on all the datasets are all smaller than 0.001. As the precision of p-values from t-distribution table can only reach 0.001, the complete p-value results are not shown. Therefore, we can conclude that our proposed method obtains superior performance compared with the other methods at nearly 100% confidence.

4.6. Sensitivity Analysis

In this section, the sensitivity analysis over predefined parameters was performed to confirm that they are not problem-specific. The four key predefined parameters are filter window size, the number of feature maps, dropout rate and l_2 norm constraint. They are not included in the hyper-parameters which are updated during the training phase. The authors of [41] claimed the filter window size and the number of feature maps may have large effects on the performance while regularization may have relatively small effects. However, Kim [15] demonstrated that dropout is a very good regularizer to

improve learning performance. The two statements contradict each other in some way. Therefore, it is necessary to do a sensitivity analysis on the key parameters.

We conducted experiments on the seven datasets to evaluate the parameter effects. The basic configurations for the four parameters are set as (3, 100, 0.5, 3) for (filter window size, the number of the feature maps, dropout rate, l_2 norm constraint) respectively. When analyzing the effect of one parameter, we hold all the other parameters constant at the basic configuration values. Because the accuracy ranges of different datasets are quite different, we show the percent change compared with the base point rather than the actual accuracy for each dataset.

The effects of the four parameters are depicted in Figure 4. We can see from Figure 4(a) that different datasets may have distinct optimal filter window sizes. For those datasets with very long sentences (e.g. CR and SUBJ), it is appropriate to choose a relatively large window size. Figure 4(b) demonstrates that the number of feature maps has a strong impact on the learning performance of the model. When the number of feature maps is small, the accuracy can be very small. However, as the size increases, the performance does not improve too much and can even deteriorate because of the problem of over-fitting. Furthermore, the model complexity increases quickly with the increasing number of feature maps. Therefore, we set the number of feature maps as 100 or 200 in our experiments.

The authors of [41] claimed that the dropout regularization does not help too much in performance improvement. However, our experiments indicate that the dropout is a very effective regularizer. As indicated by Figure 4(c),

