A Novel Attention Based CNN Model for Emotion Intensity Prediction

Hongliang Xie, Shi Feng(✉), Daling Wang, and Yifei Zhang

School of Computer Science and Engineering, Northeastern University, Shenyang, China
x123872842@163.com,
{fengshi,wangdaling,zhangyifei}@cse.neu.edu.cn

Abstract. Recently, classifying sentiment polarities or emotion categories of social media text has drawn extensive attentions from both academic and industrial communities. However, limited efforts have been paid for emotion intensity prediction problem. In this paper, we propose a novel attention mechanism for CNN model that associates attention based weights for every convolution window. Furthermore, a new activation function is incorporated into the full-connected layer, which can alleviate the small gradient problem in function's saturated region. Experiment results on benchmark dataset show that our proposed model outperforms several strong baselines and achieves comparable performance with the state-of-the-art models. Unlike the reported models that used different neural network architectures for different emotion categories, our proposed model utilizes a unified architecture for intensity prediction.

Keywords: Emotion intensity prediction · CNN · Attention mechanism

1 Introduction

The social media platforms such as Facebook, Twitter and Instagram have aggregated huge amount of personal feelings and altitudes. In recent years, classifying sentiment polarities (Positive/Negative) or emotion categories (Anger/Fear/Joy/Sadness) of the user generated content has drawn extensive attentions from both academic and industrial communities. However, in text we not only convey the emotion category of our feeling, but also the intensity of that emotion. Therefore, the sentences with the same emotion category may have quite different intensities, as shown in the following example tweets.

Tweet A: *Just got back from seeing @GaryDelaney in Burslem. AMAZING!! Face still hurts from laughing so much #hilarious.*
Tweet B: *What a great training course, lots of photos, fun and laughter. Photo's will be up soon #Boostercourse #fun.*

The above two examples are selected from WASSA-2017 emotion intensity detection dataset, and both of them are associated with emotion label 'Joy'. We can easily observe that Tweet A express Joy much more intensively than Tweet B. In the original dataset, the gold intensity label of Tweet A is 0.980 while Tweet B's gold score is 0.740. The traditional emotion category classification methods could not distinguish the

M. Zhang et al. (Eds.): NLPCC 2018, LNAI 11108, pp. 365–377, 2018.
https://doi.org/10.1007/978-3-319-99495-6_31

intensity difference of these tweets. In this paper, we regard emotion intensity predicting as a regression problem. The goal of the algorithm is to predict real-valued score ranging from 0 to 1, which refers to the degree or amount of this emotion when given the input sentence and the emotion label of this sentence expressed. A score of 1 means that this sentence expresses the highest intensity emotion, and the score of 0 means that this sentence expresses the lowest intensity emotion.

Automatically determining the intensity of emotion felt by speaker has potential applications in commerce, public health, intelligence gathering, and social welfare [1]. The previous studies have achieved promising results for classifying the social media text according to their embedded emotions. However, limited efforts have been paid for tweet emotion intensity prediction problem, which has brought in some brand new challenges. Firstly, emotion intensity prediction is a finer-grained problem than emotion category classification and this new task needs to capture the nuances of different emotional words and provide more effective feature representation for tweets. Secondly, the length limitation of tweets results in sparseness problem in the feature space and sets up obstacles for extracting effective features. Thirdly, Twitter has an extremely large user base which leads to rich textual content, including nonstandard language, creatively spelled words (e.g. happee), and hash-tagged words (e.g. #luvumon) [1]. These informal writing styles of tweets increase the difficulties for understanding the semantics in short text.

To tackle these challenges, in this paper, we propose a novel attention mechanism for Convolutional Neural Network with a revised activation function that are suitable for the tweet emotion intensity prediction problem. Deep learning method has already shown some promising results on this topic. For example, Goel et al. leveraged the ensemble CNN and RNN models to achieve the best performance in WASSA-2017 shared task on emotion intensity [2]. However, they had to train different models for different emotion categories and the basic models they used treated all words equally in the modeling of sentences. Different from the existing methods, our proposed CNN model associates attention based weights for every convolution window. Furthermore, a new activation function in full-connected layer is proposed which can alleviate the small gradient problem in saturated region. In summary, our key contributions are as follows.

