Chinese Microblog Sentiment Analysis by Adding Emoticons to Attention-Based CNN

Yi-Jen Su[1], Chao-Ho Chen[2], Tsong-Yi Chen[2], Cheng-Chan Cheng[2]

[1] Department of Computer Science and Information Engineering, Shu-Te University, Taiwan
[2] Department of Electronic Engineering, National Kaohsiung University of Science and Technology, Taiwan
iansu@stu.edu.tw, {thouho, chentso, 1105305117}@nkust.edu.tw

Abstract

Nowadays, people are used to sharing their views and ideas on social media platforms, which generates enormous amounts of data every day. This research adopted sentiment analysis to disclose embedded information in Chinese short texts, which can serve as an integral part of social media monitoring and analytics. The research proposed a deep learning method, Attention-of-Emoticons Based Convolutional Neural Network (AEB-CNN), by integrating emoticons and attention-based mechanisms with CNN to enhance the accuracy of sentiment analysis. An implementation was carried out by TensorFlow; the accuracy of sentiment polarity of Chinese microblogs reached somewhere between 85.1% and 89.1% while achieving shorter execution time compared to other methods when the size of training dataset ranged from 10,000 to 30,000 sentences.

Keywords: Sentiment analysis, Attention-based, CNN, Emoticon

1 Introduction

Since Tim O'Reilly and Dale Dougherty published Web 2.0 in 2004, there has been significant change in how Internet users interact with other users on social media services, such as blogs, microblogs, and forums. SMS has attracted a large number of Internet users to share their views and ideas via SMS platforms to interact with others. The interactions between users easily lead to formation of subgroups with high-density relationships between members. Most of the time, the profuse texts contain valuable information that can be used for various purposes.

Sentiment analysis (or Opinion Discovery), a key process of social media monitoring and analytics, detects the sentiment polarity of public opinions by collecting enormous numbers of microblogs from social media platforms. Most related research works adopted natural language processing (NLP), n-gram, sentiment dictionary, or bag-of-mouth to parse and retrieve potentially useful information from the collected texts; nevertheless, there is room for improvement when it comes to the accuracy of such sentiment analysis.

The neural network was first successfully applied to handwriting recognition by Hinton et al., who proposed the deep belief network (DBN) in 2006 [1]. In recent years, with the rapid advances in computer technology, deep learning applications have been widely adopted in a great variety of research domains including sentiment analysis for text data. In order to take into consideration the word sequence in a sentence for better sentiment analysis results, Rumelhart et al. [2] proposed the hidden layer of recurrent neural network (RNN) [2], in which the state store unit can be used to merge the vectors of the current term and its previous state to make predictions about the next term. The operation is similar to human speech – there exist certain correlations between terms and their sentences. However, RNN is susceptible to vanishing gradients. When RNN is used in sentiment analysis, the farther a term is set, the less influence it has over the predictive term; however, such calculation overlooks the fact that the human mind may process memories differently, such as having particular memories concerning specific matters for some reason. Therefore, Hochreiter and Schmidhuber [3] proposed the long short-term memory network (LSTM) to compensate for the disadvantages of RNN. Xinjie Zhou et al. [4] used LSTM to enhance accuracy in cross-language sentiment analysis when there is not enough external knowledge. Yequan Wang et al. [5] combined the attention mechanism and LSTM to improve the accuracy rate of sentiment analysis.

Convolutional neural network (CNN) has been the core technique in deep learning for image processing in recent years. In 2012, Krizhevsky et al. [6] proposed AlexNet to create a new trend of CNN. Recently, some researchers have applied CNN to solve problems in the field of NLP research, including Kim [7], who adopted CNN to identify positivity/negativity in movie comments, and Kalchchbrenner et al. [8], who applied

*Corresponding Author: Yi-Jen Su; E-mail: iansu@stu.edu.tw
DOI: 10.3966/160792642020052103019

CNN to solve problems of classification raised at TREC (Text Retrieval Conference).

When CNN is processing the input data, the attention mechanism can assist CNN in focusing only on important features while ignoring the rest to enhance the result of classification. For example, Wenpeng Yin et al. [9] proposed combining single attention and CNN to solve the text entailment problem; Linlin Wang et al. [10] integrated multi-attention mechanisms with CNN to classify sentence relations. Combining attention mechanisms and CNN for solving NLP-related research problems yields better results because CNN can speed up computation convergence and compensate for the computation time needed for the attention mechanism.

This research proposes an attention-based mechanism with emoticons that can significantly reduce the manual tagging time for each aspect word derived from the output of the segmentation process. In addition, CNN was also adopted to capture the local features of each sentence, which makes it possible to focus attention on terms with higher sentiment polarity and ignore the rest for more effective sentiment identification. Based on our experiment, the proposed sentiment analysis method was able to reach significantly higher accuracy.

