THU-HCSI at SemEval-2019 Task 3: Hierarchical Ensemble Classification of Contextual Emotion in Conversation

Xihao Liang, Ye Ma, Mingxing Xu
Department of Computer Science & Technology
Tsinghua University, Beijing, China
(liangxh16, ma-y17)@mails.tsinghua.edu.cn, xumx@tsinghua.edu.cn

Abstract

In this paper, we describe our hierarchical ensemble system designed for the SemEval-2019 task3, *EmoContext*. In our system, we trained three sets of classifiers for four-emotion classification, *Happy-Sad-Angry* classification and *Others*-or-not classification respectively. Through three steps of voting, the predicted labels of these three sets of classifiers are combined to make the final prediction. Experiment results show that the ensemble approach manages to obtain better predictive performance in comparison with the base classifiers and our system has achieved the top 10 performance in the final evaluation of *EmoContext*.

1 Introduction

Sentiment analysis is a task to identify the emotion conveyed by written language. With the popularization of the Internet and instant message applications, text has become one of the most familiar media by which people express their ideas and communicate with each other. Automatic emotion classification can help people and robots better understand the messages or comments written by other people and make proper responses, which makes this study field increasingly important.

Approaches to sentiment analysis can be classified into three categories: rule-based approaches, non-neural machine learning approaches, and deep learning approaches. Significantly, neural networks have achieved state-of-art performance in many recent studies. Gupta et al. (2017) used a LSTM-based model to identify the emotion in 3-turn conversations on Twitter but the manipulation of the turn information was not mentioned. For emotion detection on TV show transcripts, a sequence-based convolutional neural network which can make use of the information in the previous lines was designed by Zahiri and Choi (2017). Hazarika et al. (2018) proposed a conversational memory network based on both CNN (Lecun and Bengio, 1998) and gated recurrent unit (Chung et al., 2014) to recognize emotion in dyadic dialogue videos, featuring its ability to manipulate the information of different speakers. Designing a neural network of the proper architecture to capture the information conveyed by different types of text data still remains a valuable topic.

In this paper, we describe our approach to SemEval-2019 task3, *EmoContext* (Chatterjee et al., 2019), which aims to encourage more research of contextual emotion detection in textual conversation. Datasets of 3-turn conversations are provided and the participating systems are required to predict the contextual emotion of the last turn in the conversation: *Happy*, *Sad*, *Angry* or *Others*. The system performance is evaluated by the micro-averaged F1-score for *Happy*, *Sad* and *Angry* (hereinafter referred to as *HSA*) on the given Test set. Our system is composed of three sets of CNN-based classifiers trained for four-emotion classification, *Happy-Sad-Angry* classification and *Others*-or-not classification respectively. Through three steps of voting, the predicted labels of the base classifiers are combined to make the final prediction. Our system has achieved F1-score of 0.7616 in the final evaluation on the Test set, which is the top 10 performance out of 165 participating systems.

This paper is organized as follows. In Section 2, we describe the datasets provided by the task organizers. In Section 3, our system and the strategies of training base classifiers are detailed. In Section 4, we present and discuss the evaluation results of our system and the base classifiers. Conclusion is given in Section 5.

Turn 1	Turn 2	Turn 3	label
I live in uttra khand degreee	ohh nice! love that place! ∧.∧ what degree & where?	☺☺ sryyy i really got to goo	happy others

Table 1: Example training samples in Train set.

Dataset	Others	Happy	Sad	Angry
Train	14948	4243	5463	5506
Dev	2338	142	125	150
Test	4677	284	250	298

Table 2: Class distribution in each dataset.

Parameter	Value
# of classifiers in each set	5
thr_{II}	2
thr_{III}	3
std. of Gaussian noise	0.1
kernel size of CNN	5
filter number of CNN	128
# of cells of the first dense layer	32
dropout keep prob.	0.5
l_2	0.2
initial learning rate	0.005
decay of learning rate	0.9
minibatch size	100

Table 3: Configuration of our system.

2 Data

Task organizers have released three datasets. Each sample in these datasets contains a 3-turn conversation(Table 1), in which Turn 1 was written by User 1, Turn 2 is User 2's reply to Turn 1 and Turn 3 is User 1's reply to Turn 2. The emotion label of Turn 3 is manually annotated for each conversation, which is one of the four emotions: *Happy*, *Sad*, *Angry* and *Others*. Class distributions of these three datasets are shown in Table 2. As the Train set and Dev set are both allowed to be used for model training at the phase of final evaluation, these two datasets are combined (hereinafter referred to as the training set) to develop our system.

3 System Description

3.1 Preprocessing

Through our observation of the 3-turn conversations in the datasets, we realize the writing style resembles that of the tweets and comments in Twitter, featuring emoticons, informal usage of language, spelling errors and so on. Hence, we utilize the tweet processor, *ekphrasis*[1] (Baziotis et al., 2017). The preprocessing steps include (1) Twitter-specific tokenization, (2) spell correction, (3) word normalization for numbers and dates, (4) annotation for all-capital words, elongated words and repeated punctuations, (5) conversion of emoticons to emotion labels.