- We propose a novel attention mechanism for CNN which makes our model pay more attention to the words contributing to emotion intensity prediction.
- We introduce a novel activation function in full-connected layer for regression problem whose result is real-valued and ranges from 0 to 1.
- We conduct experiment on benchmark dataset. Experiment results show that our proposed model outperforms several strong baselines and achieves comparable performance with the state-of-the-art non-ensemble models. Unlike that reported models used different neural network architectures for different emotion categories, our proposed model utilizes a unified architecture for intensity prediction.

2 Related Work

Modeling emotion intensity has attracted more and more attentions from researchers. The WASSA-2017 EmoInt is a shared task to predict emotion intensity value of the given English tweet with emotion category label, which was held in conjunction with EMNLP-2017. And SemEval-2018 also organizes a shared task to predict emotion intensity value whose dataset has English, Arabic, and Spanish tweets. Continuously holding these shared tasks show that the academic community is gradually aware of the importance of this problem.

Methods for solving emotion intensity prediction can be divided into two categories. One is traditional feature-based methods which rely on a set of features selected from preprocessing steps. And they usually need another external resources. Mohammad et al. created a lot of features includes Word N-grams, Character N-grams, Word Embedding and many Affection Lexicons features [1]. Then they used a L2-regularized L2-loss SVM regression model to predict the emotion intensity. Köper et al. used affective norms and automatically extended resources to build features, and then utilized random forest classifier to predict the emotion intensity [3]. Duppada et al. leveraged the word embedding and emoji embedding [4] to build features, and then used many regression algorithms to predict the emotion intensity [5]. The shortage of those methods are that they rely on manually created affection lexicons and preprocessing steps.

Another category of methods are based on the deep learning technology. Goel et al. used the feed-forward neural network, multitask deep neural network and sequence modeling using CNNs and LSTMS to predict the emotion intensity of tweets [2]. And they also employed the ensemble results of those models to get the state-of-art result in the WASSA-2017 EmoInt shared tasks. John el al. fed affect clues, sentiment polarity and word embedding into the deep neural network to predict the emotion intensity [6]. Lakomkin et al. used the character-level and word-level recurrent neural network models to predict the emotion intensity, and showed that the effectiveness of using the character-level models to model the noisy and short texts [7]. The shortage of these methods are that they treat all word equally in the modeling of sentence. But when we judge the emotional intensity of sentences in reality, different words have quite different effects. Our experiments demonstrate that we can use attention mechanism to associate different words with different weights in CNN framework, and the proposed attention mechanism can indeed help improve the performances of models in predicting emotion intensity.

3 Proposed Method

Given a sentence *s* and its emotion label, the emotion intensity prediction is a task to get a real value ranges from 0 to 1 which is in proportion to the emotion intensity. The schematic overview of our attention based convolutional neural network is shown in Fig. 1. Given a sentence *s*, we use the pre-trained word embedding matrix transform it into a matrix $\mathbf{X}$. Secondly, we use a LSTM layer encodes this sentence into a vector representation $\mathbf{v}_s$ from the matrix representation $\mathbf{X}$. Then we use $\mathbf{v}_s$ and $\mathbf{X}$ to compute attention score $\mathbf{v}_a$ by using general method [8]. Each element in the $\mathbf{v}_a$ represents the

attention score of each word in s. Next we can scale our word embedding in each convolutional window by the corresponding score. But to make the expectation of convolutional output unchanged, we need to apply the softmax function to the attention scores in that convolutional window to transform it into a probability distributions and multiply the score with the convolutional window size to get the genuine attention score in that convolutional window. Due to the same word in different convolutional windows having different attention scores, it will scale with different factors. So we need transform the $\mathbf{X}$ into the genuine convolutional input $\mathbf{Z}$. After getting the genuine input, we use a CNN layer extracts the features of the sentences which denotes as $\mathbf{v_f}$. Finally, we use a fully-connected layer and a revised activation function to get the predicted emotion intensity.