2 Related Works

This sentiment analysis research on Chinese microblogs adopts several techniques, including (1) using emoticons to speed up the Chinese term corpus construction, (2) utilizing word2vec to transform a term in a microblog text into a vector, where each term is represented by a real number vector that will then serve as the input of CNN, and (3) using CNN to train the Chinese sentiment analysis model with an attention mechanism to improve accuracy.

2.1 Word Vector

In the field of NLP, one-hot encoding, one of the simplest word vector methods, employs the "term vector" that represents a term with a long n-dimension vector. The length of a vector is the total number of terms in the corpus. As for the components of a vector, 1 represents the current term while 0 represents the others. The representation scheme has two drawbacks. One concern is the curse of dimensionality problem, which causes the size of vector space to expand so fast that it results in sparse data distribution, especially during deep learning processing. The other concern is that each term is independent from one another, making it difficult to find the relation between two terms using vectors.

The term vector representation maps a term to a real number vector with lower dimension. All vectors form a term-vector space. By adding distance to the problem space, all related terms or similar terms can be identified by way of Euclidean Distance or Cosine Similarity to overcome the weakness of one-hot encoding and solve the curse of dimensionality problem in order to improve the accuracy of sentiment polarity determination. As shown in Table 1, for example, all terms are mapped to a 5-dimention vector space. Each vector represents a term in the Chinese sentence "今天玩的很開心" (literal meaning: "Today - play - very – happy"; translation: "[I] had great fun today"), which was segmented into 5 Chinese semantic terms, 今天, 玩, 的, 很, 開心.

Table 1. Examples of 5-dimension term vectors

Term	Term Vector
今天	[2.0270526, 0.5960824, -0.15063474, 3.275252, 1.1437435]
玩	[2.9823174, 1.2120187, -1.2100589, 1.9672757, -1.1742682]
的	[-0.62663907, 0.5856641, 1.9925734, 2.535172, -1.6748154]
很	[1.9026364, 1.5810974, 4.159332, 0.46042323, -6.6090765]
開心	[2.8012135, 1.6971004, 3.3443644, -0.42471766, -1.6906428]

2.2 Convolutional Neural Network (CNN)

CNN is one of the most representative neural network architectures in deep learning, having made something like a "quantum leap" in image recognition technology. Two major factors led RNN to be replaced by CNN in NLP research. First, CNN shows excellent feature retrieving functions. Second, CNN demonstrates better performance on classification results and training speed than RNN. In addition, when Kalchbrenner adopted CNN to capture the sentiment polarity of audience opinions based on movie comments and Twitter posts, the research result proved CNN to have higher accuracy [11]. Yoon also proposed a brand-new CNN-Text model with term-vectors to determine sentiment polarity. The architecture contains 5 layers – the input layer, convolutional layer, pooling layer, fully connected layer, and output layer, as shown in Figure 1.

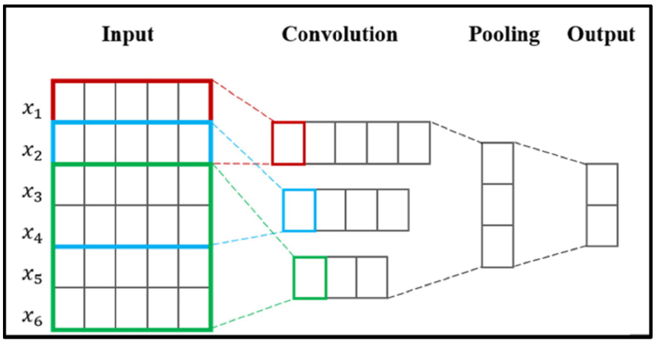

Figure 1. Architecture of CNN

(1) The input layer serves as a Chinese segmentation process for dividing the input Chinese sentence into individual terms; each term is then converted into a term vector to reduce the number of dimensions in the

distributed space. As each term is mapped to a k-dimension vector $x_i \in R^k$, a sentence with fixed length n is represented as the following Equation (1).

$$x_{1:n} = x_1 \oplus x_2 \oplus x_3 \oplus \ldots \oplus x_n \quad \textbf{(1)}$$

(2) The convolutional layer has varying-sized convolutional kernels. It processes the convolutional computation to derive the input layer's local features, which are represented as c_u. The computation is shown as follows in Equation (2),

$$c_u = f_c(W_c \cdot x_{u:u+d-1} + b_c) \quad \textbf{(2)}$$

where W_c stands for the weight of the convolutional kernel, d stands for the maximum length of the convolutional kernel, b_c is the bias value, and f_c stands for the Activation Function. The convolutional kernels vary in size and therefore differ in length and width, which are represented by their respective length d and width k. $X_{u:u+d-1}$ shows the matrix of the convolutional kernel. Once it receives the input sentence with length l, the convolution computation can be used to derive the feature map as shown in Equation (3),

$$c = [c_1, c_2, c_3, \ldots, c_{n-d+1}] \quad \textbf{(3)}$$

where C, the feature map, stands for a 1-dimension matrix of c_i with length n-d+1.

