Japanese Twitter Texts Sentiment Analysis Method based on an Improvement of Bi-directional LSTM and CNN Models

Zhengyi CHEN† and Iwao FUJINO‡

† Graduate School of Information and Telecommunication, Course of Information and Telecommunication Engineering, Master's Program, Tokai University 2-3-23 Takanawa, Minato-ku, Tokyo, 108-8619 Japan

‡ School of Information and Telecommunication Engineering, Department of Communication and Network Engineering, Tokai University 2-3-23 Takanawa, Minato-ku, Tokyo, 108-8619 Japan

E-mail: † 9ljnm001@mail.u-tokai.ac.jp, ‡ fujino@tokai.ac.jp

Abstract This paper proposes a method for classifying the sentiment polarity of Japanese Twitter texts using deep learning. In recent years, sentiment analysis has become one of the most important issues in social media mining. Since there are many opinions and evaluations about government policies and corporate products posted on Twitter, sentiment analysis of these texts is expected to play a significant role in policy making and marketing strategy planning. In this study, we confirmed that the introduction of an CNN and Bi-LSTM models provides higher classification accuracy compared to conventional RNN models or CNN model.

Keyword Sentiment Analysis, Natural Language Processing, CNN, Bi-LSTM

1. Introduction

With the rapid development of mobile network and social network service, users are increasingly willing to use social software such as Twitter, Facebook, to share and express their positions and opinions to current affairs or hot spot on the Internet. For these large amounts of text with emotional information generated every day. It has become an urgent need to analyze their sentiment accurately and efficiently in various industries.

Sentiment analysis is a common application of natural language processing (NLP) methods, and it is a hot research around world in recent years. Its task is to help users quickly acquire, organize and analyze relevant evaluation information, and analyze, process, summarize and reason about subjective texts with emotional overtones[1]. Sentiment analysis contains more tasks, such as sentiment classification, opinion extraction, opinion quiz and opinion summary.

Sentiment classification as an important branch of natural language processing, traditional sentiment classification is mainly based on sentiment lexicon and machine learning, and the latest method is based on deep learning.

Sentiment Lexicon method: This type of method uses a sentiment lexicon , domain dictionary and other manually written dictionary templates to obtain the final text sentiment polarity , where the most critical is to have a sentiment dictionary that can accurately evaluate the intensity of sentiment. For example, NRC Hashtag Dictionary and Sentiment140 Dictionary[2]. These two dictionaries were created specifically for Twitter text mining, Hashtag and emoticon are used as signals for a positive or negative tweet respectively.

Machine Learning method: The important point is selecting the suitable features to characterize text. Extracting the features from the manually annotated sentiment class of the text for the training and construction of the classifier. For example, in Pang Bo and Lee Lillion's paper, By extracting features such as lexicality and negation of text, text sentiment classification is performed using supervised methods[3].

Deep Learning method: Deep learning is an important branch of machine learning. In the initial. It has achieved significant breakthroughs and excellence in the field of image and speech recognition[4]. while in recent years Deep Learning has been increasingly used in text classification. Compared with traditional text classification methods, such as Naive Bayesian Model, K Nearest Neighbor, Support Vector Machine, deep learning does not need the artificial design features[5], but use the deep learning model to extract text features automatically. The speed of text classification is significantly improved, and the classification results is better than the result of traditional text classification.

In recent years, deep learning algorithms have achieved excellent results in the field of natural language processing. Among them, Convolutional Neural Network (CNN) makes full use of the structure of multi-layer perceptron and has good ability on learning complex, high-dimensional and non-linear mapping relations, it is widely used in image recognition tasks and speech recognition tasks, and achieved very good results[6][7]. In 2014, N. Kalchbrenner, E. Grefenstette and P. Blunsom proposed to apply CNN to natural language processing and designed a Dynamic Convolutional Neural Network (DCNN) model to process text of different lengths[8]; And in 2014, Yoon Kim proposed a model for English text classification[9]. He uses preprocessed word vectors as input and convolutional neural networks to achieve sentence-level classification tasks. In 2015, Y Zhang and B Wallace tested the sensitivity of CNN in sentence classification[10].

