THU_NGN at SemEval-2019 Task 3: Dialog Emotion Classification using Attentional LSTM-CNN

Suyu Ge , Tao Qi , Chuhan Wu , Yongfeng Huang
Department of Electronic Engineering, Tsinghua University Beijing 100084, China
{gesy17,qit16,wuch15,yfhuang}@mails.tsinghua.edu.cn

Abstract

With the development of the Internet, dialog systems are widely used in online platforms to provide personalized services for their users. It is important to understand the emotions through conversations to improve the quality of dialog systems. To facilitate the researches on dialog emotion recognition, the SemEval-2019 Task 3 named EmoContext is proposed. This task aims to classify the emotions of user utterance along with two short turns of dialogues into four categories. In this paper, we propose an attentional LSTM-CNN model to participate in this shared task. We use a combination of convolutional neural networks and long-short term neural networks to capture both local and long-distance contextual information in conversations. In addition, we apply attention mechanism to recognize and attend to important words within conversations. Besides, we propose to use ensemble strategies by combing the variants of our model with different pre-trained word embeddings via weighted voting. Our model achieved 0.7542 micro-F1 score in the final test data, ranking 15^{th} out of 165 teams.

1 Introduction

The analysis of emotions in dialog systems where limited number of words appear with strong semantic relations between them deserves special attention in domain of natural language processing (NLP) due to both interesting language novelties and wide future prospects (Gupta et al., 2017). By analyzing the emotions through conversations, service providers can design better chatting strategies according to users' emotion patterns, which can improve user experience. Therefore, SemEval-2019 task 3 (Chatterjee et al., 2019) aims to call for research in this field. Given a textual dialogue, i.e., a user utterance along with two turns of context, systems need to classify the emotion of user utterance into four emotion classes: happy, sad, angry or others.

The field of sentiment analysis has been extensively studied. For example, SemEval-2018 Task 2 (Barbieri et al., 2018) once called for the study on relevance between tweet texts and emojis. However, understanding textual conversations is challenging in absence of voice modulations and facial expressions, which participants at this task are asked to deal with. Apart from diminishing the negative impact caused by class size imbalance, ambiguity, misspellings and slang, their systems should mainly focus on capturing the intricate interplay between two turns of conversations.

Traditional sentiment analysis requires a lot of feature engineering, such as n-grams and features extracted from sentiment lexicons (Mohammad and Turney, 2013; Kiritchenko et al., 2014a), and then feed them into a classifier such as Support Vector Machines (SVM) (Bollen et al., 2011; Kiritchenko et al., 2014b). However, manual feature engineering usually needs a large amount of domain knowledge. With the rapid development and ambiguity of social dialogues, these feature engineering strategies fade gradually and begin to be supplanted by neural networks (Tang et al., 2015; İrsoy and Cardie, 2014; Wang et al., 2016), which usually take word embeddings as inputs to incorporate rich semantic and syntactic information (Collobert and Weston, 2008). However, dialog emotion analysis is still very challenging, since dialog conversations can be very noisy and informal. In addition, the emotions evoked by conversations are usually highly context-dependent.

In this work, we propose an end-to-end attentional LSTM-CNN network as a unified model without hand-crafted features. In our approach, we use a combination of LSTM and CNN to capture both local and long-distance information. We use attention mechanism to select important words to learn more informative word representations. In addition, we use a data balancing method by setting a cost-sensitive loss function for training.

Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019), pages 340–344
Minneapolis, Minnesota, USA, June 6–7, 2019.

Besides, we use ensemble strategies by using a combination of the variants of our model with different pre-trained word embeddings. Our model achieved 0.7542 micro-F1 score on the test set, and extensive experiments validate the effectiveness of our approach. The source code can be found in our repository on github.[1]

2 Our Approach

The framework of our attentional LSTM-CNN model is illustrated in Figure 1. Each layer of network is introduced from bottom to top in the following sections.