(3) In general, the pooling layer serves as a max-over-time pooling operation to derive a convolutional feature map through the convolutional computation, which outputs the feature with the maximum influence in the feature map, as shown in Equation (4):

$$\overline{c} = \max(c) \quad \textbf{(4)}$$

(4) After the convolutional layer and pooling layer complete iteration learning, an input feature enters the fully connected layer to merge all distributed features, which then serves as the input of the classifier to predict the final result.

2.3 Attention Mechanism

The attention mechanism was first proposed in the image recognition research domain. The original idea was that when people look at a picture, they tend to focus on only a few areas of interest, and use less of their attention resources on other areas. In the same vein, people tend to use limited attention resources to rapidly filter through important information, a mechanism that the human brain developed through evolution. In particular, paying attention on visual images can enhance effectiveness and accuracy. For example, as illustrated in Figure 2, the picture shows the human brain applying attention mechanism to understand a picture. People have much higher concern for the elephants than for the sky and grass.

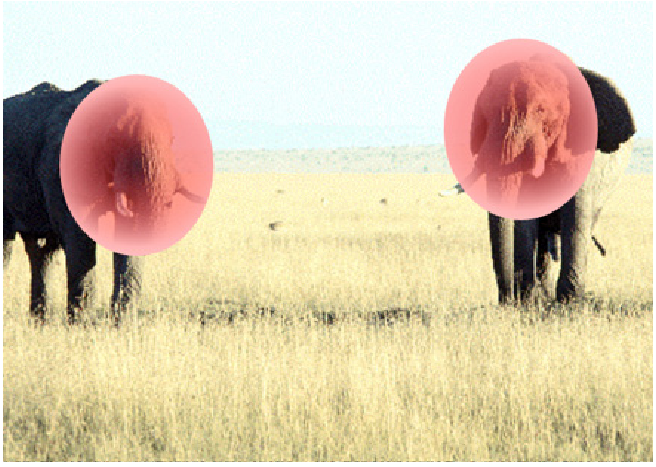

Figure 2. Example of attention mechanism on an image

In 2014, a team at Google Mind [12] applied RNN with attention mechanism to classify images. They followed the order of input images and utilized their program to repeatedly observe the areas that received interest. Then, they took the observation result as a basis to incrementally adjust the location of observation areas in order to predict the final classification result.

As shown in Table 2, for example, when attention mechanisms are employed in NLP operations, the cell value represents the attention weight. If the weight is higher, it means that the term bears higher attention, and vice versa. The attention weight can effectively filter out meaningless terms and identify the most influential terms, thereby securing higher accuracy in the predictions made.

Table 2. Example of attention mechanism on NLP

今天	玩	的	很	開心
0.037	0.148	0.010	0.010	0.037

Bahdanau et al. [13] was the first to apply attention mechanisms to neural machine translation (NMT). Under the attention mechanism, the proposed sequence follows the model sequence and learns the target term and the resource term with the information of attention weight. There are three main equations – Equations (5), (6), and (7) – that are used in calculating the focus areas, as shown in the following,

$$f(x_j, x_k) = x_j^T \cdot x_k \quad \textbf{(5)}$$

$$f(x_j, x_k) = x_j^T \cdot W \cdot x_k \quad \textbf{(6)}$$

$$f(x_j, x_k) = W \cdot [x_j; x_k] \quad \textbf{(7)}$$

where x_j and x_k represent the input. Equation (5) shows the calculation of inner product. Equation (6) increases the weight W and employs iteration learning in order to yield the value of the maximum attention weight.

Equation (7) applies concatenation-based attention and iteration learning through weight W to identify the maximum attention weight.

3 Research Method

To improve the accuracy of sentiment polarity judgement for Chinese microblogs, this research proposes the Attention of Emoticon-Based Convolutional Neural Network (AEB-CNN) model. The basic concept of the AEB-CNN model is to identify the local features of a sentence via CNN and to recognize the part of a sentence that bears a high sentiment value by combining emoticons and attention mechanisms, as shown in Figure 3.