Although convolutional neural networks have made great breakthroughs in text classification, but convolutional neural networks focus more on local features and ignore the contextual meaning of words, which has an impact on the accuracy of text classification. So another component of the model in this paper, Bidirectional Long Short-Term Memory, is needed to solve the problem that the convolutional neural network model ignores the contextual meaning of words. Neural networks play an increasingly important role in the automatic learning and representation of features, and for serialized input, Recurrent Neural Network is able to integrate the neighborhood location information effectively to handle various tasks of natural language processing[11]. Long short-term memory (LSTM) is a special kind of RNN, which is mainly designed to solve the gradient disappearance and gradient explosion problems during the training of long sequences[12][13]. In short, LSTM can perform better in longer sequences than normal RNN[14]. There are several variants of recurrent neural network models, and the main one used for text classification is Bidirectional Recurrent Neural Network (RNN), because the semantic information of words in text is not only related to the information before the words, but also related to the information after the words[15].

In this paper, we propose a Bi-LSTM and CNN fusion model to deal with the problem of Twitter text sentiment analysis. BILSTM is used instead of traditional RNN and LSTM to solve the problem of gradient disappearance or gradient explosion, and Bi-LSTM also overcomes the problem that LSTM cannot fully consider the contextual meaning. The fusion of convolutional neural network and Bi-LSTM brings the advantage of both the extraction of local features by CNN and the global features of text by BILSTM, and the problem of ignoring contextual meaning in text classification by CNN is solved by Bi-LSTM, which improves the overall accuracy of feature fusion model in text classification.

2. Related Work

Deep neural networks have recently been shown to achieve highly competitive performance in many emotional classification tasks due to their abilities of exploring in a much larger hypothesis space. There are many excellent models have been presented.

In Z. Ding, R. Xia, J. Yu, X. Li and J. Yang's paper, Densely Connected Bidirectional LSTM with Applications to Sentence Classification 2018[16], they proposed a novel multilayer RNN model called densely connected bidirectional long short-term memory (DC-Bi-LSTM), which essentially represents each layer by concatenating its hidden states with those of all previous layers, and then passes the representation of each layer recursively to all subsequent layers. The performance of their DC-Bi-LSTM model has been shown to be stronger than the simple Bi-LSTM model. It is well known that RNN is one of the most popular architectures in NLP due to its cyclic structure which is well suited for processing variable length text. RNNs can use a distributed representation of words by first converting the tokens that make up each text into a vector to form a matrix. matrix includes two dimensions, one is the time-step dimension and the other is the feature vector dimension. Most existing models usually utilize one-dimensional (1D) max pooling operation or attention-based operation only on the time-step dimension to obtain a fixed-length vector.

Then, in P. Zhou, Z. Qi, S. Zheng, J. Xu, H. Bao and B. Xu's paper: Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling in 2016[17], they consider that features on the feature vector dimension are not independent of each other, and simply applying one-dimensional pooling operations independently on the time-step dimension may break the structure of the feature representation, applying two-dimensional (2D) pooling operation over the two dimensions may sample more meaningful features for sequence modeling tasks. So they propose to apply the two-dimensional maximum pooling operation to obtain a fixed-length representation of the text. Model Bi-LSTM with two-dimensional max pooling is thus derived. This model was experimented on six text classification tasks, including sentiment analysis, question classification, subjectivity classification, and newsgroup classification. Compared with other neural network models, excellent performance results were achieved in most of these tasks, and the highest accuracy was achieved especially in sentiment analysis.

3. CNN and Bi-directional LSTM model

Since our proposed model in this paper is a fusion of CNN model and Bi-directional LSTM model. At first, each individual model is described in section 3.1, 3.2 and 3.3. Then the architecture of our proposed combined model is shown in section 3.4.

3.1 Convolution Neural Network model

Convolutional Neural Network (CNN) is a feed-forward neural network with artificial neurons that respond to surrounding units within a portion of the coverage area, and is excellent for large image processing, as well as for text processing. It consists of a convolutional layer and a pooling layer. An example of CNN architecture as shown in Figure 1.

