YNU-HPCC at EmoInt-2017: Using a CNN-LSTM Model for Sentiment Intensity Prediction

You Zhang, Hang Yuan, Jin Wang and **Xuejie Zhang**
School of Information Science and Engineering
Yunnan University
Kunming, P.R. China
xjzhang@ynu.edu.cn

Abstract

The sentiment analysis in this task aims to indicate the sentiment intensity of the four emotions (e.g. anger, fear, joy, and sadness) expressed in tweets. Compared to the polarity classification, such intensity prediction can provide more fine-grained sentiment analysis. In this paper, we present a system that uses a convolutional neural network with long short-term memory (CNN-LSTM) model to complete the task. The CNN-LSTM model has two combined parts: CNN extracts local n-gram features within tweets and LSTM composes the features to capture long-distance dependency across tweets. Our submission ranked tenth among twenty two teams by average correlation scores on prediction intensity for all four types of emotions.

1 Introduction

Advanced Social Network Services (SNSs) such as Twitter, Facebook, and Weibo provide an online platform, where people share their personal interests, activities, thoughts, and emotions. Sentiment analysis technology is used to automatically draw affective information from text. In recent researches, the majority of existing approaches and works on sentiment analysis aim to complete classification tasks. In contrast, it is often useful to know the degree of an emotion expressed in text for applications such as movies, products, public sentiments and politics.

Such attractive applications provide the motivation for the WASSA-2017 shared task on Emotion Intensity (EmoInt) (Mohammad and Bravo-Marquez, 2017), which is a competition focused on automatically determining the intensity of emotions in tweets. The task involves one-dimensional sentiment analysis, which requires a system for determining the strength (with a real-value score between 0 and 1) of an emotion expressed in a tweet. All tweets are divided into four datasets, each of which expresses an emotion including anger, fear, joy, and sadness. The tweets with higher scores correspond to a greater degree of emotion.

In the relevant research field of sentiment analysis, it has been shown that many models are available for both categorical approaches and dimensional approaches. A categorical approach focuses on sentiment classification, while a dimensional approach aims to predict the intensity of emotions. Recently, many methods have been successfully introduced for categorical sentiment analysis, such as word embedding (Liu et al., 2015), convolutional neural networks (CNN) (Kim, 2014; Jiang et al., 2016; Ouyang et al., 2015), recurrent neural networks (RNN) (Liu et al., 2015; Irsoy and Cardie, 2014), long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997; Li and Qian., 2016; Sainath et al., 2015), and bi-directional LSTM (BiLSTM) (Brueckner and Schulter, 2014). We have aimed to employ those methods for dimensional sentiment analysis, and the results show that our approach is feasible. In general, CNN can extract local n-gram features within texts but may fail to capture long-distance dependency. LSTM can address this problem by sequentially modeling texts cross messages (Wang et al., 2016).

In this paper (and for this competition), we primarily introduce a CNN-LSTM model combining CNN and LSTM. First, we construct word vectors from pre-trained word vectors using word embedding. The CNN applies convolutional and max-pooling layers, which are then used to extract n-gram features. Finally, LSTM composes those features and outputs the result. By combining CN-

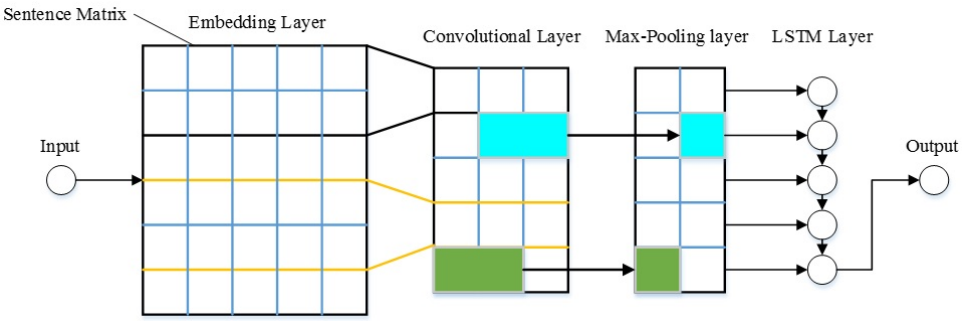

Figure 1: The architecture of CNN-LSTM model.

N and LSTM, the model can extract both local information within tweets and long-distance dependency across tweets. Our experiment reveals that the proposed model has the highest performance with data for anger and joy, while a simple CNN performs best for fear and sadness.

The remainder of this paper is organized as follows. In section 2, we described CNN, LSTM and their combination. The comparative experimental results are presented in section 3. Finally, a conclusion is drawn in section 4.