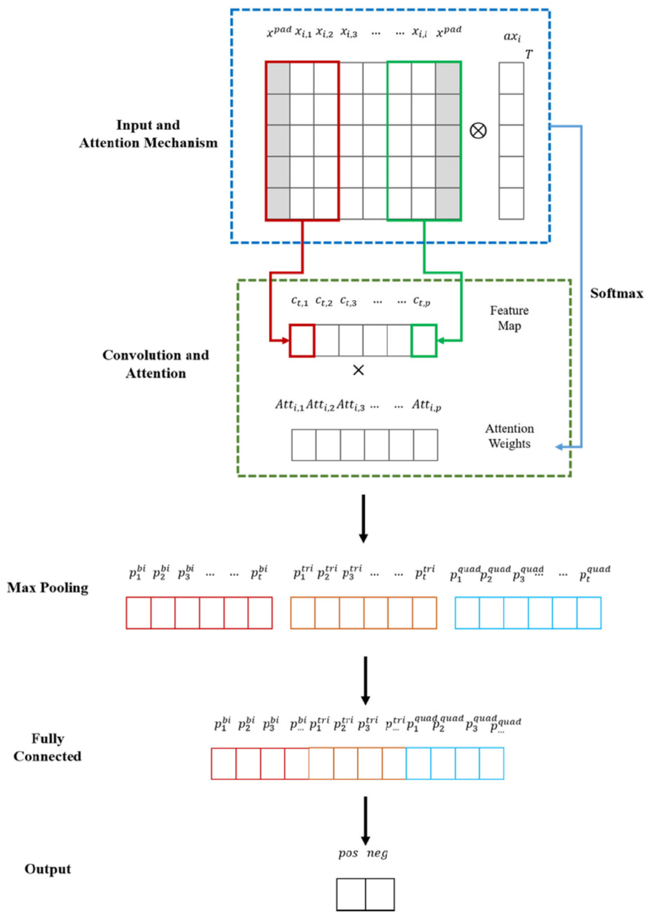

Figure 3. The proposed architecture of AEB-CNN

3.1 Data Preprocessing

During data preprocessing, we collected all training sentences with emoticons on Plurk, a famous Chinese SMS platform, using Plurk API. A sentence that contains emoticons is easily identifiable in terms of sentiment polarity being either positive or negative. For example, the emoticon :-D stands for positive sentiment while the emoticon X-(signifies negative sentiment.

After collection of Chinese texts from Plurk, the first step that followed was an operation to remove all nonsense terms or symbols, e.g. URL, number digits, nonsense symbols, and the like from the collected dataset, as shown in Table 3. The purpose is to prevent the training result of term vectors from being useless.

Table 3. Example of nonsense terms

No Effective Term
66666666
xddddddddddddddd
==、@@
http://www...
ss53gh413sadas454

The second operation is a Chinese-language segmentation work to divide each collected sentence into separate terms. The most popular segmentation tools include CKIP (Chinese Knowledge and Information Processing) [14], Jieba [15], LTP (Language Technology Platform) [16], ICTCLAS (Institute of Computing Technology, Chinese Lexical Analysis System) [17], and THULAC (THU Lexical Analyzer for Chinese) [18]. The drawback of CKIP is that there is a limit to the number of times allowed for running sentence segmentation. The segmentation process of Jieba runs fast, but it has difficulties recognizing certain terms, such as Chinese idioms, people's names, and more. By contrast, although LPT and CTCLAS enjoy higher accuracy, their processing speed is not fast enough. Given the above, this research chose to adopt THULAC to complete the segmentation for all collected Chinese texts mainly in consideration of shorter segmentation time.

This research selected THULAC as the segmentation model for three reasons. First, the model is known for higher accuracy in Chinese-language segmentation. Second, THULAC provides more API functions for the segmentation process. Lastly, THULAC's fast speed is advantageous in the case of a large amount of collected Chinese sentences. A comparison between different segmentation models is shown in Table 4.

Table 4. Comparison of different segmentation models

Algorithm	Precision	Recall	F-Measure	Speed
LTP	0.96	0.94	0.95	149.8KB/s
ICTCLAS	0.93	0.94	0.94	490.5KB/s
Jieba	0.85	0.78	0.81	2314.8KB/s
THULAC	0.94	0.90	0.92	1221.0KB/s

Following the above, Word2vec uses the segmentation result as the input to generate term vectors, as shown in Equation (8),

$$s_i = \{x_{i,1}, x_{i,2}, x_{i,3}, \ldots, x_{i,n}\} \quad \textbf{(8)}$$

where s_i denotes the i-th sentence and $x_{i,j}$ stands for the term vector of the j-th term of the i-th sentence.

The value distribution of unk (unknown word) and pad (padding word) shows a uniform distribution pattern with values that fall between -0.1 and +0.1 for random initial term vectors.

3.2 Wide Convolution

After data preprocessing, all terms are converted into term vectors and all input sentences are represented in the form of s_i. The CNN-Text model, proposed by Yoon, has two problems that preclude it from being applied in this research. First, the lengths of all input sentences are different. After running narrow convolution computation, the length of the convolutional feature map might differ from the length of the input sentence. This causes the proposed AEB-CNN model to work poorly. Second, the first term and the last term of a sentence have less computation chance under CNN-Text's narrow convolution.

To solve the above two problems, the research adopts Wide Convolution to generate a convolutional feature map. Based on the length of a convolution kernel, each sentence adds one or more pads to keep the convolution result the same in length as the input sentence, as shown in Figure 4.