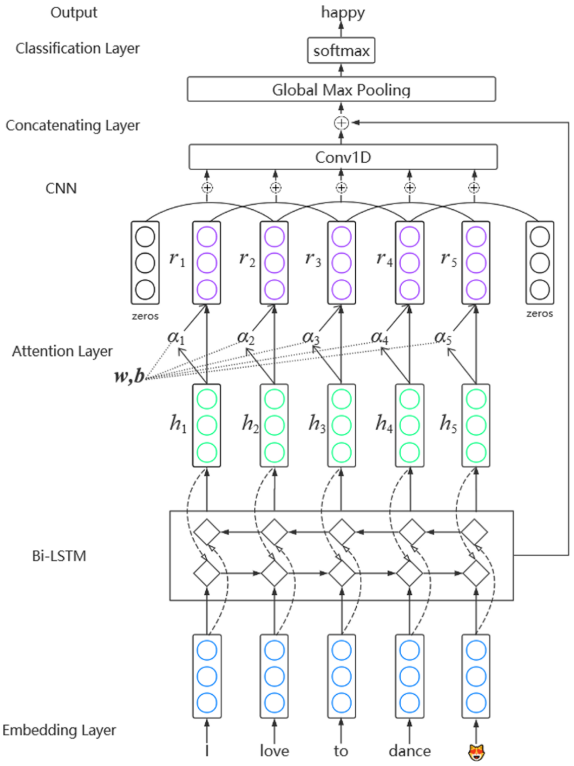

Figure 1: The architecture of our attentional LSTM-CNN model, the output is generated by soft voting ensemble after the softmax layer.

2.1 Word Embeddings

The first layer is a word embedding layer, which aims to convert the sequence of words in conversations into a low-dimensional vector sequence. We harness three types of pre-trained word embeddings, i.e., word2vec-twitter (Godin et al., 2015), pre-trained ekphrasis (Baziotis et al., 2017) vectors and GloVe (Pennington et al., 2014), to initialize the word embedding matrix.

[1] github.com/gesy17/Semeval2019-Task3-Emocontext

2.2 Bi-LSTM Layer

Considering the close relevance between two turns of dialogues, we use Bi-LSTM as encoder to capture abstract information from both directions. It consists of a forward LSTM $\overrightarrow{f}$ that encodes the sentence from $\boldsymbol{x_1}$ to $\boldsymbol{x_t}$ and a backward LSTM $\overleftarrow{f}$ that encodes the sentence backward. we concatenate the hidden representations in both directions, we get final representation of a word $\boldsymbol{x_i}$:

$$\boldsymbol{x_i} = \overleftarrow{\boldsymbol{x_i}} || \overrightarrow{\boldsymbol{x_i}} \quad \boldsymbol{x_i} \in \boldsymbol{R}^{2d}, \tag{1}$$

where $||$ denotes the concatenation operation and d is the size of each LSTM.

2.3 Attention Mechanism

An attention layer is incorporated after the Bi-LSTM layer to automatically select and attend to important words. The input of the attention layer is the hidden state vector $\boldsymbol{h_i}$ at each time step. The attention weight α_i for this time step can be computed as:

$$\boldsymbol{m_i} = \tanh \boldsymbol{h_i}, \tag{2}$$

$$\alpha_i = \boldsymbol{w}^T \boldsymbol{m_i} + b, \tag{3}$$

$$\alpha_i = \frac{\exp(\alpha_i)}{\sum_j exp(\alpha_j)}, \tag{4}$$

where w and b are the parameters of the attention layer. The output of attention layer at the i^{th} time step is:

$$\boldsymbol{r_i} = \alpha_i \boldsymbol{h_i} \tag{5}$$

2.4 CNN Layer

We use a convolutional neural network (CNN) to capture local contexts. Inspired by the residual connection for in ResNet (He et al., 2016), which combines the CNN outputs with original inputs to get better accuracy and shorter training time of deep CNN, we apply a merge layer to combine Bi-LSTM outputs and CNN outputs together. Our experiment proves that this structure can achieve a higher accuracy due to its full usage of both chronological information and local contextual information. Finally, max pooling is applied to the concatenated vectors to build conversation representations.

2.5 Emotion Classification

To make the final emotion prediction, we use a dense layer with softmax activation function to classify emotions. Considering the unbalanced data in both training set and testing set, we

	Happy	Sad	Angry	Average
Precision	0.7452	0.8117	0.7329	0.7598
Recall	0.6796	0.7760	0.7919	0.7488
F1	0.7109	0.7935	0.7613	0.7542

Table 1: Evaluation result on our final submission.

choose a cost-sensitive cross entropy loss function (Santos-Rodrguez et al., 2009) to modify the attention our model gives to different emotion categories. The loss function we use is formulated as:

$$\mathcal{L} = -\sum_{i=1}^{N} w_{y_i} y_i log(\hat{y_i}), \quad (6)$$

where N is the number of dialogue sentences, y_i is the emotion label of the i^{th} dialogue, $\hat{y_i}$ is the prediction score, and w_{y_i} is the loss weight of emotion label y_i. w_{y_i} is defined as $\frac{\sum_{k=1}^{C}\sqrt{N_k}}{\sqrt{N_{y_i}}}$, where C is the number of emotion categories and N_j is the number of texts with emotion label j. Consequently, this helps our model place higher weights towards infrequent emotion categories.