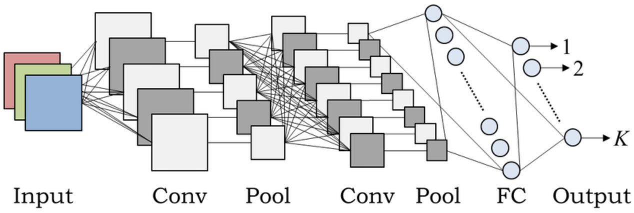

Figure 1. The architecture of CNN

3.1.1 Convolutional layer

The function of the convolutional layer is to perform feature extraction on the input data, and it contains several convolutional kernels inside. Each element of the convolution kernel corresponds to a weight coefficient and a bias vector. Each element of the convolutional kernel corresponds to a weight factor and a bias vector, similar to a neuron in a feedforward neural network. Each neuron in the convolutional layer is connected to multiple neurons in a region close to the previous layer, and the size of the region depends on the size of the convolutional kernel, which is called the "receptive field", and its meaning is analogous to the receptive field of the visual cortical cells. There are M filters in the convolution layer, the convolution calculating through a filter and the width $\boldsymbol{k}$ of sliding convolution kernel, a filter $\boldsymbol{F_m}$ (1≤m≤M) generates the feature map as follows:

$$\boldsymbol{y_i^m = f(x_{i:i+w-1} \otimes w^m + b^m)} \qquad (1)$$

$\boldsymbol{w^m} \in \boldsymbol{R^{k*d}}$ is weight matrix of the filter $\boldsymbol{F_m}$. $\boldsymbol{b^m}$ is a bias of the filters $\boldsymbol{F_m}$. $\boldsymbol{d}$ is the dimension of the word vector, $\otimes$ is tensor product, indicates the convolution operation. $\boldsymbol{x_{i:i+k-1}}$ shows that the filter $\boldsymbol{F_m}$ extract feature from $\boldsymbol{x_i}$ to $\boldsymbol{x_{i+k-1}}$, $\boldsymbol{f}$ is a non-linear activation function. Here it is the ReLU function. $\boldsymbol{y_i^m}$ represents the local eigenvectors obtained by the convolution operation. As the filter slides from top to bottom relying on a step size of 1, it walks through the entire sentence matrix and finally obtains the set of local eigenvectors.

3.1.2 Maxpooling layer

After feature extraction in the convolutional layer, the output features are passed to the pooling layer for feature selection and information filtering. Maxpooling has local invariance and can extract significant features while reducing the parameters of the model, thus reducing the overfitting of the model. Because only significant features are extracted and insignificant information is discarded, the parameters of the model are reduced, which can alleviate the overfitting to some extent.

3.2 Bi-directional LSTM model

Since RNN can learn the input of any time sequence, but as the input increases, it is difficult to learn the relationship between connections, which generates the problem of long dependence, that is the perception of some nodes in the previous time decreases, and then the phenomenon of gradient disappearance or gradient explosion will occur.

LSTM can solve the above problems of RNN, the core of which is to use memory cells to remember long-term historical information and manage it with a gate mechanism, the gate structure does not provide information, but is only used to limit the amount of information, adding gates is actually a multi-level feature selection method.

Forget gate: The function of the forget gate is to decide what information should be discarded or retained. The information from the previous hidden state and the current input information are passed to the sigmoid function at the same time, and the output value is between 0 and 1. The closer to 0 means the more it should be discarded, and the closer to 1 means the more it should be retained. The structure is shown in Figure 2.

$$\boldsymbol{f_t = \sigma(W_f * [h_{t-1}, x_t] + b_f)} \qquad (2)$$

Input gate: The input gate is used to update the cell state.

First, pass the information of the hidden state of the previous layer and the current input information to the

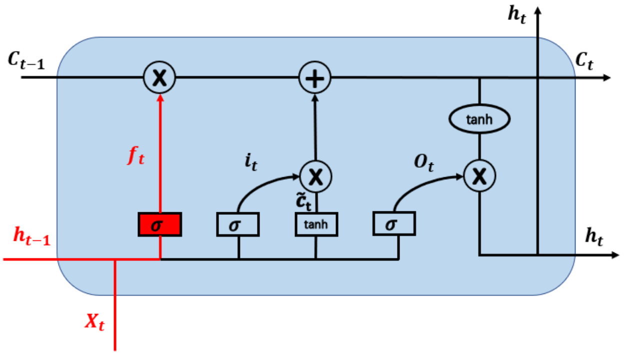

Figure 2. The architecture of forget gate

sigmoid function. Adjust the value between 0 and 1 to decide which information to update. 0 means not important, 1 means important. Secondly, the information of the hidden state of the previous layer and the current input information are passed to the tanh function to create a new candidate value vector. Finally, the output value of sigmoid is multiplied by the output value of tanh. The output value of sigmoid will determine which information in the output value of tanh is important and needs to be preserved. The structure is shown in Figure 3.