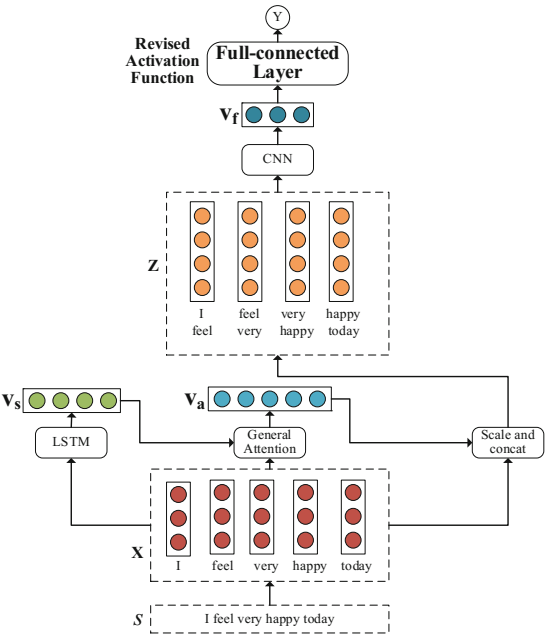

Fig. 1. Schematic overview of attention based Convolutional Neural Network

3.1 Input Representation

Given a sentence $s = (\omega_1, \omega_2, \ldots, \omega_n)$. We transform every word ω_i into a real-valued vector through by looking up work embedding matrix to provide lexical-semantic features. The shape of the word embedding matrix $\mathbf{W_v}$ is $|V| \times d_w$, where $|V|$ is the vocabulary size and d_w is the embedding size. So we map every word ω_i into a row vector $\mathbf{W_v^i} \mathbb{R}^{1\times d_w}$.

We can get the word embedding matrix from the pretrained word vectors. A common tool for training word embedding is Word2Vec [9, 10]. This tool leverages a lot of unsupervised domain corpus as input, and gets the trained word embedding matrix for

every word in the dataset. And the trained word vectors can capture a large number of precise syntactic and semantic word relationships.

Alternatively, we can use the random initialized word embedding matrix. We can fine-tune this word embedding matrix when we train CNN model. Kim demonstrates that random initialized word embeddings can also get a good results [11].

There is a big difference between the vocabulary of WASSA-2017 EmoInt training set and test set. The vocabulary size information of training set and test set is in the Table 1. If we use the random initialized word embedding matrix, many words that only appears in test set cannot be trained in the training phase, which has a bad effect on our experimental results. Therefore, we use pretrained word vectors to bridge the vocabulary gap and alleviate difficulties introduced by the informal writing styles of tweets, such as the creatively spelled words, emojis and so on. We use the public available embeddings [12] which were trained on 400 million tweets for ACL WNUT 2015 shared task.

Table 1. The vocabulary size about training set, test set and the overlap vocabulary size

Emotion	Training set	Test set	Overlap in Training & Test set
Anger	3345	3127	1129
Fear	4253	3664	1445
Joy	3263	3057	1110
Sadness	3605	3102	1189

3.2 Attention Mechanism

We propose a novel attention mechanism for CNN to make our model treat different words with different weights. The LSTM based sentence representation and the word vector is utilized to compute the weight on this word.

Sentence Representation

We use a LSTM layer [13] to encode our input matrix $\mathbf{X}$ to a vector $\mathbf{v}_s$ which is further used to compute the attention weight of each word in the sentence. The formulas are as follow:

$$\mathbf{i}_t = \text{sigmoid}\left(\mathbf{W}_\mathbf{i}\mathbf{h}_{t-1} + \mathbf{U}_\mathbf{i}\mathbf{X}_t + \mathbf{b}_\mathbf{i}\right) \tag{1}$$

$$\mathbf{f}_t = \text{sigmoid}\left(\mathbf{W}_\mathbf{f}\mathbf{h}_{t-1} + \mathbf{U}_\mathbf{f}\mathbf{X}_t + \mathbf{b}_\mathbf{f}\right) \tag{2}$$

$$\mathbf{o}_t = \text{sigmoid}\left(\mathbf{W}_\mathbf{o}\mathbf{h}_{t-1} + \mathbf{U}_\mathbf{o}\mathbf{X}_t + \mathbf{b}_\mathbf{o}\right) \tag{3}$$

$$\widetilde{\mathbf{c}}_t = \tanh\left(\mathbf{W}_\mathbf{c}\mathbf{h}_{t-1} + \mathbf{U}_\mathbf{c}\mathbf{X}_t + \mathbf{b}_\mathbf{c}\right) \tag{4}$$

$$\mathbf{c}_t = \mathbf{f}_t * \mathbf{c}_{t-1} + \mathbf{i}_t * \widetilde{\mathbf{c}}_t \tag{5}$$

$$\mathbf{h}_t = \mathbf{o}_t * \tanh\left(\mathbf{c}_t\right) \tag{6}$$

the above formula, $\mathbf{X_t}$ is the t-th word vector in the input matrix $\mathbf{X}$. The weights $\mathbf{W_i}, \mathbf{U_i}, \mathbf{W_f}, \mathbf{U_f}, \mathbf{W_o}, \mathbf{U_o}, \mathbf{W_c}, \mathbf{U_c}$ and bias $\mathbf{b_i}, \mathbf{b_f}, \mathbf{b_o}, \mathbf{b_c}$ are the parameters of LSTM layer. And $\mathbf{c_t}, \mathbf{h_t}$ are the value of cell state and hidden state at timestep t. We use the last hidden state $\mathbf{h_n}$ as the vector representation of that sentence which is also denoted as $\mathbf{v_s}$.

