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, three sets of classifiers are trained for different sub-targets and the predicted labels of these base classifiers are combined through three steps of voting to make the final prediction. Effective details for developing base classifiers are highlighted. 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 performance within the top 10 ranks in the final evaluation of *EmoContext*.

1 Introduction

Sentiment analysis is a task to identify the emotion conveyed by written language (Balahur et al., 2011; Lee et al., 2018). 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 the others and make proper responses, which makes this study field increasingly important (Sun et al., 2018).

Deep learning approaches have achieved state-of-art performance in many recent studies of sentiment analysis (Rodrigues do Carmo et al., 2017). Gupta et al. (2017) used an LSTM-based model to identify the emotion in 3-turn conversations on Twitter. For emotion detection on TV show transcripts, a sequence-based convolutional neural network which can associate 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. These studies focused on the architecture of the neural networks, but the optimization of the classification system based on the feature of the datasets are rarely discussed, which is crucial when dealing with specific real-world problems.

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 *Angry*, *Happy*, and *Sad* (hereinafter referred to as *AHS*) on the given Test set. According to our observation of the dataset and single classifier's performance on it, we design a hierarchical ensemble classification system, which is composed of three sets of base classifiers, and three steps of voting to combine their predicted labels to make the final prediction. Our system has achieved F1-score of 0.7616 in the final evaluation, which is within the top 10 performances out of 165 participating systems.

This paper is organized as follows. In Section 2, our system and the strategies of training base classifiers are demonstrated. In Section 3, experiments are detailed and the evaluation results of our system and the base classifiers are presented and discussed. Conclusion is given in Section 4.

2 System Description

As we notice that the distinction among *AHS* and that between *Others* and *AHS* are significant, our system is designed to contain three sets of classifiers trained for different sub-targets and the pre-

Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019), pages 345–349
Minneapolis, Minnesota, USA, June 6–7, 2019.

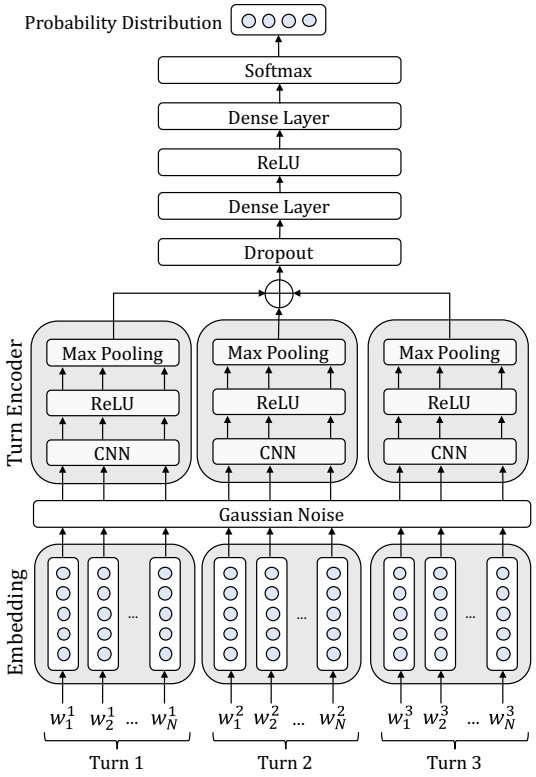

Figure 1: The architecture of the CNN-based classifiers in our system.

dicted emotion of a 3-turn conversation is obtained and refined through three steps of voting.

2.1 Classifier

Every base classifier in our system shares the same architecture (Fig 1). The input to the classifier is a 3-turn conversation represented as three sequences of tokens. An embedding layer is used to map the input to three sequences of vectors. Each turn is encoded to a vector individually by feeding its sequence of vectors into a CNN layer followed by a rectified linear unit (ReLU) and max pooling, as we notice that each turn's contribution to the prediction is different and the predictive clue to the contextual emotion can be captured within the turn. Three vectors are then obtained and concatenated as the feature vector of the 3-turn conversation. It is finally fed into a dense layer with a ReLU and another dense layer with softmax to get the probability distribution over all predicted classes.

2.2 Hierarchical Ensemble

Three steps of voting are designed (Algorithm 1) and a set of classifiers are trained for each step. For the first step, a set of four-emotion classifiers

Algorithm 1 Hierarchical ensemble

Input: the majority vote of Set A, $Label^A_{MV}$; the majority vote of Set B, $Label^B_{MV}$, and its voting count, $Count^B_{MV}$; the voting count for *Others* in Set C, $Count^C_{Others}$; two chosen thresholds, thr_{II} and thr_{III}
Output: the predicted labels after each step of voting, $Label_I$, $Label_{II}$, $Label_{III}$;

$Label_I \leftarrow Label^A_{MV}$;
if $Label_I \neq Others$ & $Count^B_{MV} \geq thr_{II}$ **then**
  $Label_{II} \leftarrow Label^A_{MV}$;
else
  $Label_{II} \leftarrow Label_I$;
end if
if $Count^C_{Others} >= thr_{III}$ **then**
  $Label_{III} \leftarrow Others$;
else
  $Label_{III} \leftarrow Label_{II}$;
end if

(hereinafter referred to as Set A) are trained on the given dataset and the original labels. For each test case, majority voting is applied to Set A to get the base predicted label, $Label^A_{MV}$. This set of predicted labels are hereinafter referred to as Prediction I.