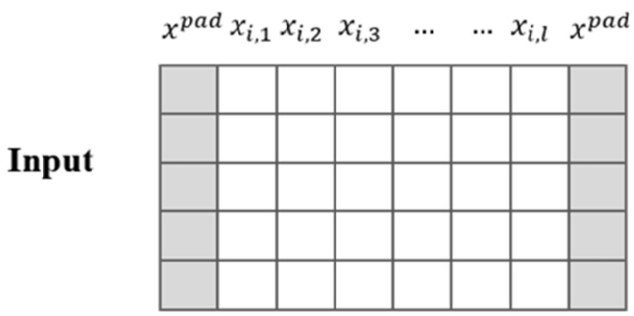

Figure 4. Example of all sentences have the same length by adopting pads

The number of pads in the wide convolution can be decided using Equation (9),

$$pad\_num = output\_num - input\_num + kernel\_size - 1 \quad \textbf{(9)}$$

where *pad_num* stands for the number of pads to be added with the input sentence, *input_num* for the length of the input sentence, *kernel_size* for the length of the convolution kernel, and *output_num* for the output length of the feature map.

Figure 5 is an illustration of the wide convolution's computation architecture. Both the red and green frames represent convolution kernels. $C_{t,u}$ is the *u*-th convolutional feature of the *t*-th feature map.

If the length of a convolution kernel is *h*, the convolution computation for input sentences matrix is as shown in Equation (10),

$$c_{t,u} = f_c(W_c \cdot x_{u:u+d-1} + b_c) \quad \textbf{(10)}$$

where w_c is the convolution weight and $w_c \in R^{h*k}$, b_c is the bias value, $b_c \in R$, f_c is the activation function, and $x_{u:u+d+1}$ is the matrix of the convolution kernel.

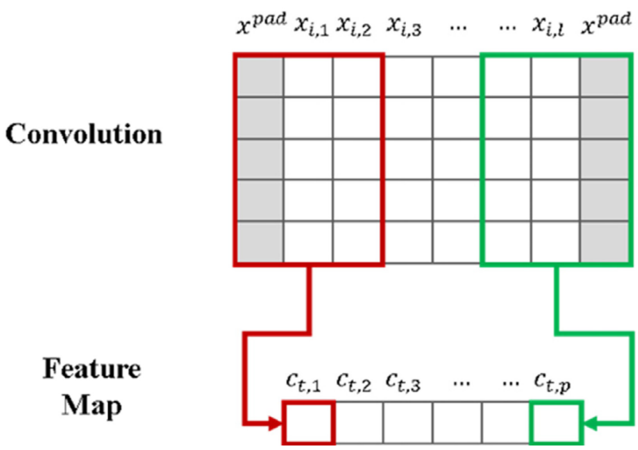

Figure 5. Architecture of wide convolution

When the length of a sentence is *l*, wide convolution computation can be used to get the feature map, as shown in Equation (11).

$$c_t = [c_{t,1}, c_{t,2}, c_{t,3}, \ldots, c_{t,p}] \quad \textbf{(11)}$$

where C_t, a 1-dimention array, stands for the convolution feature map, while $C_t \in R^p$, the length of the *t*-th feature map matrix, is *p*, the same as the length of input matrix *l*.

3.3 Attention Mechanism with Emoticons

In the attention weight computational process, all terms of a sentence s_i are converted into term vectors. Then, these term vectors are used to generate inner products, which are then used to calculate the attention weight aw_i to subsequently produce the attention weights matrix by way of the Softmax algorithm, as shown in Figure 6 as well as Equations (12) and (13).

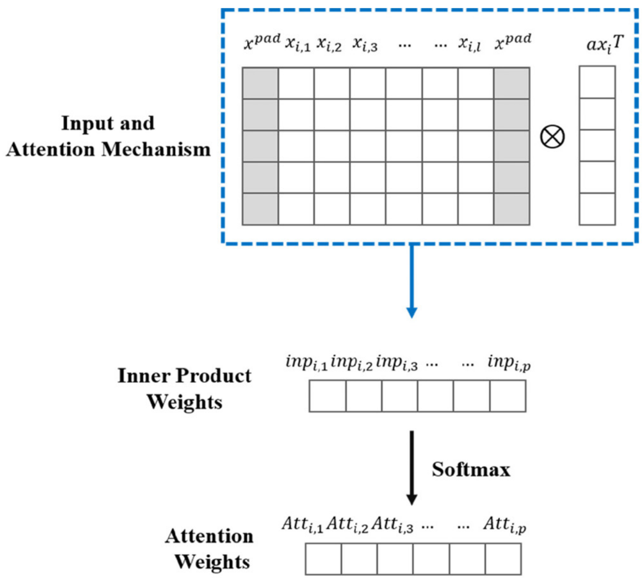

Figure 6. Structure of attention mechanism

$$inp_{i,j} = f(x_{i,j}, aw_i) \quad \textbf{(12)}$$

$$Att_{i,j} = \frac{\exp(inp_{i,j})}{\sum_{j=1}^{l}\exp(inp_{i,j})} \quad \textbf{(13)}$$

where x^{pad} is filled in for the wild convolution while $x_{i,j}$ and aw_i are combined to produce the inner product $inp_{i,j}$. Equation (13) shows the application of Softmax algorithm to derive the attention strength of a term, represented as $Att_{i,j}$.