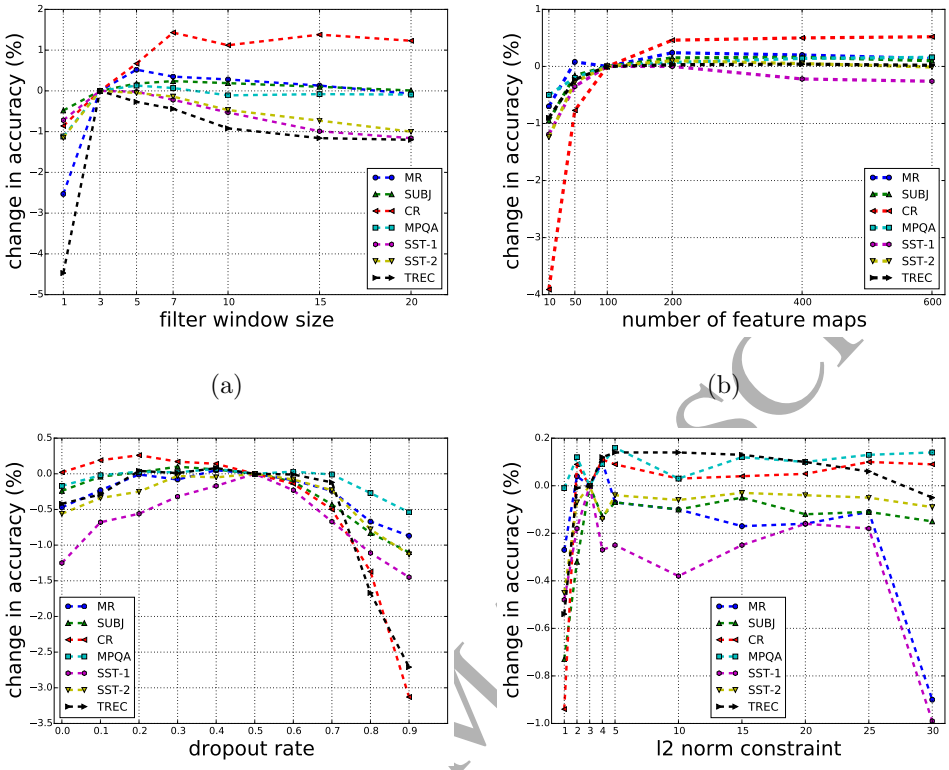


Figure 4: Sensitivity analysis of (a) filter window size, (b) number of feature maps, (c) dropout rate, and (d) l_2 norm constraint. The plots depict the percent changes of accuracy compared with basic configurations for the four parameters respectively on seven datasets. The basic configurations are set as (3, 100, 0.5, 3). In (c), the zero value of dropout rate means that no dropout is used. (Readers are referred to the web version of the article for a better interpretation)

the performance for some datasets can improve a lot (more than 1%) with the help of dropout. We believe that the power of dropout can be even stronger if we use more complex models. However, it is true that the performance varies very little when the dropout rate ranges from 0.2 to 0.6. Too large

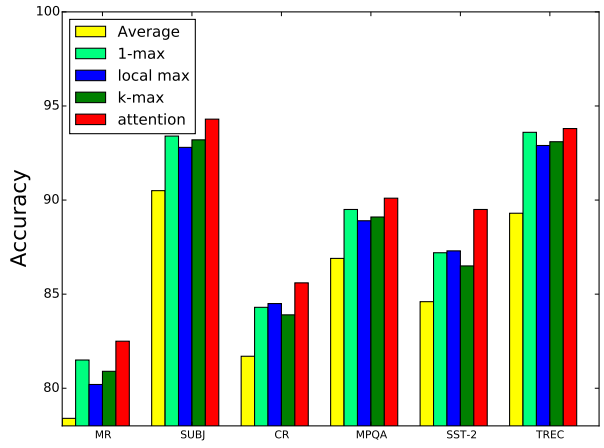


Figure 5: Accuracy comparison by the impact of pooling strategies. Five pooling strategies are compared on six datasets. (Readers are referred to the web version of the article for a better interpretation)

values of dropout rate can lead to under-fitting thus resulting in very bad performance. We set the dropout rates for different datasets in the range of $[0.3, 0.5]$. At last, Figure 4(d) shows that the l_2 norm constraint indeed has limited effect on the learning performance as long as the gradient does not become too large.

4.7. Comparison of Pooling Strategies

In this section, we verify the effectiveness of the attention pooling strategy by comparing with existing pooling strategies. For local max pooling, we set the pooling size as 10. For k-max pooling, k is set as 3. Only pooling strategies are different while all the other parameters are set the same. For our model, the intermediate sentence representation generated by the BLSTM model is not included in the input of the softmax layer in the testing phase for fairness in comparison. Comparison of accuracy is depicted in Figure 5.

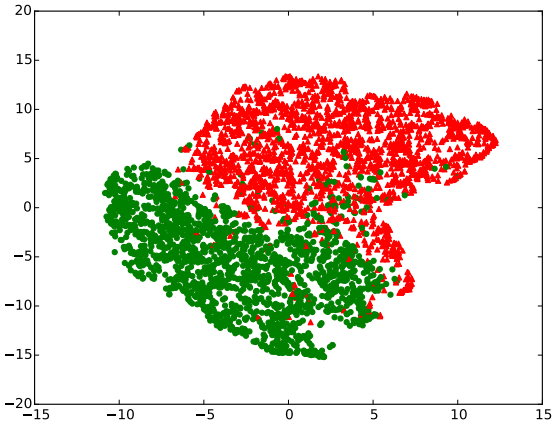
We leave SST-1 out for a better comparison because the accuracy range for SST-1 is much smaller than those of other datasets. The comparison result of SST-1 is similar to that of SST-2.

We can see from the figure that the attention pooling strategy outperforms all the other pooling strategies across all the datasets. The average pooling strategy performs the worst, showing that setting equal weights to all the features is unreasonable. It is surprising that 1-max pooling strategy achieves better performance than k-max and local max pooling in most cases although the latter two preserve more information. The 1-max pooling even achieves competitive performance in the dataset TREC compared with attention pooling. This demonstrates the significance of the maximal feature. After all, we can conclude that the new pooling scheme is effective because it achieves the best performance. The primary reason is because the new pooling scheme can extract the most significant information contained in the sentences.