2 The CNN-LSTM model for Sentiment Intensity Prediction

The dimensional sentiment analysis in this task is intended at producing continues numerical values according to sentiment intensity. Figure 1 shows the overall framework of our model. First, a simple tokenizer is used to transform tweets into an array of tokens, which are the input of the model, and are then mapped in a feature matrix or sentence matrix by an embedding layer. Then, n-gram features are extracted when the feature matrix passes through the convolutional and max pooling layers. LSTM finally composes these useful features to output the final regression results by linear decoder.

2.1 Convolutional Neural Network

In our model, the CNN outputs are used as the inputs for the LSTM. Additionally, a simple CNN model can be produced for our task by directly using a linear regression layer as the output layer. The CNN architecture for the task is described below.

Embedding layer. The embedding layer is the first layer of the model. In this technique, words are encoded as real-valued vectors in a high dimensional space. The layer allows for the initialization of vocabulary words vectors through the pre-trained word vectors matrix. A tweet used as an input is transformed into a sequence of numerical word tokens such as $t_1, t_2, ..., t_N$, where t_N is a number representing a real word and N is the length of the token vector. To keep the size of the results identical for tweets with varying lengths, we limit the maximum value of N to the maximum length of the tweet from all tweets. Any tweet shorter than N will be padded to N using zero.

Convolutional Layer. In a convolutional layer, m filters are used to extract local n-gram features from the matrix of the previous embedding layer. In a sliding window of width w indicating a w-gram feature can be extracted, a filter $F_l (1 \leq l \leq m)$ learns the feature map y_i^l as follows:

$$y_i^l = f(T_{i;i+w-1} \circ W^l + b^l) \quad (1)$$

Where $\circ$ denotes a convolution operation, $W \in \mathbb{R}^{w \times d}$ is the weight matrix from the output of the previous layer, b is a bias, and $T_{i:i+w-1}$ denotes the token vectors $t_i, t_{i+1}, ..., t_{i+w-1} (if k > 0, t_k = 0)$. The result of filter F_l will be $y^l \in \mathbb{R}^d$, where y_i^l is the i-th element of y^l. Here we use ReLU as the activation function for fast calculation.

Max-pooling and Dropout layer. The max-pooling layer is used to down-sample and consolidate the features learned in the previous layer with a common method that takes the maximum of the input value from each filter. First, eliminating non-maximal values can reduce the computation for upper layers. Second, we choose a maximum value, because the salient feature is the most distinguishable trait of a tweet.

CNNs have a habit of overfitting, even with pooling layers. Thus, we introduce a dropout layer (Tobergte and Curtis, 2013) after both a convolution and max-pooling layer.

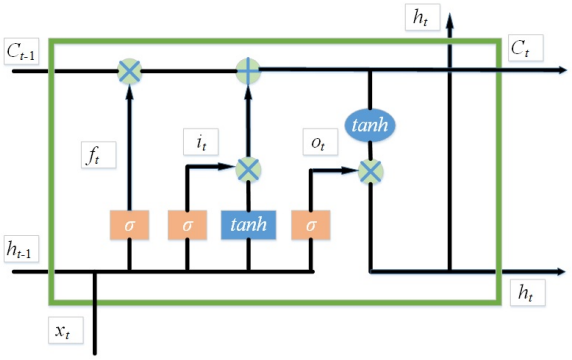

Figure 2: Architecture of LSTM cell.

2.2 Long Short-Term Memory

Recurrent Neural Networks (RNN) are a special type of neural network suitably designed for processing sequence problems. However, in a simple RNN, the gradients can produce very small numbers, which is referred to as the vanishing gradient problem (Bengio et al., 2002). The LSTM network is trained using back propagation (BP) over time and can effectively address this problem. Thus, we consider it to be the second part of our model. In addition, we could use the output of the word embedding layer as an input to the LSTM to obtain a simple LSTM model.

LSTM layer. The LSTM has memory blocks (cells) that contains outputs and gates that manage the blocks for the memory updates. In figure 2, we show how a memory block calculates hidden states h_t and outputs C_t using the following equations:

- Gate

$$\begin{aligned} f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\ i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\ o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \end{aligned} \tag{2}$$

- Transformation

$$\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_C) \tag{3}$$

- State update

$$\begin{aligned} C_t &= f_t * C_{t-1} + i_t * \tilde{C}_t \\ h_t &= o_t * \tanh(C_t) \end{aligned} \tag{4}$$

Where x_t is the input vector; C_t is the cell state vector; W and b are cell parameters; f_t, i_t, and o_t are gate vectors; and σ denotes the sigmoid function.