Figure 7 demonstrates four exemplifying sentences that illustrate the attention mechanism computation. The greater the value of a term is, the higher influence this term bears in sentiment judgement.

A	今天	玩	的	很	開心	，	期待	下次	的	出遊
	0.12212336	0.4894375	0.00193216	0.00193216	0.12526338	0.00193216	0.05547103	0.07246048	0.00193216	0.01403722
B	早安	好友	們	一起	加油	*pad*	*pad*	*pad*	*pad*	*pad*
	0.24002655	0.14109854	0.00324814	0.17359838	0.23763435	0.00272524	0.00272524	0.00272524	0.00272524	0.00272524
C	氣	到	心臟	好痛	，	到底	在	想	些	什麼
	1.7786574e-01	3.6175043e-05	3.4540712e-03	2.7638025e-04	3.6175043e-05	3.6175043e-05	3.6175043e-05	8.1046230e-01	3.7097386e-03	3.6175043e-05
D	可惡	的	下雨天	破壞	我	的	計畫	*pad*	*pad*	*pad*
	9.3416911e-01	2.2935310e-04	4.1383214e-02	4.2509129e-03	2.2935310e-04	2.2935310e-04	6.8266443e-03	1.7374688e-04	1.7374688e-04	1.7374688e-04

Figure 7. Examples of the attention mechanism

3.4 Feature Extraction

In the feature extraction process, both the attention weight matrix and the convolution feature map matrix are used in matrix multiplication. Then, the result serves as the input of the activation function to generate the hidden layer, as shown in Equation (14),

$$h_t = f_h(W_h \cdot (c_t \times Att_i) + b_h) \quad \textbf{(14)}$$

where the convolution feature map matrix c_t is multiplied first by the attention weight matrix Att_i, then by the weight of the hidden layer W_h, added to the bias of the hidden layer b_h, and, finally, run through the activation function of the hidden layer f_h to generate hidden layer.

In order to identify semantic features with the most influence, semantic features are retrieved by convolution computation of the input data, feature extraction by attention mechanism, and down-sampling feature space by pooling. In Figure 8, the output of the feature extraction is the hidden layer h_t. Because each convolution feature map selects its largest semantic feature, a total of t largest features are generated. The research sets the lengths of convolution kernel at 2, 3, and 4, which correspond to bigram, trigram, and quadgram in n-gram models. As a result, three max pooling matrices, $pt_t^{b_i}, pt_t^{tri}, pt_t^{quad}$, enter the fully connected layer.

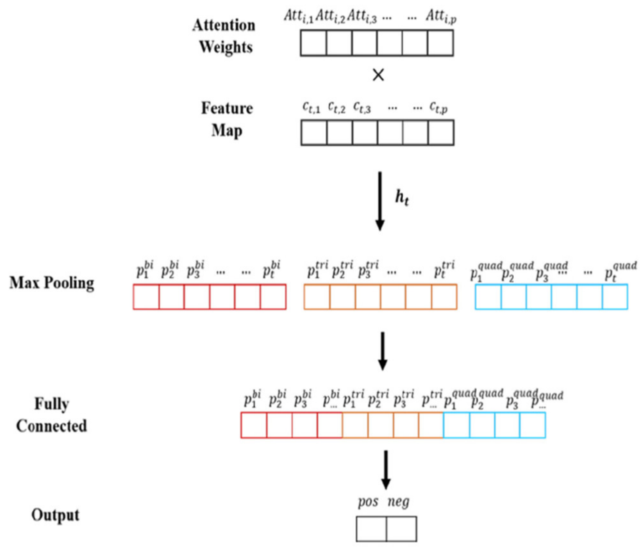

Figure 8. Feature extraction architecture of down sampling

The research adopted the max-over-time pooling method to sample the semantic features, as show in Equation (15),

$$p_t^h = \max(h_{t,1}, h_{t,2}, h_{t,3}, \dots, h_{t,p}) \quad \textbf{(15)}$$

which employs the convolution kernel with length h to output the t-th feature map. The max-over-time pooling operation runs from $h_{t,1}$ to $h_{t,p}$ in order to derive the fully connected layer matrix pt_t^h .

Lastly, the fully connected layer generates the prediction result, as shown in Equation (16),

$$y(s_i) = \mathrm{softmax}(W_s \cdot p_T^H + b_s) \quad \textbf{(16)}$$

where $y(s_i)$ stands for the prediction result of the $_i$-th sentence, W_s represents the weight matrix of the fully connected layer, p_T^H is the output of the fully connected layer, and b_s is the biased value of the fully connected layer.