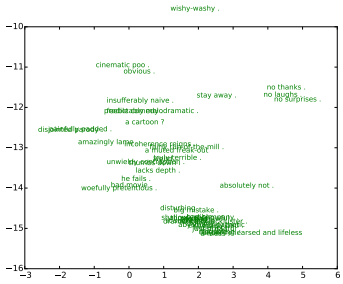
4.8. Visualization of Sentence Representation Space

To further show that the proposed model is able to learn appropriate sentence representations, we visualize the semantic space of sentence vectors of SST-2 dataset produced by our model by mapping the high-dimension sentence vector to a two-dimension plane. This mapping procedure is done by using PCA and t-SNE [21]. We first map the sentence representation vectors to the 50-dimensional space with PCA and then map the obtained 50-dimension vectors to the 2-dimension plane with t-SNE[11]. Visualization

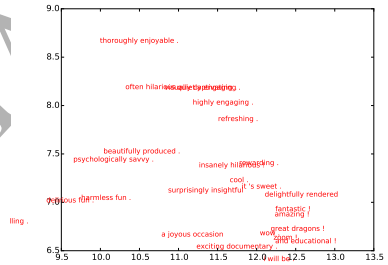
[11]The two dimension reduction procedures are done using the scikit-learn toolkit [26].



(a)



(b)



(c)

Figure 6: Visualization of the sentence representation space of SST-2 dataset. (a) Distribution of sentence vectors from different classes. (b) and (c) Enlarged views from negative (green dots) and positive (red triangles) areas respectively. The dots and triangles are replaced by the represented sentences in (b) and (c). (Readers are referred to the web version of the article for a better interpretation and are suggested to amplify the figures (b) and (c) to look at the texts)

results are shown in Figure 6.

Figure 6(a) shows the distribution of sentence vectors where red triangles

denote positive sentiment and green dots denote negativeness. It is clear that sentence vectors belonging to the same classes are tightly clustered together in the semantic space, which enables better classification performance. This is very interesting because no explicit attempt has been made to separate sentences from different classes. We also depict in Figure 6(c) and Figure 6(b) examples of positive and negative areas extracted from Figure 6(a). The dots and triangles are replaced by the sentences they represent. It can be seen that the sentences with similar semantics lie close to each other. This proves that the new model is very effective in modelling sentence representation.

5. Conclusions

In this paper, a new neural sentence model termed *Attention Pooling-based Convolutional Neural Network* has been successfully developed and an innovative attention pooling strategy has been proposed. The bidirectional long-short term memory model is exploited to generate an intermediate sentence representation which is used to obtain attention weights for pooling. The attention weights help extract the most significant information contained in the sentence. Furthermore, combining the bidirectional long-short term memory with the convolutional structure enables the model to extract comprehensive information, namely historical, future and local context of any position in a sequence.

Our new model can be trained end-to-end with limited hyper-parameters. It can extract comprehensive and the most significant information contained in a sentence. The sentence representation learning ability of the proposed model is very powerful in that it can implicitly separate the sentences from

different classes in the semantic space. Experimental results demonstrate that the new model outperforms state-of-the-art approaches on seven benchmark datasets for text classification. For future works, the new method can be applied to other natural language processing tasks and even computer vision applications. The proposed method uses bidirectional long-short term memory to generate intermediate sentence presentation. Other models, such as the gated recurrent units and auto encoders, may also be subject of future investigation. One problem of the model is heavy computational burden although the model has only one convolutional and one pooling layer. Improving computational efficiency of this model can be a promising future research direction.

Acknowledgment

The authors would like to acknowledge the funding support from the Ministry of Education, Singapore (Tier 1 AcRF, RG29/15), the National Natural Science Foundation of P. R. China (under Grants 51009017 and 51379002), Applied Basic Research Funds from Ministry of Transport of P. R. China (under Grant 2012-329-225-060), and Program for Liaoning Excellent Talents in University (under Grant LJQ2013055). Yong Zhang is supported by the NTU Research Scholarship.

References

[1] Ba, J., Mnih, V., Kavukcuoglu, K.. Multiple object recognition with visual attention. In: Proceedings of 3rd International Conference on Learning Representations (ICLR). 2015. .