The last layer of our network utilizes a weighted soft voting ensemble method to fully take the advantage of different word embeddings. It should be mentioned that we design exactly the same network architecture with only word embeddings as slight differences. This soft voting method strengthens robustness and modifies our model to predict the class with the highest class probability.

3 Results and Analysis

3.1 Experimental settings

In our experiments, the word2vec-twitter embedding (Godin et al., 2015) was trained on 400 million microposts, which has a vocabulary of 3,039,345 words and 400-dimensional word representations. The Ekphrasis model leverages a collection of 330 million Twitter messages to generate word embeddings. It also uses GloVe as pretrained word vectors. Besides, a pre-processing pipeline is developed to enable users to get word vectors in a directly numerical form[2]. We also incorporate the GloVe embedding model and select the cased 300-dimension version[3] obtained by training on 2.2M data crawling from the Internet, containing 840B tokens in total.

[2]github.com/cbaziotis/ekphrasis
[3]nlp.stanford.edu/projects/glove

With word2vec-twitter embedding and GloVe embedding, we send raw texts to NLTK TweetTokenizer and randomly generate word vectors for all emojis and those out of vocabulary words appearing more than 3 times. Moreover, as to ekphrasis embedding, we use the pipeline provided by it. The pre-processing steps included in it are: Twitter-specic tokenization, spell correction, word normalization, word segmentation (for splitting hashtags) and word annotation.

In the experiment, we pertain the original dimension of the word embeddings and send them to a 400 dimension Bi-LSTM, adding to totally 800 dimension in LSTM layer. In the next CNN layer, the number of filters is 256, with filter length of 3. After each layer, we employ dropout with a drop rate of 0.2 to mitigate overfitting. Additionally, rmsprop (Tieleman and Hinton, 2012) is chosen as optimizer and Keras library (Chollet et al., 2015) is used for implementation.

3.2 Performance Evaluation

The final submission which scores micro $F1$ 75.42 is equipped with both the attention mechanism and weighted soft voting ensemble. The final result is shown in table 1, it suggests that our model performs relatively lower on happy emotion due to lack of training data and ambiguity. We evaluate parts of our network in the following paragraphs. The baseline we use is LSTM-CNN architecture(**LSTM-CNN**), baseline with concatenating layer is denoted as **LSTM-CNN+CL**. Upon this, attention mechanism is added, which is written as **LSTM-CNN+CL+AT**. Finally, a weighted soft voting is introduced, namely **LSTM-CNN+CL+AT+WE**. The result comparison is shown in table 2.

Concatenating Layer. By combining the outputs of Bi-LSTM and CNN layer, the model learns both local feature and long-term context, with the most obvious improvement in Word2vec-twitter, F1 score increasing from 0.7307 to 0.7483.

Attention Mechanism. Adding attention into network helps our network select those more essential words in the case of Ekphrasis and Glove word embeddings, but Word2vec-twitter witnesses a slight decline. This may be due to the randomness of out-of-vocabulary words and emoji word vectors. Overall speaking, attention benefits the study of word importance to some degree.

Weighted Soft Voting Ensemble. We place

	Word2vec-twitter	Ekphrasis	GloVe
LSTM-CNN	0.7307	0.7313	0.7429
LSTM-CNN+CL	**0.7483**	0.7355	0.7450
LSTM-CNN+CL+AT	0.7388	**0.7392**	**0.7460**
LSTM-CNN+CL+AT+WE		**0.7542**	

Table 2: Results on test data under various system framework.

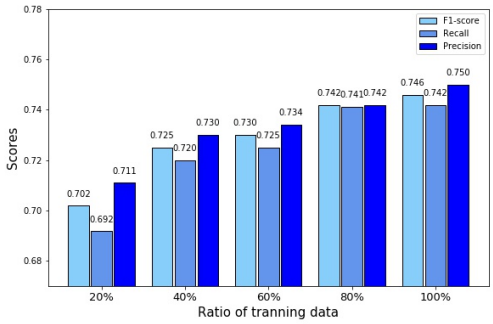

Figure 2: Influence of reducing training data on evaluation scores.