4 Experiment Results

To prove the effectiveness of the AEB-CNN model on Chinese microblog sentiment analysis, the research randomly collected Chinese microblogs with emoticons from the famous Chinese SMS platform Plurk. All collected microblogs were divided into two datasets – the training set and the testing dataset. In the experiment, there were 10,000 sentences in the training set, which included 5,000 positive sentences and 5,000 negative sentences. The testing dataset, on the other hand, consisted of 3,000 sentences that were divided into two subgroups, each labeled positive or negative, with both having the same number of 1,500 sentences.

Based on research observations, most users of PTT, a famous BBS service platform in Taiwan, do not like to include emoticons in their Chinese texts. Therefore, the research collected 9,607,232 sentences from PTT from the period of 2018/03/01~2018/06/01 to serve as the term vector training set. After completing the training process, a total of 122,218 terms were retrieved to serve as the sentiment dictionary. The parameters applied by word2vec included window size 5, embedding size 300, and min count 10.

Table 4. Comparison of 4 sentiment analysis methods

Model	Accuracy	Precision	Recall	F1
SVM	77.3%	83.0%	68.6%	75.1%
LSTM	76.8%	83.1%	67.2%	74.3%
CNN	80.5%	78.9%	83.2%	81.0%
AEB-CNN	85.8%	81.7%	92.2%	86.6%

As illustrated in Table 4, the experiment collected 10,000 sentences as the training dataset and another 1,000 sentences as the testing dataset. When compared against another 3 famous research methods, including CNN, LSTM and SVM (support vector machine) [19], the AEB-CNN method demonstrated better accuracy and recall in sentiment analysis, indicating the superior effectiveness of the proposed method.

Figure 9 shows the varied accuracy when it comes to different-sized training datasets. The five datasets range in size from 10,000, 15,000, 20,000, 25,000, to 30,000 sentences. In all cases, the proposed AEB-CNN model displayed better accuracy performance than the other models. It is worth noting that the accuracy of LSTM always falls under 76.8% and at the 4th position.

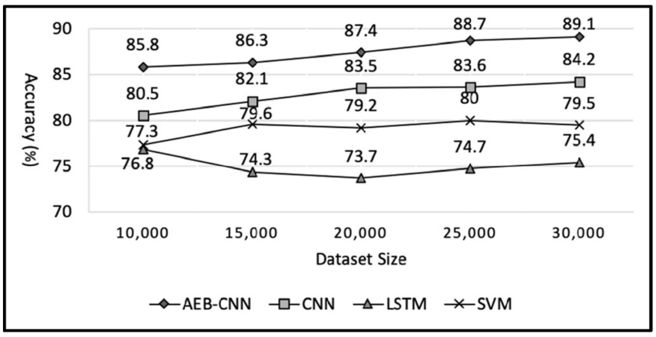

Figure 9. Accuracy of 4 sentiment analysis methods

Table 5 illustrates the different execution times when handling different numbers of sentences. The execution time includes the operations of preprocessing of the testing sentences, Chinese segmentation, term vector encoding, and the classification of sentiment polarity. It can be clearly seen that the CNN method yields the shortest execution time in all 5 different-sized testing datasets. The proposed method, AEB-CNN, comes in second place.

Table 5. Execution time comparison of 4 sentiment analysis methods

No. of Sentences / Method	1,000	1,500	2,000	2,500	3,000
SVM	28.525s	30.522s	33.321s	36.291s	38.816s
LSTM	8.812s	10.190s	12.146s	12.626s	14.473s
CNN	6.210s	7.283s	8.511s	9.882s	11.184s
AEB-CNN	6.422s	7.610s	8.725s	10.068s	11.439s

5 Conclusion

This research proposed the AEB-CNN model by integrating emoticons and attention mechanisms with CNN. By including emoticons in the calculation of attention weight, important key terms received high attention while terms with low emotional strength were ignored, thus enhancing the accuracy in identifying sentiment polarity of short texts as being either positive or negative. It is especially worth noting that the proposed method did not require manual tagging of aspect words or target words, which in turn saved a lot of time. The experiment results indicated that the proposed method showed high accuracy in identifying sentiment polarity in Chinese microblogs.

References

[1] G. E. Hinton, S. Osindero, Y. W. Teh, A Fast Learning Algorithm for Deep Belief Nets, *Neural Computation*, Vol. 18, No. 7, pp. 1527-1554, July, 2006.

[2] D. E. Rumelhart, G. E. Hinton, R. J. Williams, Learning Internal Representations by Error Propagation, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Vol. 1, pp. 318-362, MIT Press Cambridge,

1986.

[3] S. Hochreiter, J. Schmidhuber, Long Short-Term Memory, *Neural Computation*, Vol. 9, No. 8, pp. 1735-1780, December, 1997.