[2] Bahdanau, D., Cho, K., Bengio, Y.. Neural machine translation by jointly learning to align and translate. In: Proceedings of 3rd International Conference on Learning Representations (ICLR). 2015. .

[3] Bengio, Y., Ducharme, R., Vincent, P., Janvin, C.. A neural probabilistic language model. The Journal of Machine Learning Research 2003;3:1137–1155.

[4] Boureau, Y.L., Bach, F., LeCun, Y., Ponce, J.. Learning mid-level features for recognition. In: 2010 IEEE converences on Computer Vision and Pattern Recognition (CVPR). IEEE; 2010. p. 2559–2566.

[5] Boureau, Y.L., Roux, N.L., Bach, F., Ponce, J., LeCun, Y.. Ask the locals: multi-way local pooling for image recognition. In: 2011 IEEE International Conference on Computer Vision (ICCV). IEEE; 2011. p. 2651–2658.

[6] Collobert, R., Weston, J.. A unified architecture for natural language processing: Deep neural networks with multitask learning. In: Proceedings of the 25th international conference on Machine learning (ICML). ACM; 2008. p. 160–167.

[7] Frege, G.. Sense and reference. The philosophical review 1948;:209–230.

[8] Funahashi, K.i., Nakamura, Y.. Approximation of dynamical systems by continuous time recurrent neural networks. Neural networks 1993;6(6):801–806.

[9] Graves, A., Schmidhuber, J.. Framewise phoneme classification with

bidirectional lstm and other neural network architectures. Neural Networks 2005;18(5):602–610.

[10] Hochreiter, S., Schmidhuber, J.. Long short-term memory. Neural computation 1997;9(8):1735–1780.

[11] Hu, M., Liu, B.. Mining and summarizing customer reviews. In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining (KDD). ACM; 2004. p. 168–177.

[12] Huang, E.H., Socher, R., Manning, C.D., Ng, A.Y.. Improving word representations via global context and multiple word prototypes. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1 (ACL). Association for Computational Linguistics; 2012. p. 873–882.

[13] Johnson, R., Zhang, T.. Effective use of word order for text categorization with convolutional neural networks. In: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL). 2015. p. 103–112.

[14] Kalchbrenner, N., Grefenstette, E., Blunsom, P.. A convolutional neural network for modelling sentences. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL). 2014. p. 655–665.

[15] Kim, Y.. Convolutional neural networks for sentence classification. In:

Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2014. p. 1746–1751.

[16] Le, Q.V., Mikolov, T.. Distributed representations of sentences and documents. In: Proceedings of the 31th international conference on Machine learning (ICML). 2014. p. 1188–1196.

[17] LeCun, Y., Bengio, Y., Hinton, G.. Deep learning. Nature 2015;521(7553):436–444.

[18] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.. Gradient-based learning applied to document recognition. Proceedings of the IEEE 1998;86(11):2278–2324.

[19] Li, X., Roth, D.. Learning question classifiers. In: Proceedings of the 19th international conference on Computational linguistics-Volume 1 (ACL). Association for Computational Linguistics; 2002. p. 1–7.

[20] Maas, A.L., Hannun, A.Y., Ng, A.Y.. Rectifier nonlinearities improve neural network acoustic models. In: Proceedins of The 30th International Conference on Machine Learning Workshop on Deep Learning for Audio, Speech, and Language Processing (ICML). volume 30; 2013. .

[21] Van der Maaten, L., Hinton, G.. Visualizing data using t-sne. Journal of Machine Learning Research 2008;9(2579-2605):85.

[22] Mikolov, T., Chen, K., Corrado, G., Dean, J.. Efficient estimation of word representations in vector space. In: Proceedings of Workshop at First International Conference on Learning Representations (ICLR). 2013. .

[23] Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.. Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems (NIPS). 2013. p. 3111–3119.

[24] Pang, B., Lee, L.. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In: Proceedings of the 42nd annual meeting on Association for Computational Linguistics (ACL). Association for Computational Linguistics; 2004. p. 271–278.

[25] Pang, B., Lee, L.. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (ACL). Association for Computational Linguistics; 2005. p. 115–124.

[26] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 2011;12:2825–2830.