Attention in Convolutional Window

After get the vector representation of sentence, we need to use the sentence vector $\mathbf{v_s}$ and every word vector compute the word attention score. We use the general method [8] to compute the attention weight. In other words this process can be expressed as follow:

$$\mathbf{v_a} = \mathbf{X}\mathbf{W}\mathbf{v_s} \tag{7}$$

In the Eq. 7, we get a vector $\mathbf{v_a}\mathbb{R}^n$. Each element in the vector represents the attention score of each word in the sentence respecting with the sentence vector $\mathbf{v_s}$.

As we all know, convolutional neural networks share weight parameters through the convolution kernel. Due to this fact, we cannot apply attention weight to the convolutional weights directly. And the convolution operation is equal to the sum of each element in the tensor produced by the element-wise product of the input matrix and the convolutional weights. So we can apply the attention weights to the input matrix.

In our model, we use attention mechanism to each convolutional windows. In other words, we multiply every word vector in the convolutional windows by the corresponding attention weight. To make the expectation of the convolution result unchanged, we need to transform the weights in the vector slice corresponding to this convolutional windows to a probability distribution and then we multiply the transformed probability distribution by the convolutional window size to get the genuine attention weight vector. At last, we can multiply the word vector by the genuine attention weight to get the genuine convolution layer input. This process can be expressed as follow:

$$\boldsymbol{\alpha}_i = l * \text{softmax}\big(\mathbf{v_a}[i{:}i+l]\big), i \in \{0, 1, \ldots, n-l\} \tag{8}$$

$$\mathbf{Z_i} = \left[\boldsymbol{\alpha}_{i0} * \mathbf{X}_{(i+0)}, \boldsymbol{\alpha}_{i1} * \mathbf{X}_{(i+1)}, \ldots, \boldsymbol{\alpha}_{i(l-1)} * \mathbf{X}_{(i+l-1)}\right] \tag{9}$$

In the above equations, we transform the input matrix $\mathbf{X}$ to the genuine convolutional layer inputs $\mathbf{Z}$. Each row of the matrix $\mathbf{Z}$ is corresponded to the input of one convolution operation step. Because sentence length is n and the convolutional window size is l. So the number of rows of matrix $\mathbf{Z}$ is n-l + 1. And $\mathbf{Z_i}$(the i-th row of the matrix $\mathbf{Z}$) is a row vector which dimension is $l \times d_w$. From the above Eq. 8, we can easily know that attention weight of the same word is different when it is in different convolutional operation step. That is the reason why should make the new matrix $\mathbf{Z}$.

CNN uses max pooling to extract the significant feature in that convolutional windows. Inspired by adding attention to word embedding [14], our proposed attention mechanism is able to scale the word embedding in the convolutional window according to their genuine attention score. That will amplify the related word embedding which is more important for predicting emotion intensity and shrink unrelated embedding. So we can extract the significant feature more easily than original CNN architecture.

Convolutional Layer

In the above section, we get the genuine convolutional layer's input matrix **Z**. Then we can use the convolution operation on this matrix. But we need to notice that we have considered the convolutional window size in the process of transforming **X** to the genuine input matrix **Z**. So the convolutional window in this convolutional filter is 1. And the input channel in our model is also 1. Then we get our convolutional filters has the shape $[l \times d_w, 1, 1, k]$. The parameter k is a hyperparameter which controls the number of features extracted for each convolution window.

We use max-pooling over the dimension corresponding to convolutional window after the convolution operation. After the pooling operation, we can get a vector which has k elements for each convolutional window size. Then we add the non-linear operation on that vector. We use the ReLU unit as the activation function [15].