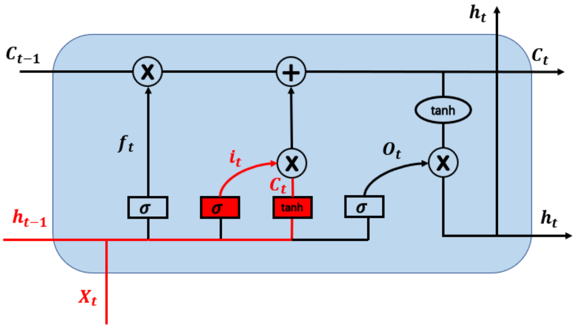

Figure 3. The architecture of input gate

$$i_t = \sigma(W_i * [h_{t-1}, x_t] + b_i) \tag{3}$$

$$\tilde{c}_t = \tanh(W_c * [h_{t-1}, x_t] + b_c) \tag{4}$$

The current state: The next step is to calculate the cell state. First, the cell state of the previous layer is multiplied by the forgetting vector point by point. If it is multiplied by a value close to 0, it means that this information needs to be discarded in the new cell state. Then add this value to the output value of the input gate point by point, and update the new information found by the neural network to the cell state. At this point, the updated cell state is obtained. The structure is shown in Figure 4.

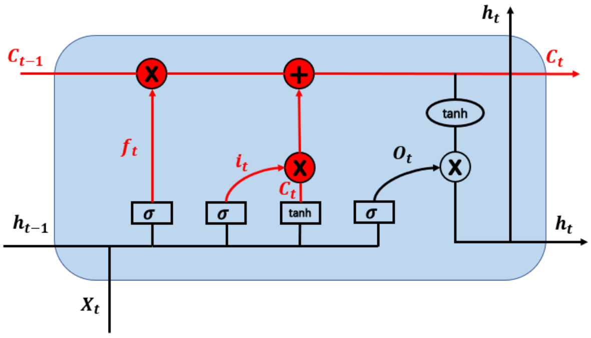

Figure 4. The architecture of current state

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \widetilde{C_t} \tag{5}$$

The output gate: The output gate is used to determine the value of the next hidden state. The hidden state contains the information previously entered. First, we pass the previous hidden state and current input to the sigmoid function, and then pass the newly obtained cell state to the tanh function. Finally, the output of tanh is multiplied by the output of sigmoid to determine the information that the hidden state should carry. Then the hidden state is used as the output of the current cell, and the new cell state and the new hidden state are passed to the next time step. The structure is shown in Figure 5.

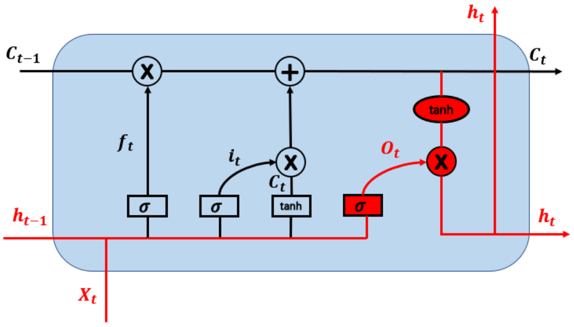

Figure 5. The architecture of output gate

$$O_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{6}$$

$$h_t = o_t \cdot \tanh(C_t) \tag{7}$$

W_f, W_i, W_c, W_o are the weight of the LSTM. b_f, b_i, b_c, b_o are the bias of the LSTM. h_t is the hidden state in time t. σ is the activation function sigmoid, $tanh$ is the hyperbolic tangent function.

Although LSTM solves the problem of gradient disappearance and gradient explosion of RNN, LSTM can only learn the information before the current word, but it cannot use the information after the current word.

Since the semantics of a word is not only related to the previous historical information, but also closely related to the information after the current word, this paper uses Bi-LSTM instead of LSTM to solve the problem of gradient disappearance or gradient explosion, and also to fully consider the information of the current word's preceding and following languages. By using Bi-LSTM to learn the sentence matrix, the obtained text features are global in nature and fully consider the contextual information of the word in the text.