[27] Pennington, J., Socher, R., Manning, C.D.. Glove: Global vectors for word representation. In: Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP). 2014. p. 1532–1543.

[28] Schuster, M., Paliwal, K.K.. Bidirectional recurrent neural networks. Signal Processing, IEEE Transactions on 1997;45(11):2673–2681.

[29] Socher, R., Huval, B., Manning, C.D., Ng, A.Y.. Semantic compositionality through recursive matrix-vector spaces. In: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP&CNLP). Association for Computational Linguistics; 2012. p. 1201–1211.

[30] Socher, R., Lin, C.C., Manning, C., Ng, A.Y.. Parsing natural scenes and natural language with recursive neural networks. In: Proceedings of the 28th international conference on machine learning (ICML). 2011. p. 129–136.

[31] Socher, R., Manning, C.D., Ng, A.Y.. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In: Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop. 2010. p. 1–9.

[32] Socher, R., Pennington, J., Huang, E.H., Ng, A.Y., Manning, C.D.. Semi-supervised recursive autoencoders for predicting sentiment distributions. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics; 2011. p. 151–161.

[33] Socher, R., Perelygin, A., Wu, J.Y., Chuang, J., Manning, C.D., Ng, A.Y., Potts, C.. Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of the conference on empirical methods in natural language processing (EMNLP). 2013. p. 1631–1642.

[34] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.. Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research 2014;15(1):1929–1958.

[35] Turian, J., Ratinov, L., Bengio, Y.. Word representations: a simple and general method for semi-supervised learning. In: Proceedings of the 48th annual meeting of the association for computational linguistics (ACL). Association for Computational Linguistics; 2010. p. 384–394.

[36] Wang, S., Manning, C.D.. Baselines and bigrams: Simple, good sentiment and topic classification. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2 (ACL). Association for Computational Linguistics; 2012. p. 90–94.

[37] Wiebe, J., Wilson, T., Cardie, C.. Annotating expressions of opinions and emotions in language. Language resources and evaluation 2005;39(2-3):165–210.

[38] Xu, K., Ba, J., Kiros, R., Courville, A., Salakhutdinov, R., Zemel, R., Bengio, Y.. Show, attend and tell: Neural image caption generation with visual attention. In: Proceedings of the 32nd International Conference on Machine Learning (ICML). 2015. p. 2048–2057.

[39] Zaremba, W., Sutskever, I.. Learning to execute. arXiv preprint arXiv:14104615 2014;.

[40] Zeiler, M.D.. Adadelta: an adaptive learning rate method. arXiv preprint arXiv:12125701 2012;.

[41] Zhang, Y., Wallace, B.. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. arXiv preprint arXiv:151003820 2015;.

[42] Zou, W.Y., Socher, R., Cer, D.M., Manning, C.D.. Bilingual word embeddings for phrase-based machine translation. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2013. p. 1393–1398.

Algorithm 1 Pseudo-code for Attention Pooling-based Convolutinoal Neural Network

1: The input and output of the proposed algorithm are N variable-length sentences and their corresponding labels P_c^g. The labels have been represented by 1-of-K coding scheme.
2: Considering one sentence, construct the sentence matrix using pre-trained word vectors with Eq. (1);
3: **for** i in [1, H] **do**
4: For the ith convolutional neural network with window size m^i, exploit one convolutional layer to obtain the local representations $c = [c_1, c_2, \cdots, c_T]$ using Eq. (4);
5: Employ the bidirectional long-short term memory to obtain the intermediate sentence representation $\widetilde{s}$;
6: Calculate the attention weights using Eq. (6-7);
7: Obtain the sentence representation using attention pooling with Eq. (8);
8: **end for**
9: Concatenate the H sentence representations to form the final sentence feature vector;
10: Feed the final sentence representation into a softmax classifier to predict the class label;
11: With all the training sentences and labels, update parameters of the model using the loss function Eq. (11) with the Adadelta update rule.
12: At testing phase, the intermediate sentence representation is concatenated with the sentence representation obtained by the convolutional and pooling layers to form the input of the softmax classifier.

Author Biography

Mr. Yong Zhang received his B. Eng. degree in Zhejiang University, China in 2013. He has been a full-time Ph.D. student in the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore since 2013. His research interests include machine learning, natural language processing, and computational intelligence.