3.2 Word Embeddings

Word embedding is to capture semantic, syntactic and sentiment information of a word or token and represent it by a vector. We use the pretrained embedding model provided by Baziotis et al. (2018) [2] (hereinafter called NtuaW2V) in our system, which contains 300-dimensional vectors trained on Twitter messages that are also preprocessed by *ekphrasis*. In addition, we tried another pretrained model provided by Mikolov et al. (2013) [3] (hereinafter called GoogleW2V), which contains 300-dimensional vectors trained on Google News dataset, in order to get an insight into the effect of different embedding models.

3.3 Classifier

Our system is composed of neural network classifiers sharing the same architecture (Fig 1). The input to the classifier is a 3-turn conversation, treated as 3 sequences of tokens after the preprocessing step. An embedding layer is used to project the input to 3 sequences of vectors. This layer is initialized by the pretrained word embeddings mentioned in Section 3.2. For tokens that are not covered in the embedding model but occur in no less than two training samples, we randomly initialize their embedding vectors.

Each sequence of vectors is then fed into a CNN layer followed by max pooling respectively to get a vector as the representation of the corresponding turn. Three vectors are concatenated as the feature vector of the 3-turn conversation and it is then fed into a dense layer with a rectified linear unit (ReLU) and another dense layer with softmax to get the probability distribution over all predicted classes.

[1] https://github.com/cbaziotis/ekphrasis

[2] https://github.com/cbaziotis/ntua-slp-semeval2018

[3] https://code.google.com/archive/p/word2vec/

3.4 Hierarchical Ensemble

In order the improve the predictive performance, we design three steps of voting to combine the predicted labels of base classifiers. Three sets of classifiers are trained. Set A is a set of four-emotion classifiers trained on the given dataset and the original labels. Set B is a set of triple classifiers trained only on the samples of *HSA*. Classifiers in Set C are binary classifiers, which are also trained on the given dataset but the labels for samples of *HSA* are changed to *Not Others*.

The first step is the majority voting of Set A to get the base predicted labels (hereinafter referred to as Prediction I). At the second step, majority voting is applied to Set B and we change the predicted label to the new major vote if the sample is not predicted as *Others* in Prediction I and the voting count of the majorly voted class exceeds a given threshold thr_{II}, by which the prediction for *HSA* is refined. This set of predicted labels are hereinafter referred to as Prediction II. At the third step, classifiers in Set C vote for each sample whether it belongs to *Others* or not. Then we change the predicted label to *Others* if more than thr_{III} classifiers in Set C vote for *Others*. This set of predicted labels are used as the final prediction (hereinafter referred to as Prediction III). We train a set of classifiers for voting at each step instead of using single classifier in order to make the revision of the predicted labels statistically more reliable.

3.5 Regularization

To alleviate the problem of overfitting, the following strategies are applied.

- Early stopping is used. For each classifier, we randomly select 90% of the training set for training and the 10% left for validation so that the information captured by each base classifier will diverse. Although the evaluation metric is micro-averaged F1 score for *HSA*, we choose the set of neural network weights that achieves the best micro-averaged precision on the validation set when we train the classifiers in Set A. As we notice that the precision is significantly sensitive to the class distribution and the task organizers have emphasized the difference between the class distributions of the provided datasets in advance, we need the precision of the base classifiers to be as high as possible.

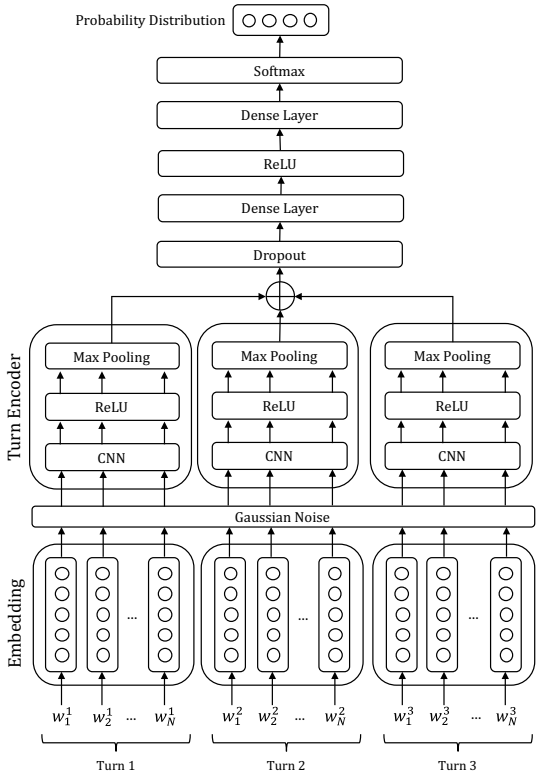

Figure 1: The architecture of the CNN-based classifiers in our system.

- New samples of *Others* are automatically created. For each sample of the class *Others* in the selected 90% training set, we create a new *Others* sample by randomly remove one token in one of the turns to simulate the situation when the user misspells a word and that misspelled token is not known to our embedding models. This is inspired by our observation that most of the *Others* samples in the training set still belong to *Others* even if one of the tokens is missing or misspelled. However, we do not generate samples for *HSA* automatically in case a discriminative token is removed and a missleading sample will be made.