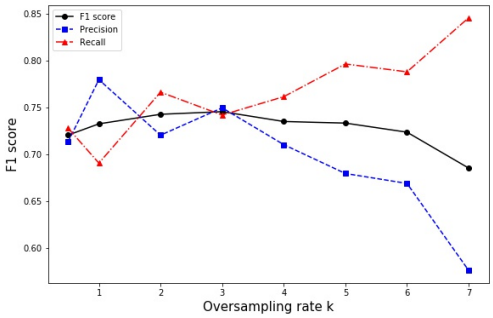

Figure 3: Influence of oversampling rate k on evaluation scores.

the highest weight on those showing good performances on dev dataset in our final submission. The significant improvement of F1 score at the bottom of table 2 indicates the power of ensemble. Results show that GloVe performs better than other two word embeddings, thus given more weight in practice. The result is ensembled from eight predictions , with the quantity we use of Word2vec-twitter, Ekphrasis, GloVe is respectively 2:3:3.

Quantity of training data. Since many methods in sentiment analysis rely heavily on high quality labeled data, we test our model with different reduction portion rate of training data. It can be seen in figure 2 that although there exists certain degree of performance reduction when the data amount is limited, our approach remain a F1 score of approximately 0.70 even with only 20% data, which proves that our approach can be widely applied to even when there exists shortage of labeled data.

Oversampling rate. The oversampling rate is defined to be the rate of loss weight between the class "others" and other three emotion categories. We officially set the oversampling rate k to be 3, meaning the loss weight rate between "others" and other three emotion categories is 3:1:1:1. To test the effectiveness of our choice of k, we select k to be in range from 0.5 to 7 and report the changes on F1 score, precision and recall in figure 3. It should be noticed that the scores are extremely unstable when $k < 2$, which may due to the sparsity of emotion labels in training data.

4 Conclusion

In this paper, we propose an attentional LSTM-CNN based neural network with concatenating layer for SemEval-2019 Task 3, i.e., predicting emotion categories of online dialogues. To strengthen robustness, weighted soft voting ensemble is exploited.

Acknowledgments

This work was supported by the National Key Research and Development Program of China under Grant number 2018YFC1604002, and the National Natural Science Foundation of China under Grant numbers U1836204, U1705261, U1636113, U1536201, and U1536207.

References

Francesco Barbieri, Jose Camacho-Collados, Francesco Ronzano, Luis Espinosa Anke, Miguel Ballesteros, Valerio Basile, Viviana Patti, and Horacio Saggion. 2018. Semeval 2018 task 2: Multilingual emoji prediction. In *Proceedings of*

The 12th International Workshop on Semantic Evaluation, pages 24–33. Association for Computational Linguistics.

Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. Datastories at semeval-2017 task 4: Deep lstm with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 747–754, Vancouver, Canada. Association for Computational Linguistics.

Johan Bollen, Huina Mao, and Alberto Pepe. 2011. Modeling public mood and emotion: Twitter sentiment and socio-economic phenomena. *Icwsm*, 11:450–453.

Ankush Chatterjee, Kedhar Nath Narahari, Meghana Joshi, and Puneet Agrawal. 2019. Semeval-2019 task 3: Emocontext: Contextual emotion detection in text. In *Proceedings of The 13th International Workshop on Semantic Evaluation (SemEval-2019)*, Minneapolis, Minnesota.

Franois Chollet et al. 2015. Keras. `https://github.com/fchollet/keras`.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 160–167, New York, NY, USA. ACM.

Frderic Godin, Baptist Vandersmissen, Wesley De Neve, and Rik Van De Walle. 2015. Multimedia lab @ acl w-nut ner shared task: Named entity recognition for twitter microposts using distributed word representations. In *Workshop on User-generated Text*.

Umang Gupta, Ankush Chatterjee, Radhakrishnan Srikanth, and Puneet Agrawal. 2017. A sentiment-and-semantics-based approach for emotion detection in textual conversations. *arXiv preprint arXiv:1707.06996*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

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

Svetlana Kiritchenko, Xiaodan Zhu, Colin Cherry, and Saif Mohammad. 2014a. Nrc-canada-2014: Detecting aspects and sentiment in customer reviews. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 437–442. Association for Computational Linguistics.

Svetlana Kiritchenko, Xiaodan Zhu, and Saif M Mohammad. 2014b. Sentiment analysis of short informal texts. *Journal of Artificial Intelligence Research*, 50:723–762.

Saif M. Mohammad and Peter D. Turney. 2013. Crowdsourcing a wordemotion association lexicon. *Computational Intelligence*, 29(3):436–465.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543.

Ral Santos-Rodrguez, Dario Garca-Garca, and Jess Cid-Sueiro. 2009. Cost-sensitive classification based on bregman divergences for medical diagnosis.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2015. Target-dependent sentiment classification with long short term memory. *CoRR, abs/1512.01100*.

Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31.

Yequan Wang, Minlie Huang, xiaoyan zhu, and Li Zhao. 2016. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 606–615. Association for Computational Linguistics.