3.3 Attention Mechanism

"Sequence to Sequence Learning with Neural Networks[18]" introduces an RNN Seq2Seq model based on an Encoder and a Decoder to build a neural network-based End-to-End machine translation model, where Encoder encodes the input **X** into a fixed-length hidden vector **Z**, and Decoder decodes the target output Y based on the hidden vector **Z**. This is a very classical sequence-to-sequence model, but suffers from two obvious problems.

1. compressing all the information of input **X** into a fixed-length hidden vector **Z**, ignoring the length of input **X**. The performance of the model drops sharply when the input sentence length is very long, especially longer than the initial sentence length in the training set.

2. It is unreasonable to encode the input **X** into a fixed length and assign the same weight to each word in the sentence. For example, in machine translation, between the input sentence and the output sentence, it is often the case that one or several words in the input correspond to one or several words in the output. Therefore, assigning the same weight to each word of the input, which does not distinguish, tends to be a degradation of the model performance.

The same problem exists in the field of image recognition, where convolutional neural network CNNs do the same for each region of the input image, which does not distinguish, especially when the size of the processed image is very large. Therefore, in 2015, Dzmitry Bahdanau in "Neural machine translation by jointly learning to align and translate[19]" proposed the Attention Mechanism for different input **X** parts by assigning different weights to them, thus achieving soft differentiation.

In 2017, Z. Yang proposed to use Attention Mechanism on the sentiment classification task of the text, and then made a good classification effect[20]. In this paper, we use Global Attention. Global Attention is one type of the Attention Mechanism. The same as the traditional Attention model. All the hidden states are used to compute the weights of the Context vector, that is the variable-length alignment vector $\boldsymbol{a_t}$, whose length is equal to the length of the input sentence at the encoder side. The structure is shown in Figure 6.

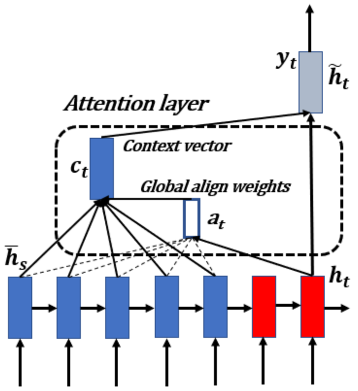

Figure 6. The architecture of Global Attention

At moment **t**, hidden state $\boldsymbol{h_t}$ based on decoder, the hidden state $\boldsymbol{h_s}$ of the source. computes a variable-length vector of hidden alignment weights $\boldsymbol{a_t}$ with the following equation:

$$a_t(s) = \frac{exp\left(score(h_t, \bar{h}_s)\right)}{\sum_{s'} exp\left(score(h_t, \bar{h}_{s'})\right)} \qquad (8)$$

Among them, ***score*** is a function used to evaluate the relationship between $\boldsymbol{h_t}$ and $\boldsymbol{h_s}$, that is the alignment function, which is generally calculated in three ways with the following formula:

$$score(h_t, \bar{h}_s) = \begin{cases} h_t^T \bar{h}_s & dot \\ h_t^T W_a \bar{h}_s & general \\ V_a^T tanh(W_a[h_t{:}\,\bar{h}_s]) & concat \end{cases} \qquad (9)$$

Once the alignment vector $\boldsymbol{a_t}$ is obtained, the context vector $\boldsymbol{c_t}$ can be obtained by weighted averaging.

3.4 Bi-LSTM and CNN model

As shown in Figure 7, the model proposed in this paper based on the above CNN model, Bi-LSTM model and the Attention Mechanism. The CNN are mainly used to extract different local features of words between sentences, while the Bi-LSTM is used to get sentence context semantic information.