Prof. Meng Joo Er is currently a Full Professor in Electrical and Electronic Engineering, Nanyang Technological University, Singapore. He has authored 5 books, 16 book chapters and more than 500 refereed journal and conference papers in his research areas of interest. His areas of research interests are Intelligent control theory and applications, computational intelligence, robotics and automation, sensor networks, biomedical engineering and cognitive science.

In recognition of the significant and impactful contributions to Singapore's development by his research works, Professor Er won the Institution of Engineers, Singapore (IES) Prestigious Engineering Achievement Award twice (2011 and 2015). He is also the only dual winner in Singapore IES Prestigious Publication Award in Application (1996) and IES Prestigious

Publication Award in Theory (2001). He received the Teacher of the Year Award for the School of EEE in 1999, School of EEE Year 2 Teaching Excellence Award in 2008, the Most Zealous Professor of the Year Award 2009 and Outstanding Mentor Award 2014. He also received the Best Session Presentation Award at the World Congress on Computational Intelligence in 2006 and the Best Presentation Award at the International Symposium on Extreme Learning Machine 2012. Under his leadership as Chairman of the IEEE CIS Singapore Chapter from 2009 to 2011, the Singapore Chapter won the CIS Outstanding Chapter Award 2012. In recognition of his outstanding contributions to professional bodies, he was bestowed the IEEE Outstanding Volunteer Award (Singapore Section) and the IES Silver Medal in 2011.On top of this, he has more than 50 awards at international and local competitions. Currently, Professor Er serves as the Editor-in-Chief of 2 international journals, namely the Transactions on Machine Learning and Artificial Intelligence and International Journal of Electrical and Electronic Engineering and Telecommunications, an Area Editor of International Journal of Intelligent Systems Science and Technology, an Associate Editor of thirteen refereed international journals including the IEEE Transactions on Cybernetics, Information Sciences, Neurocomputing, Asian Journal of Control as well as an editorial board member of the EE Times. Professor Er is a highly sought-after speaker and he has been invited to deliver more than 60 keynote speeches and invited talks overseas. Due to outstanding achievements in research and education, he is listed Who's Who in Engineering Singapore, Second Edition, 2013.

Prof. Ning Wang (S'08-M'12-SM'15) received his B. Eng. degree in Marine Engineering and the Ph.D. degree in control theory and engineering from the Dalian Maritime University (DMU), Dalian, China in 2004 and 2009, respectively. From September 2008 to September 2009, he was financially supported by China Scholarship Council (CSC) to work as a joint-training PhD student at the Nanyang Technological University (NTU), Singapore. In the light of his significant research at NTU, he received the Excellent Government-funded Scholars and Students Award in 2009. From August 2014 to August 2015, he worked as a Visiting Scholar at the University of Texas at San Antonio. He is currently a Full Professor with the Marine Engineering College, DMU, Dalian 116026, China.

Dr. Wang received the Nomination Award of Liaoning Province Excellent Doctoral Dissertation, the DMU Excellent Doctoral Dissertation Award and the DMU Outstanding PhD Student Award in 2010, respectively. He also won the Liaoning Province Award for Technological Invention and the honour of Liaoning BaiQianWan Talents, Liaoning Excellent Talents, Science and Technology Talents the Ministry of Transport of the P. R. China, Youth Science and Technology Award of China Institute of Navigation, and Dalian Leading Talents. His research interests include fuzzy neural systems, deep learning, nonlinear control, self-organizing fuzzy neural modeling and control, unmanned vehicles and autonomous control. He currently serves as Associate Editors of the Neurocomputing and the International Journal of

Fuzzy Systems.

Dr. Mahardhika Pratama received his PhD degree from the University of New South Wales, Australia in 2014. He completed his PhD in 2.5 years with a special approval of the UNSW higher degree committee due to his outstanding PhD research achievement. Dr. Pratama is currently working at the Department of Computer Science and IT, La Trobe University, Melbourne, Australia as Lecturer. Prior to joining La Trobe University, he was with the Centre of Quantum Computation and Intelligent System, University of Technology, Sydney as a postdoctoral research fellow of Australian Research Council Discovery Project. Dr. Pratama received various competitive research awards in the past 5 years and published over 50 high-quality papers.