- Embedding layer is not fine-tuned as the embedding space of the tokens in the training set will be changed but the embedding vectors of the tokens in the Test set but not in the training set are not adjusted, which will lead to wrong representations of these tokens in the new embedding space.

- Gaussian noise is added after the embedding layer, by which the model is more robust

Classifier	Acc	Prec	Rec	F1
Base	**0.9244**	0.7247	0.7661	**0.7445**
no extra Others	0.9206	0.6974	0.7932	0.7420
Early stopping				
F1-score	0.9112	0.6521	0.8093	0.7222
Recall	0.9020	0.6190	**0.8253**	0.7073
Emb				
GoogleW2V	0.9226	**0.7421**	0.7163	0.7286

Table 4: Performance of the base classifier and its variants on the Test set.

when dealing with tokens of similar meaning, as they are supposed to be projected to adjacent embedding vectors.

- Dropout (Srivastava et al., 2014) layer is added before the stacked dense layers (Fig 1).
- L2 regularization is applied to the weights of the stacked dense layers.

4 Experiments and Discussion

4.1 Implementation Details

Tensorflow (Abadi et al., 2015) is used to develop our models. For network optimization, we choose Adam algorithm (Kingma and Ba, 2014). The configuration of our system and the hyperparameters of the base classifiers are shown in Table 3.

4.2 Results and Discussion

Table 4 shows the performance of our base classifier and its variants on the Test set. Note that **Base** refers to the model trained as mentioned in Section 3 and **Prec**, **Rec** and **F1** refers to the micro-averaged ones for *HSA*.

According to Table 4, adding automatically generated *Others* samples can improve the accuracy, precision and F1-score as more *Others* samples are correctly predicted but less *HSA* samples are recalled as the cost.

We also observe the performance difference when different metric is used as the indicator for choosing the network weights in early stopping, which is rarely discussed but the results show that the effect is significant. It is because of the remarkable difference between the class distributions of the training set and the Test set, which is emphasized by the task organizers.

On the other hand, NtuaW2V and GoogleW2V, the embedding models used to initialize the embedding layer, are compared. Results show that using NtuaW2V can achieve better F1-score while using GoogleW2V achieve the best precision but also the worst recall among the variants, which shows the importance of choosing suitable pretrained embedding models. The data on which the embedding model is trained and how the data are preprocessed should be the crucial keys to it.

Prediction	Acc	Prec	Rec	F1
(Base)	0.9244	0.7247	0.7661	0.7445
I	0.9278	0.7360	0.7740	0.7545
II	0.9281	0.7383	**0.7764**	0.7569
III	**0.9305**	**0.7553**	0.7680	**0.7616**

Table 5: Performance of our system after each step of voting on the Test set.

Table 5 illustrates the performance of our system after each step of voting. Results show that the first two steps bring slight improvement on all four metrics and the final step improves F1-score by raising the precision at the cost of recall. The improvement also implies that, in comparison with Set A, classifiers in Set B are more effective to distinguish *HSA* samples and those in Set C are more precise to classify whether a sample belongs to *Others*. Although these classifier sets are all trained on the given dataset, Set B and Set C manage to work as a patch to Set A as they only focus on the simplified classification problems.

5 Conclusion and Future Work

In this paper, we present our system used for SemEval2019 Task3, *EmoContext*. This system is composed of three sets of CNN-based neural network models trained for four-emotion classification, *Happy-Sad-Angry* classification and *Others*-or-not classification respectively. Three steps of voting are used to combine the predicted labels of the base classifiers and make the final prediction. Experiment results show that our training strategies are effective and the ensemble approach manages to improve the performance in comparison with the base classifiers. We highlight the importance of choosing pretrained embedding models and picking the right metric as the indicator for choosing network weights in early stopping. In order to achieve a better result based on our system, improving the performance of the base classifier is crucial. The architecture of neural networks and the features used as the input should be the fields that worth further exploration.

References

Martin Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, G.s Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, and Xiaoqiang Zheng. 2015. Tensorflow : Large-scale machine learning on heterogeneous distributed systems.

Christos Baziotis, Nikos Athanasiou, Pinelopi Papalampidi, Athanasia Kolovou, Georgios Paraskevopoulos, Nikolaos Ellinas, and Alexandros Potamianos. 2018. Ntua-slp at semeval-2018 task 3: Tracking ironic tweets using ensembles of word and character level attentive rnns.

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. pages 747–754.

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.

Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555.

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

Devamanyu Hazarika, Soujanya Poria, Amir Zadeh, Erik Cambria, Louis-Philippe Morency, and Roger Zimmermann. 2018. Conversational memory network for emotion recognition in dyadic dialogue videos. pages 2122–2132.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Yann Lecun and Yoshua Bengio. 1998. *Convolutional networks for images, speech, and time series*.

Tomas Mikolov, Ilya Sutskever, Chen Kai, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, 26:3111–3119.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Sayyed M. Zahiri and Jinho D. Choi. 2017. Emotion detection on tv show transcripts with sequence-based convolutional neural networks.