For the second step, a set of three-emotion classifiers (hereinafter referred to as Set B) are trained only on the data of *AHS*. For each test case, majority voting is applied to Set B to get a new label $Label^B_{MV}$ and the predicted label for this case is changed to $Label^B_{MV}$ if $Label^A_{MV}$ is not *Others* and more than thr_{II} classifiers in Set B voted for $Label^B_{MV}$, by which the prediction for *AHS* is refined. This set of predicted labels are hereinafter referred to as Prediction II.

For the final step, a set of binary classifiers (hereinafter referred to as Set C) are trained on the given dataset but the labels for *AHS* are changed to *Not Others*. For each test case, we change its predicted label to *Others* if more than thr_{III} classifiers in Set C vote for *Others*, by which more *Others* samples are recalled. This set of predicted labels (hereinafter referred to as Prediction III) are used as the final prediction of our system.

2.3 Regularization

Common methods to alleviate the problem of overfitting are applied. Embedding layer is not fine-tuned. If this layer is tuned through training, the embedding space will be changed but the vectors of the tokens that exist only in the Test set are not adjusted, which may lead to wrong representations of these tokens. Gaussian noise is added after the embedding layer, by which the model is more robust when dealing with tokens of similar mean-

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.

ing. L2 regularization is applied to the weights of the stacked dense layers. Dropout (Srivastava et al., 2014) layer is added. Two more strategies are specifically highlighted.

- For each classifier, 90% of the training data is randomly selected for training and the 10% left for validation so that different information is captured. Although the evaluation metric is micro-averaged F1-score for *AHS*, for classifiers in Set A, the set of neural network weights that achieves the best micro-averaged precision on the validation set through the training process is chosen. 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.
- For each sample of the class *Others* in the training data, a new *Others* sample is created by randomly removing 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 data still belong to *Others* even if one of the tokens is missing or misspelled. However, samples for *AHS* are not automatically generated in case a discriminative token is removed and a misleading sample will be made.

3 Experiments and Discussion

3.1 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 annotated for each conversation, which is one of the four emotions: *Angry*, *Happy*, *Sad*, and *Others*. Class distributions of these three datasets are shown in Table 2.

3.2 Preprocessing

We notice that the writing style of the conversations in the datasets resembles that of the tweets and comments on 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, through which each 3-turn conversation is converted to three sequences of tokens.

3.3 Word Embeddings

We deploy 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. The pretrained embeddings are used to initialize the embedding layer. For tokens that are not covered in the embedding model but occur in no less than two training samples, their embedding vectors are randomly initialized.

3.4 Implementation Details

Tensorflow (Abadi et al., 2016) is used to develop our models. For network optimization, we choose Adam algorithm (Kingma and Ba, 2014).

[1] https://github.com/cbaziotis/ekphrasis
[2] https://github.com/cbaziotis/ntua-slp-semeval2018
[3] https://code.google.com/archive/p/word2vec/

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.

The configuration of our system and the hyper-parameters of the classifiers are shown in Table 3.

3.5 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 2 and **Prec**, **Rec**, and **F1** refer to the micro-averaged ones for *AHS*.

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

We also observe the performance difference when a different metric is used as the indicator for choosing the network weights, 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 data and the Test set, which is emphasized by the task organizers.

On the other hand, the embedding models used to initialize the embedding layer are compared. Results show that using NtuaW2V achieves better F1-score while using GoogleW2V achieves 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 keys to it.

Table 5 and 6 illustrate 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 the recall. The improvement also implies that, in comparison with Set A, classifiers in Set B are more effective to distinguish *AHS* 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.

Classifier	Acc	Prec	Rec	F1
BASE	**0.9244**	0.7247	0.7661	**0.7445**
w/o extra Others	0.9206	0.6974	0.7932	0.7420
choose weight by				
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.

Prediction	Acc	Prec	Rec	F1
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.

4 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, *Angry-Happy-Sad* 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 the ensemble approach manages to improve the performance in comparison with the base classifiers. Automatic generation of random *Others* samples is proven effective and the importance of choosing pretrained embedding models and picking the right metric as the indicator for choosing network weights is highlighted. 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.

Prediction	Others	Happy	Sad	Angry
I	0.9595	0.7153	0.7806	0.7671
II	0.9595	**0.7190**	0.7801	0.7704
III	**0.9608**	0.7180	**0.7960**	**0.7709**

Table 6: F1-score for each emotion after each step of voting on the Test set.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, and Xiaoqiang Zheng. 2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems.

Alexandra Balahur, Jesús M Hermida, and Andrés Montoyo. 2011. Detecting implicit expressions of sentiment in text based on commonsense knowledge. In *Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis*, pages 53–60. Association for Computational Linguistics.

Christos Baziotis, Nikos Athanasiou, Pinelopi Papalampidi, Athanasia Kolovou, 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.

Rodrigo Rodrigues do Carmo, Anísio Mendes Lacerda, and Daniel Hasan Dalip. 2017. A majority voting approach for sentiment analysis in short texts using topic models. In *Proceedings of the 23rd Brazillian Symposium on Multimedia and the Web*, pages 449–455. ACM.

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*.

Gichag Lee, Jaey Un Jeong, Seungwan Seo, Czang Yeob Kim, and Pilsung Kan G. 2018. Sentiment classification with word localization based on weakly supervised learning with a convolutional neural network. *Knowledge-Based Systems*, 152:S0950705118301710.

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.

Xiaojie Sun, Menghao Du, Hua Shi, and Wenming Huang. 2018. Text sentiment polarity classification method based on word embedding. pages 99–104.

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