[4] X. Zhou, X. Wan, J. Xiao, Attention-based LSTM Network for Cross-Lingual Sentiment Classification, *2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, Austin, Texas, USA, 2016, pp. 247-256.

[5] Y. Wang, M. Huang, X. Zhu, L. Zhao, Attention-based LSTM for Aspect-level Sentiment Classification, *2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, Austin, Texas, USA, 2016, pp. 606-615.

[6] A. Krizhevsky, I. Sutskever, G. E. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, *2012 25th International Conference on Neural Information Processing Systems (NIPS'12)*, Lake Tahoe, Nevada, USA, 2012, pp. 1097-1105.

[7] Y. Kim, Convolutional Neural Networks for Sentence Classification, *2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, Doha, Qatar, 2014, pp. 1746-1751.

[8] N. Kalchbrenner, E. Grefenstette, P. Blunsom, A Convolutional Neural Network for Modelling Sentences, *2014 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, Baltimore, Maryland, USA, 2014, pp. 655-665.

[9] W. Yin, H. Schutze, B. Xiang, B. Zhou, ABCNN: Attention-Based Convolutional Neural Network for Modeling Sentence Pairs, *Transactions of the Association for Computational Linguistics*, Vol. 4, pp. 259-272, 2016.

[10] L. Wang, Z. Cao, G. Melo, Z. Liu, Relation Classification via Multi-Level Attention CNNs, *2016 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, Berlin, Germany, 2016, pp. 1298-1307.

[11] X. Yang, C. Macdonald, I. Ounis, Using Word Embeddings in Twitter Election Classification, *Information Retrieval Journal*, Vol. 21, pp. 183-207, June, 2018.

[12] V. Mnih, N. Heess, A. Graves, K. Kavukcuoglu, Recurrent Models of Visual Attention, *2014 27th International Conference on Neural Information Processing Systems (NIPS'14)*, Montreal, Canada, 2014, pp. 2204-2212.

[13] D. Bahdanau, K. Cho, Y. Bengio, Neural Machine Translation by Jointly Learning to Align and Translate, *2015 3rd International Conference on Learning Representations (ICLR-2015)*, San Diego, CA, USA, 2015.

[14] Y.-J. Su, R.-C. Chen, C.-M. Hsiung, Y.-Q. Chen, S.-W. Yu, H.-W. Huang, Using Prefix Tree to Improve the Performance of Chinese Sentiment Analysis, *2016 30th IEEE International Conference on Advanced Information Networking and Applications* (AINA-2016), Crans-Montana, Switzerland, 2016.

[15] Jieba Chinese Text Segmentation, https://github.com/ldkrsi/jieba-zh_TW.

[16] LTP (Language Technology Platform), https://github.com/HIT-SCIR/pyltp.

[17] ICTCLAS (Institute of Computing Technology, Chinese Lexical Analysis System), http://ictclas.nlpir.org/.

[18] M. Sun, X. Chen, K. Zhang, Z. Guo, Z. Liu, THULAC: An Efficient Lexical Analyzer for Chinese, http://thulac.thunlp.org/.

[19] C. Cortes, V. Vapnik, Support-Vector Networks, *Machine learning*, Vol. 20, No. 3, pp. 273-297, September, 1995.

Biographies

Yi-Jen Su received his Ph.D. degree in electrical engineering from National Cheng Kung University, Taiwan, in 2007. He is an Associate Professor in the Department of Computer Science and Information Engineering at Shu-Te University. He has published more than 70 papers in journals and conference proceedings. His current research interests include social network analysis, sentiment analysis, data mining, artificial intelligence, and image processing.

Chao-Ho (Thou-Ho) Chen received his Ph.D. degree in electrical engineering from National Taiwan University, Taiwan, in 1992. He has been the consultants of HuperLab. Co. and Vivotek Inc.. From 2002, he worked at National Kaohsiung University of Applied Sciences (changed as National Kaohsiung University of Science and Technology (NKUST) from 2018) in Taiwan, and has been the chairman (also the founder) of department of computer sciences and information and the director of computer and network center. He obtained the best paper awards of IIHMSP2007, NCWIA-2011, -2013, -2014, -2016, ITAOI-2017 (excellent paper award), and ITAOI-2018. His research interests mainly involve image/video processing and computer vision, and there are about 200 papers published and 30 ROC/USA patents.

Tsong-Yi Chen received his M.S. and Ph.D. degrees from the Illinois Institute of Technology, Chicago, USA. He is an assistant professor in the Department of Electronic Engineering, National Kaohsiung University of Science and Technology in Taiwan. He has published more than 100 papers in journals and conference proceedings since 2000. His research interests including digital image/video processing, computer vision, image processing, digital watermarking, visual surveillance, and expert systems.

Cheng-Chan Cheng received his M.S. degree in Department of Electronic Engineering from National Kaohsiung University of Applied Sciences, Taiwan, in 2018. His recent research interests include image/video processing and deep learning, and sentiment analysis.