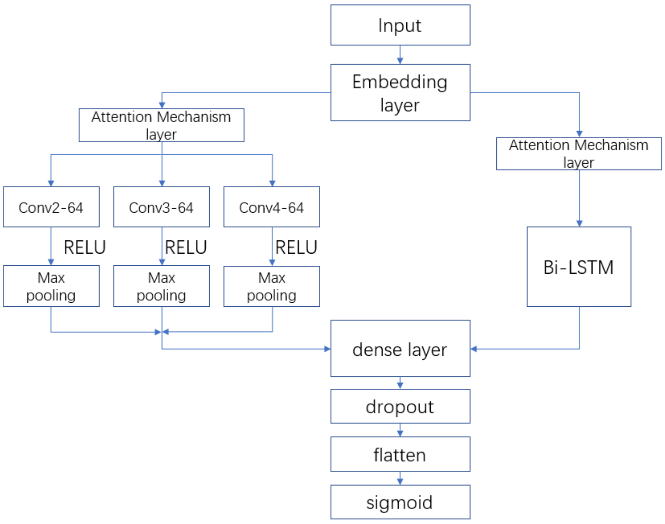

Figure 7. The architecture of CNN and Bi-LSTM model

The first layer of the convolutional neural network part is the word embedding layer, which takes as input the sentence matrix of the word embedding layer, the column of the matrix are the dimensions of the word vector, and the row of the matrix is the sequence length. The second layer is the attention mechanism layer, mainly to extract important word information between sentences. The third layer is the convolution layer, which performs convolution operations to extract local features. The dimension of word embedding vector sets 300, the three selected filter sizes are 2,3,4 with 64 feature maps each, the stride set 1, the padding sets VALID. The fourth layer performs a maximum pooling operation, extracts key features, discards redundant features, generates a fixed dimensional feature vector, and stitches together the features output from the three pooling operations as part of the input features of the fully connected layer.

The first layer of the Bi-LSTM part is the word embedding layer, the sentence matrix of the embedding layer is used as the input, and the dimension of each word vector is set to 300 dimensions; The second layer is the attention mechanism layer, mainly to extract important word information between sentences. The third layer and the fourth layer are both hidden layers, the size of the hidden layer is 32, the current input is correlated with both the front and back sequences, the input sequences are input to the model from two directions, the historical information and the future information of the two directions are saved through the hidden layer, and finally the output of the two hidden layers are partially spliced to get the final Bi-LSTM output.

Then we combine the output feature of the CNN with the output of the Bi-LSTM before the dense layer. And make the concatenated feature as the input of the dense layer. After that enter the dropout layer and the flatten layer. The direct effect of dropout layer is to reduce the number of intermediate features, thus reducing redundancy, increasing the orthogonality between individual features at each layer (the sparse view of data representation also supports this interpretation precisely). The Flatten layer is used to "flatten" the input. That is, to flatten a multi-dimensional input into one dimension.

Put the output from the dense layer as the input into the sigmoid function layer. Finally get the prediction results Y.

$$\boldsymbol{Y = \sigma(Wh + b)} \qquad \textbf{(10)}$$

The loss function is Cross Entropy Loss Function, the input data is the output of the sigmoid function.

$$\boldsymbol{Loss = -\frac{1}{N}\sum_{i}[y_i \cdot log(p_i) + (1 - y_i) \cdot log(1 - p_i)]} \qquad \textbf{(11)}$$

4. Experiments

4.1 Experimental Environment

The experimental environment of our paper is as follows: the operating system is Windows10, the CPU is Intel Core i7-9750H, the GPU is GeForce RTX 2060, RAM is 16GB， the development environment is Keras 2.3.1, and the development tool uses Visual Studio Code.

4.2 Experimental Dataset

The dataset of this paper is divided into two parts. First part is the Japanese version of Wikipedia. After data pre-processing, The data size is about 2.86 GB. Then we get the word vector by the skip-gram of the word2vec. Second part is the Japanese texts. 5059 Japanese texts on twitter were collected as the dataset about iphone6[21]. We set the number 1 to represent the positive emotion and the number 2 to represent the negative emotion. There are 2527 positive texts and 2532 negative texts in the dataset. And 95% of the dataset will be used as training data and 5% as test data. Samples of the Japanese texts are:

Positive dataset samples:

iPhone6は画面デカくて感動している。
iPhone6の画質良すぎる

Negative dataset samples:

iPhone6ってどうしても慣れないな… もちにくい、、誤字率最近つらい(すべてiPhone6のせい)

First, we remove punctuation from each sample, then we use MeCab to split the words. MeCab splitting returns a generator that cannot be tokenized directly, so we convert the splitting result into a list and index it so that the text evaluated in each case becomes a segment of indexed numbers corresponding to the words in the pretrained word embedding model.

After we convert the text into tokens, the length of each index is not equal, so in order to facilitate the training of the model we need to standardize the length of the index, above we choose the length 236 which can cover 95% of the length of the training samples, next we padding and truncating, we generally use the 'pre' method, which will fill in the front of the text index 0, because according to the practice of some research materials, if the text index is filled in the back of 0, it will cause some adverse effects on the model.