Output Layer. This layer outputs the final regression result, which could be a CNN or CNN-LSTM model. It is a fully connected layer using a linear decoder. A layer output vector defined as,

Content	Example	Pattern
User starts with @	@Bob	<user>
ULRs	http://ie.com	<url>
Numbers	12,345	<number>
Hashtags	#emotions	hashtag

Table 1: The example of pre-processing pattern.

$$y = h(x) = W_d x + b_d \tag{5}$$

Where x is the text token vector, y is the predicted sentiment intensity of the tweet, and W_d and b_d respectively denote the weights and bias.

The model is trained by the mean absolute error (MAE) between the predicted y and actual y. Given the training set of token matrix $X = \{x_1, x_2, ..., x_n\}$, and their actual degree of the emotion is $y = \{y_1, y_2, ..., y_n\}$, so the loss function is defined as,

$$L(X, y) = \frac{1}{2n} \sum_{i=1}^{n} \|h(x_i) - y_i\|^2 \tag{6}$$

3 Experiments and Evaluation

Data pre-processing. The organizers of the competition provided four corpora, each of which corresponds to an emotion (anger, fear, joy and sadness). The training datasets contain tweets along with a real-valued score (between 0 and 1) indicating the degree of the emotion felt by the speaker. Dev sets were provided to help us tune the parameters of the model. Here, we used the Stanford tokenizer to process tweets into an array of tokens. Since the tweets in this task primarily contain English text, all punctuations are ignored and all non-English letters are treated as unknown words. A small part of text contains emojis or emoticons, which perfectly match the conditions for emotional intensity. Therefore, these emojis or emoticons are processed into related words with similar meanings. Patterns are applied to every tweet presented in Table 1. We applied the four patterns and lowed all words to map the known pre-trained tokens. Some words that do not exist in the known tokens are treated as unknown words. In the word vectors, unknown word vectors randomly generated from a uniform distribution $U(-0.25, 0.25)$.

In this experiment, we used pre-trained word vectors including GoogleNews[1] trained by the word2vec toolkit and another one trained by GloVe[2] (Pennington et al., 2014). These programs

[1] https://code.google.com/archive/p/word2vec/
[2] https://nlp.stanford.edu/projects/glove/

Model	Metrics							
	Pearson correlation coefficient (*r*)				Spearman rank coefficient (*s*)			
	Anger	Fear	Joy	Sadness	Anger	Fear	Joy	Sadness
$\text{CNN}_{word2vec}$	0.628	0.714	0.710	0.630	0.600	0.673	0.716	0.634
$\text{CNN-LSTM}_{word2vec}$	0.591	0.591	0.657	0.551	0.586	0.555	0.662	0.566
$\text{LSTM}_{word2vec}$	0.608	0.554	0.603	0.503	0.569	0.497	0.592	0.498
$\text{BiLSTM}_{word2vec}$	0.544	0.551	0.536	0.500	0.499	0.510	0.511	0.484
CNN_{GloVe}	0.621	**0.687**	0.721	**0.630**	0.623	**0.686**	0.726	**0.639**
CNN-LSTM_{GloVe}	**0.661**	0.644	**0.797**	0.542	0.627	0.607	0.728	0.532
LSTM_{GloVe}	0.642	0.614	0.755	0.539	**0.689**	0.695	**0.772**	0.519
BiLSTM_{GloVe}	0.623	0.657	0.731	0.533	0.598	0.625	0.747	0.544

Table 3: The development data experimentation results on WASSA-2017 shard task on Emotion Intensity (EmoInt).

Parameters	Emotions			
	Anger	Fear	Joy	Sadness
m	64	32	16	32
l	3	3	2	-
n	2	2	2	-
p	0.1	0.8	0.6	0.3
c	2	2	2	-
d	300	100	300	300
b	100	50	60	100
e	30	20	50	30

Table 2: The best-tuned parameters on each dataset.

were used to initialize the weight of the embedding layer in order to build 300-dimension word vectors for all tweets. GloVe is an unsupervised learning algorithm for obtaining vector representations of words.

Implementation. This experiment used Keras with a TensorFlow backend. We use two different pre-trained word vectors and four different datasets. We introduce three other models (CNN, LSTM and BiLSTM) as baseline algorithms. Details of those three models can respectively be found in (Kim, 2014; Jiang et al., 2016; Ouyang et al., 2015), (Hochreiter and Schmidhuber, 1997; Li and Qian., 2016; Sainath et al., 2015) and (Brueckner and Schulter, 2014).