3.3 Fully-Connected Layer and Activation Function

We can get an output vector of k dimensions for each convolutional window size for a sentence. So when we use t different convolutional window sizes to build the final sentence vector representation that is composed by concatenating t output vectors.

$$\mathbf{v} = [\mathbf{v}_{\mathbf{i}_1}^{\mathrm{T}}, \mathbf{v}_{\mathbf{i}_2}^{\mathrm{T}}, \ldots, \mathbf{v}_{\mathbf{i}_t}^{\mathrm{T}}]^{\mathrm{T}} \tag{10}$$

The vector $\mathbf{v}_{\mathbf{i}_t}$ is the output of convolution layer which convolutional window size is i_t. Following the convolutional layer, we use a fully-connected layer to transform the sentence vector representation to a scalar value which will be used to get the final predicted value.

One common activation function for output ranging from 0 to 1 is the sigmoid function. Its gradient in linear region is much greater than the gradient in non-linear region. This property is harmful to network learning when the input of activation function is out of its linear region. Because when the input is in the saturated region, the gradient will be very small which can easily result in the increment of parameters underflow when updating parameters and bring in difficulties during model learning. So we need a new activation function which can control its saturated region and alleviate the small gradient problem.

Inspired by the idea of ReLU function, we revised the activation function used for predicting emotion score as follow:

$$\mathrm{o}_{\max} = \max\left(0, \frac{x}{2 * gap} + 0.5\right) \tag{11}$$

$$\mathrm{o} = \min\left(1, \mathrm{o}_{\max}\right) \tag{12}$$

In the revised function, we can use the hyperparameter *gap* to control the linear region of the activation function. So we can make most of input of the activation function in its linear region. That will give efficient gradient to the former layer which is beneficial to learning parameters in the former layer.

4 Experiments

4.1 Experimental Setup

Dataset and Metrics

We conduct our experiments on the dataset of WASSA-2017 shared task on emotion intensity (EmoInt). The number of instances in this dataset is shown in Table 2. This dataset is composed of four emotion sub-dataset. Each sub-dataset is composed of the samples expressing same emotion.

Table 2. The number of instances in the experimental dataset

Emotion	Training set	Development set	Test set
Anger	857	84	760
Fear	1147	110	995
Joy	823	79	714
Sadness	786	74	673

We evaluate the models using the Spearman Rank Coefficient of the predicted score with the gold score of data and the Pearson Correlation Coefficient of the predicted score with the gold ratings. The correlation scores across all four emotions averaged to get the final coefficient score.

Settings

We use the pretrained word2vec [12] to initialize the word embedding matrix, and keep the word embedding fixed when the CNN model is training. Other weight parameters are initialized using the default Tensorflow initializer. We combine the training set and development set using the 5 fold cross validation on all emotion sub-datasets. And to reduce the influences of the random factor, we run every model 3 times in every cross-validation.

The result of test set is produced by the model that achieved the best result on validation set. But to compare with the result reported by Goel et al. [2], the result of test set in the Sect. 4.3 is produced by training on the dataset which is composed of merging original training dataset and validation dataset with determined number of epochs. Because of this, the result of our proposed model CNN-ATT-RA reported in Table 5 is better than the results reported in Tables 3 and 6.

Table 3. Pearson correlations of emotion intensity predictions with gold score on test dataset

Model	Average	Anger	Fear	Joy	Sadness
CNN	69.39	67.24	71.98	65.57	72.77
LSTM	47.10	41.56	45.75	50.92	50.18
SVM	64.82	63.90	65.20	65.40	64.80
CNN-ATT-RA	**70.01**	**68.06**	**72.62**	**66.32**	**73.05**

The network parameters are learned by minimizing the Mean Absolute Error between the actual and predicted values of emotion intensity. We optimize this loss

function by Adam optimization algorithm [16] with the default parameters. And the number of convolutional filters k is set 200 for all model. The convolutional window size are set as 2, 3 and 4. The hidden vector size of LSTM is 256. The hyperparameter *gap* for all emotions are set as 4 except *sadness* which is 10. All hyperparameters are selected by cross-validation.

4.2 Compared with Baselines

To illustrate the performance boost of our proposed attention model and the revised activation function, we compare our model with several strong baseline methods which includes the basic neural network architecture we use in our proposed model.

- **CNN:** Convolutional Neural Network using the pretrained word embeddings.
- **LSTM:** Using LSTM units to encode the sentence.
- **SVM:** Features includes that constructed by a lot of affection lexicon and many other common text features used by emotion classifier.
- **CNN-ATT-RA:** our proposed model.