4.3 Word embedding

This paper uses Japanese Wikipedia data to train the skip-grams to obtain Japanese word vectors. Since the wiki corpus is large enough to train high-quality word vectors. And then load the trained word embedding model with embedding layer

4.4 Parameters settings

For a good model, the setting of parameters is also very important. The selection of experimental parameters has a crucial impact on the final experimental results. After repeated parameter tuning experiments, the following parameters were determined.

In the CNN network model, Word vector dimension set 300, convolution kernel size sets 2, 3, 4, the number of each convolution kernel set 64. Activation function selects ReLU, stride sets 1. For the Bi-directional LSTM layers, the number of hidden units set 32. Optimization function is Adam, learning rate sets 0.001, epoch sets 40. The parameter of dropout layer set to 0.5. The sigmoid function is used in the output layer for classification.

Table 1. Result of accuracy for different model

Model	CNN	GRU	LSTM	Bi-LSTM	CNN and Bi-LSTM
precision	77.05%	67.41%	70.30%	76.28%	79.21%
recall	65.41%	61.92%	61.92%	69.19%	69.77%
f1	70.75%	64.55%	65.84%	72.56%	74.19%
accuracy	81.62%	76.88%	78.16%	82.21%	83.50%

4.5 Results and performance evaluation

The experimental results are shown in Table 1. When comparing with the CNN model with the attention layer, we adjust the parameters of the CNN model to the same parameters as the CNN model in the proposed model in this paper. When comparing with the Bi-LSTM model with the attention layer, the parameters are adjusted in the same way. We find that, every experiment shows that the accuracy of the proposed model in this paper is higher than the others, so the result shows that the superiority of the CNN and Bi-LSTM model is demonstrated.

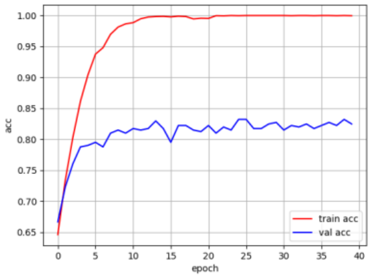

Figure 8. Accuracy of CNN and Bi-LSTM model

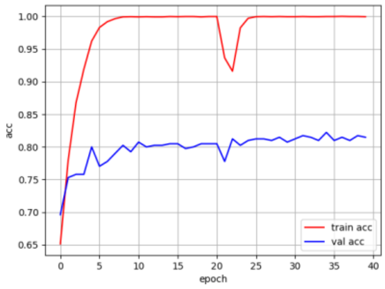

Figure 9. Accuracy of Bi-LSTM model

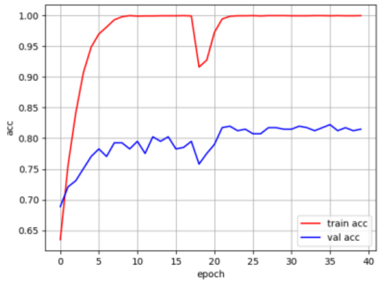

Figure 10. Accuracy of CNN model

As shown in the Figure8, 9, 10. Comparing the three models, the fusion model has a slower convergence rate but higher accuracy than the single CNN model and single Bi-LSTM model on the test set.

5. Conclusion

In this paper, we proposed a model based on Convolutional Neural Network and Bi-LSTM Network for sentiment analysis of the Japanese texts from the Twitter. This model can not only effectively extract local features of text by using convolutional neural networks, but also use Bi-LSTM to take into account the global features of the text, the contextual semantic information of the words is fully considered. Compared with other CNN and Bi-LSTM type models, this paper adds a attention mechanism in front of CNN model and Bi-LSTM model, also adds dropout layer and flatten layer in the fully connected layer, the performance has improved than the previous similar models. Compared with the single CNN model and the single Bi-LSTM model, Overall Accuracy about the classification of the proposed model is better than the others. The results show that the proposed model in this paper outperforms the compared models in terms of classification accuracy, and the model in this paper effectively improves the accuracy of Japanese text classification. In the future we will continue to explore models with higher performance on Japanese text analysis.

However, the depth of CNN used in this paper is shallow, in the future we will study the effect of fusion of deeper CNN with Bi-LSTM or other models on text sentiment classification in the future. Secondly, we can combine the advantage of the traditional Machine Learning or Sentiment Lexicon with the Deep Learning, improve the accuracy of the model on text sentiment analysis much further.