The hyper-parameters were tuned to the performance of training and dev data using the sklearn grid search function (Pedregosa et al., 2012), which can search all possible parameter combinations to evaluate models and find the best one. Different models for different data may have their own optimization parameters. For anger emotion data, the CNN-LSTMs best-tuned parameters are as follows. The number of filters (m) is 64; the length of the filter (l) is 3; the pool length (n) is 2; the dropout rate (p) is 0.1; the LSTM layer count (c) is 2, and the dimension of the LSTM hidden layer (d) is 300. The training runs with a batch size (b) of 100 and 30 epochs (e). The other three emotions shown in Table 2. The results also reveal that the models using pre-trained GloVe vectors and an Adam optimizer achieved the best performance.

Evaluation Metrics. The system is evaluated by calculating the Pearson correlation coefficient (r) and Spearman rank coefficient (s) with gold ratings. Higher r and s values indicate better performance on model prediction.

Results and Discussion. A total of twenty two teams took part in the task. Table 3 shows the detailed results of the proposed CNN-LSTM model against the three baseline models. The averaged r from the four emotions is needed to determine the bottom-line competition metric by witch the submissions will be ranked. Therefore, r is more worth considering for performance than s. The proposed CNN-LSTM model outperformed the baseline models for anger and joy data. Therefore, we chose the CNN-LSTM to create the final system to complete the subtasks of anger and joy, and ranked ninth for both r and s on anger data, eleventh for r, and thirteenth for s on joy data. In contrary, a simple CNN yielded better performance on fear and sadness data from the experimental results. Therefore, for the fear and sadness subtasks, we used a simple CNN that ranked seventh for r and eighth for s on fear data, and sixth for both r and s on sadness data.

4 Conclusion

In this paper, we described the system we submitted to WASSA-2017 Shared Task on Emotion Intensity (EmoInt). The proposed model combines CNN and LSTM to extract both local information within tweets and long-distance dependency across tweets in the regression process. Our introduced model showed good performance in the experimental results. In future work, we will attempt to introduce attention or memory mechanisms, in order to draw more useful sentiment information.

References

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. 2002. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks* 5(2):157–166. https://doi.org/10.1109/72.279181.

Raymond Brueckner and Bjorn Schulter. 2014. Social signal classification using deep blstm recurrent neural networks. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. pages 4823–4827. https://doi.org/10.1109/ICASSP.2014.6854518.

Sepp Hochreiter and Jrgex Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9(8):1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

Ozan Irsoy and Claire Cardie. 2014. Opinion mining with deep recurrent neural networks. In *Conference on Empirical Methods in Natural Language Processing*. pages 720–728.

Ming Jiang, Liqiang Jin, Feiwei Qin, Min Zhang, and Ziyang Li. 2016. Network public comments sentiment analysis based on multilayer convolutional neural network. In *2016 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*. pages 777–781. https://doi.org/10.1109/iThings-GreenCom-CPSCom-SmartData.2016.164.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *Eprint Arxiv* pages 1746–1751. https://doi.org/10.3115/v1/D14-1181.

Dan Li and Jiang Qian. 2016. Text sentiment analysis based on long short-term memory. In *2016 First IEEE International Conference on Computer Communication and the Internet (ICCCI)*. pages 471–475. https://doi.org/10.1109/CCI.2016.7778967.

Pengfei Liu, Shafiq Joty, and Helen Meng. 2015. Fine-grained opinion mining with recurrent neural networks and word embeddings. In *Conference on Empirical Methods in Natural Language Processing*. pages 1433–1443.

Saif M. Mohammad and Felipe Bravo-Marquez. 2017. WASSA-2017 shared task on emotion intensity. In *Proceedings of the Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis (WASSA)*. Copenhagen, Denmark.

Xi Ouyang, Pan Zhou, Cheng Hua Li, and Lijun Liu. 2015. Sentiment analysis using convolutional neural network. In *2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*. pages 2359–2364. https://doi.org/10.1109/CIT/IUCC/DASC/PICOM.2015.349.

Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. 2012. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research* 12(10):2825–2830. https://doi.org/10.1007/s13398-014-0173-7.2.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Conference on Empirical Methods in Natural Language Processing*. pages 1532–1543. https://doi.org/10.3115/v1/D14-1162.

Tara N. Sainath, Oriol Vinyals, Andrew Senior, and Hasim Sak. 2015. Convolutional, long short-term memory, fully connected deep neural networks. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. pages 4580–4584. https://doi.org/10.1109/ICASSP.2015.7178838.

David R. Tobergte and Shirley Curtis. 2013. Improving neural networks with dropout. *Journal of Chemical Information and Modeling* 5(13):1689–1699. https://doi.org/10.1017/CBO9781107415324.004.

Jin Wang, Liang-Chih Yu, K. Robert Lai, and Xuejie Zhang. 2016. Dimensional sentiment analysis using a regional cnn-lstm model. In *Meeting of the Association for Computational Linguistics*. pages 225–230. https://doi.org/18653/v1/P16-2037.