The experimental results on all sub-dataset are shown in Tables 3 and 4.

Table 4. Spearman coefficient of emotion intensity predictions with gold score on test dataset

Model	Average	Anger	Fear	Joy	Sadness
CNN	68.38	65.61	69.89	65.61	72.40
LSTM	45.63	39.07	43.02	50.47	49.94
CNN-ATT-RA	**69.23**	**66.73**	**70.76**	**66.25**	**73.20**

In the above results, we observe that our proposed model which uses attention weight in every convolutional window and revised activation function can improve both Pearson correlation and Spearman coefficient in all emotion dataset. Specifically, our proposed method outperforms the basic CNN with a relative 0.62% in average Pearson correlation and with a relative 0.85% in average Spearman coefficient. Our proposed method outperforms the basic LSTM with a relative 22.91% in average Pearson correlation and with a relative 23.60% in average Spearman coefficient. Moreover, those results are obtained by the fine-tuned network. And all of results reported above are average of results of 5-fold cross validation and the result of each cross validation are the average of results obtained by repeatedly running models 3 time. That ensure the improvement of results are indeed benefit from the attention mechanism and the revised activation function.

The results obtained by SVM is reported in [2]. And there is no results on the Spearman coefficient in that paper.

We can observe that the results obtained by LSTM is very poor. That may be caused by lacking of training data. Therefore, the basic LSTM model cannot learn particularly effective features that can make our prediction precisely.

4.3 Compared with Results of WASSA-2017 EmoInt

To demonstrate the effective of our models, we also compare our proposed method with the reported models in WASSA-2017 EmoInt shared task.

The Pearson correlation results obtained in test dataset is shown in Table 5. Because those models doesn't report the Spearman coefficient results on their paper and results with Spearman coefficient is largely inline with those obtained using Pearson correlation, we only compare the results on the Pearson correlation.

Table 5. Pearson correlations of emotion intensity predictions with gold scores on test dataset

Approach	Average	Anger	Fear	Joy	Sadness
FFNN	69.58	67.88	72.42	68.26	69.77
Multi-DL	66.20	64.49	67.74	65.37	67.22
CNN + LSTM	**71.79**	70.15	**72.95**	69.14	74.93
Lexicons(BL)	65.00	65.00	66.00	60.00	70.00
Ext.Twitter + BL	70.00	68.00	72.00	66.00	74.00
CNN-LSTM + BL	70.00	69.00	69.00	67.00	**76.00**
Seernet	71.52	67.84	70.52	**72.81**	74.89
CNN-ATT-RA	71.76	**71.64**	72.82	69.24	73.34

In the above results, the model FFNN, Multi-DL and CNN + LSTM are the models used by the best performing system [2]. The model Lexicons, Ext.Twitter + BL and CNN-LSTM + BL are models used by the system ranked second in the shared task [3]. The model Seernet [5] are an ensembles models which ranked third in the shared task.

As the experiment settings reported on [2], we train the model on the dataset which is composed of the merger of training dataset and validation dataset when we use the model to predict the emotional intensity of samples in the test dataset. The number of epochs for each emotion is the average of the epochs which achieved best performance in the validation dataset.

Our proposed model achieves competitive performance with the best single model of the best system (CNN + LSTM) [2]. CNN + LSTM used different architecture for different emotion sub-datasets and the architecture was selected by validation dataset. Different from that, our proposed model use a unified architecture for all emotion dataset. And our proposed model also performs better than the single model (CNN-LSTM + BL, Ext.Twitter + BL) which employed many manually constructed features.

Our proposed model achieves better performance than the ensemble model Seernet [5] which ranks third in the shared task. As we can observe in the results, our proposed model achieves relatively poor results in *joy* than the result achieved by Seernet. That may be caused by predicting emotion intensity of *joy* is harder than the other emotions and the number of training data is not enough. So the ensemble methods can get better performance. The fact that all single model achieves relative poor result than the ensemble model Seernet in *joy* sub-dataset can support our view.

4.4 Ablation Experiments

To better quantify the contribution of the attention mechanism and the revised activation function, we conduct an ablation experiments in our model as follows.

- **CNN-ATT:** Convolutional neural network with our attention mechanism.
- **CNN-RA:** Convolutional neural network with our revised activation function.

The results of the ablation experiments are shown in Tables 6 and 7.