Acknowledgment

This work was partially supported by JSPS KAKENHI Grant Number JP20K04943.

References

[1] Liu Bing, (2010). "Sentiment Analysis and Subjectivity" (PDF). In Indurkhya, N.; Damerau, F. J. (eds.). Handbook of Natural Language Processing (Second ed.)

[2] Saif M. Mohammad, Svetlana Kiritchenko, Xiaodan Zhu, “NRC-Canada: Building the State-of-the-Art in Sentiment Analysis of Tweets” In Proceedings of the seventh international workshop on Semantic Evaluation Exercises (SemEval-2013), June 2013, Atlanta, USA.

[3] Pang Bo, Lee Lillion, “Opinion Mining and Sentiment Analysis”, Foundations and Trends in Information Retrieval January 2008.

[4] Lecun Y, Bengio Y, Hinton G. Deep learning [J]. Nature, 2015, 521(7553):436.

[5] Li J, Cao Y, Wang Y, “Online Learing Algorithms for double-weighted least squares twin bounded support vector machines”[J]. Neural Processing Letters,2017,45(1):319-339.

[6] O.Abdel-Hamid, Abdel.rahman Mohamed, Hui Jiang, Li Deng “Convolutional Neural Networks for Speech Recognition”, Computer Science IEEE/ACM Transactions on Audio, Speech, and Language Processing 2014.

[7] Jianlong Fu, Heliang Zheng, Tao Mei. “Look Closer to See Better: Recurrent Attention Convolutional Neural Network for Fine-Grained Image Recognition” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 4438-4446.

[8] Nal Kalchbrenner, Edward Grefenstette and Phil Blunsom, “A Convolutional Neural Network for Modelling Sentences”, arXiv:1404.2188 [cs.CL], pp. 10-22, 2010. https://arxiv.org/abs/1404.2188.

[9] Yoon Kim, “Convolutional neural networks for sentence classification”, [EB/OL].[2014-09-03]. http://emnlp2014.org/papers/pdf/EMNLP2014181.pdf.

[10] Y. Zhang and B. Wallace, ‘‘A sensitivity analysis of (and practitioners’ guide to) convolutional neural networks for sentence classification,’’ 2015, arXiv:1510.03820.[Online].Available:http://arxiv.org/abs/1510.03820.

[11] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, Yoshua Bengio, “Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation”, [EB /OL].[2014-09-03]http://www.aclweb.org/anthology/D/D14/D14-1179.pdf.

[12] D. Li, J. Qian, “Text sentiment analysis based on long short-term memory”, [C]//IEEE International Conference on Computer Communication and the Internet. IEEE, 2016:471-475.

[13] A. Graves, “Supervised Sequence Labelling with R ecurrent Neural Networks”, [M]. Heidelberg: Springer, 2013:35-42

[14] A Sherstinsky, “Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network”, Physica D: Nonlinear Phenomena, 2020 – Elsevier.

[15] A. Mnih, GE. Hinton, “A scalable hierarchical distributed language model”, Advances in neural information processing, 2008-papers.nips.cc

[16] Z. Ding, R. Xia, J. Yu, X. Li and J. Yang, “Densely Connected Bidirectional LSTM with Applications to Sentence Classification”, CCF International Conference on Natural Language Processing and Chinese Computing, 2018.

[17] P. Zhou, Z. Qi, S. Zheng, J. Xu, H. Bao and B. Xu, “Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling”, arXiv preprint arXiv:1611.06639 (2016)

[18] Lya. Sutskever, Oriol. Vinyals and Le. Quoc V, “Sequence to Sequence Learning with Neural Networks”, Advances in Neural Information Processing Systems 27 (NIPS 2014)

[19] D. Bahdanau, K. Cho and Y. Bengio, “Neural machine translation by jointly learning to align and translate”, arXiv preprint arXiv:1409.0473, 2014-arxiv.org

[20] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola and E. Hovy, “Hierarchical Attention Networks for Document Classification”, in Proc.Conf.North Amer.ChapterAssoc. Comput. Linguistics, Hum.Lang.Technol, Jun.2016, pp. 1480–1489

[21] Ikuo Keshi, Yu Suzuki, Koichiro Yoshino, Satoshi Nakamura. Semantically Readable Distributed Representation Learning for Social Media Mining.Proceedings of the International Conference on Web Intelligence (WI ’17), pp.716-722, Leipzig, Germany, August 2017.