Table 6. Pearson correlation of ablation experiment on test dataset

Approach	Average	Anger	Fear	Joy	Sadness
CNN	69.39	67.24	71.98	65.57	72.77
CNN-ATT	69.69	**68.57**	72.15	65.82	72.23
CNN-RA	69.24	66.96	71.84	65.49	72.67
CNN-ATT-RA	**70.01**	68.06	**72.62**	**66.32**	**73.05**

Table 7. Spearman coefficient of ablation experiment on test dataset

Approach	Average	Anger	Fear	Joy	Sadness
CNN	68.38	65.61	69.89	65.61	72.40
CNN-ATT	68.66	**66.75**	69.95	65.94	72.01
CNN-RA	68.36	65.68	69.80	65.48	72.51
CNN-ATT-RA	**69.23**	66.73	**70.96**	**66.25**	**73.20**

As the results show, our proposed attention mechanism can achieve better performances in most of emotion dataset. And if the revised activation function is used alone, it may cause a slight decrease in performance. But if we combine the revised activation function with the attention mechanism, the revised activation function can improve the performance of the attention mechanism.

The reason why our revised activation function can improve the performance of our proposed attention mechanism is that we use a LSTM layer to encode the sentence which is used to compute the attention score. The neural network is distressed by the gradient vanish. If the input of sigmoid unit is out of its linear region, their gradients are much smaller than gradients at the linear region. Using our revised activation function in the output unit, we can control the range of linear region. So we can propagate bigger gradient to the LSTM unit, which can alleviate the gradient vanish problem in the network. This mechanism makes the LSTM unit to encode the sentence more precisely and brings in more accurate attention scores.

5 Conclusion

In this paper, we propose an attention mechanism which can improve the results for emotion intensity prediction problem. We also present a revised activation function used in the output unit which can make our attention mechanism work better through

controlling linear region of activation function. The experiment results obtained by our proposed model are better than several baselines and the ensemble system which rank third in the shared task. And our propose model can achieve competitive performance with the best performance single model in the task. Unlike the single model that selected different neural architectures for different emotion sub-datasets, our proposed model use a unified architecture for all emotion sub-datasets.

Acknowledgements. The work was supported by the National Key R&D Program of China under grant 2018YFB1004700, and National Natural Science Foundation of China (61772122, 61402091).

References

1. Mohammad, S.M., Bravo-Marquez, F.: WASSA-2017 shared task on emotion intensity. arXiv preprint arXiv:1708.03700 (2017)
2. Goel, P., Kulshreshtha, D.: Prayas at EmoInt 2017: an ensemble of deep neural architectures for emotion intensity prediction in tweets. In: Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, pp. 58–65 (2017)
3. Köper, M., Kim, E.: IMS at EmoInt-2017: emotion intensity prediction with affective norms, automatically extended resources and deep learning. In: Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis. pp. 50–57 (2017)
4. Eisner, B., Rocktäschel, T.: emoji2vec: Learning emoji representations from their description. arXiv preprint arXiv:1609.08359 (2016)
5. Duppada, V., Hiray, S.: Seernet at EmoInt-2017: tweet emotion intensity estimator. arXiv preprint arXiv:1708.06185 (2017)
6. John, V., Vechtomova, O.: UWat-Emote at EmoInt-2017: emotion intensity detection using affect clues, sentiment polarity and word embeddings. In: Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, pp. 249–254 (2017)
7. Lakomkin, E., Bothe, C.: GradAscent at EmoInt-2017: character- and word-level recurrent neural network models for tweet emotion intensity detection. arXiv preprint arXiv:1803.11509 (2018)
8. Luong, M.T., Pham, H.: Effective approaches to attention-based neural machine translation. arXiv preprint arXiv:1508.04025 (2015)
9. Mikolov, T., Chen, K.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
10. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International Conference on Machine Learning, pp. 1188–1196 (2014)
11. Kim, Y.: Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882 (2014)
12. Godin, F., Vandersmissen, B.: Multimedia Lab @ ACL W-NUT NER shared task: named entity recognition for Twitter microposts using distributed word representations. In: Proceedings of the Workshop on Noisy User-generated Text, pp. 146–153 (2015)
13. Hochreiter, S., Schmidhuber, J.: Long short-term memory. In: Neural Computation, pp. 1735–1780 (1997)

14. Wang, L., Cao, Z.: Relation classification via multi-level attention CNNs. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (2016)
15. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In Proceedings of the 27th International Conference on Machine Learning (2010)
16. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv: 1412.6980 (2014)