YNU-HPCC at SemEval-2018 Task 2: Multi-ensemble Bi-GRU Model with Attention Mechanism for Multilingual Emoji Prediction

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

Abstract

This paper describes our approach to SemEval-2018 Task 2, which aims to predict the most likely associated emoji, given a tweet in English or Spanish. We normalized text-based tweets during preprocessing, following which we utilized a bi-directional gated recurrent unit with an attention mechanism to build our base model. Multi-models with or without class weights were trained for the ensemble methods. We boosted models without class weights, and only strong boost classifiers were identified. In our system, not only was a boosting method used, but we also took advantage of the voting ensemble method to enhance our final system result. Our method demonstrated an obvious improvement of approximately 3% of the macro F1 score in English and 2% in Spanish.

1 Introduction

As a novel means of enhancing the visual effect and meaning of short text messages, emojis are almost indispensable to each social platform, such as Facebook, Twitter, and Instagram. These graphic facial expressions and other object symbols enrich the emotion the user wishes to express in a text-based message. Although they are significant as part of social messages, emojis have scarcely attracted attention from a natural language processing (NLP) standpoint. Notable exceptions include studies focused on emoji semantics and usages (Barbieri et al., 2017). However, the interplay between text-based messages and emojis remains virtually unexplored. The aim of SemEval-2018 task 2 (Barbieri et al., 2018) is to fill this gap by providing all participants with a large set of text-based tweets and their related emojis, which were extracted from original tweet messages, in order to determine the connection between text words and emojis.

In recent years, an increasing amount of research work has been conducted on the sentiment analysis of tweets. Emojis have always played an important role in the sentiment polarity of tweets. Novak et al. (2015) proposed a sentiment map of the 751 most frequently used emojis, by computing the sentiment emoji from the sentiment of tweets in which they occurred, and they determined a significant difference in the sentiment distribution of tweets with and without emojis. As opposed to sentiment polarity prediction of tweets, we further investigated potential tweet emojis in this task, evoked by the text part.

Neural networks involving attention mechanisms have been studied extensively in the image processing and NLP fields, and have demonstrated remarkable results, particularly convolutional neural networks (CNNs) (Collobert et al., 2011) and long short-term memories (LSTMs) (Hochreiter and Schmidhuber, 1997). Cliche (2017) assembled several CNNs and LSTMs and achieved first ranking in all of five English subtasks. Raffel and Ellis (2015) proposed a feed-forward network model with attention, which selects the most important element from each time step using learnable weights, depending on the target. As a variant of LSTM, the gated recurrent units (GRUs) (Cho et al., 2014) use gating units directly in order to modulate the data flow inside the unit, rather than consisting of separate memory cells. For this task, we firstly utilized bi-directional GRUs and the attention mechanism to train the base model, following which we boosted the base model with different sample weights. In order to achieve optimum performance, soft and hard voting ensemble methods were also used in our system.

The remainder of paper is structured as follows. Section 2 provides an overview of task 2. In

Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018), pages 459–465
New Orleans, Louisiana, June 5–6, 2018.

section 3, we describe the architecture of our base model, particularly the attention mechanism, as well as the multi-ensemble methods used in our submission. Section 4 describes our experiment, which consists of system parameters, evaluation metrics, and experimental results for the two subtasks. Finally, in section 5, we list several possible improvement points, and in section 6, we outline our main conclusions.

2 Task Overview

This multilingual emoji prediction task consisted of multi-labeled emoji classification of short tweet texts, and was divided into two subtasks based on the text language: subtask 1 in English and subtask 2 in Spanish. The most frequently used emojis in English and Spanish were employed as labels, so the two task labels differed.

Subtask-1: Emoji Prediction in English

Given a text of tweets in English, its potential emojis are predicted from 20 emoji labels, as follows:

❤️ 😍 😂 💕 🔥 😊 😎 ✨ 💙 😘 📷 🇺🇸 ☀️ 💜 😉 💯 😁 🎄 📸 😜

Subtask-2: Emoji Prediction in Spanish

Given a text of tweets in Spanish, its potential emojis are predicted from 19 emoji labels, as follows:

❤️ 😍 😂 💕 😊 😘 💪 😉 👌 🇪🇸 😎 💙 💜 😜 💞 ✨ 🎶 💘 😁

2.1 Datasets

The organizers provided a huge amount of training data, including 500K tweets in English and 100K tweets in Spanish[1]. We crawled almost the entire set of tweets with Twitter APIs[2], as parts of the training tweets were no longer available. Table 1 provides a detailed distribution of the datasets for subtasks 1 and 2.

Furthermore, all of the used emoji labels extracted from the tweets are the 20 or 19 most frequently used emojis in the languages themselves. Data samples for these classes are gradually decreasing; thus, datasets of the two subtasks are both imbalanced. In subtask 1, the greatest majority class ❤️ includes 106352 (21.7%) samples, nearly ten times that of the lowest minority class 😜, which includes only 12190 (2.5%) samples. For subtask 2, similar to subtask 1, the greatest majority class ❤️ includes 19640 (19.9%) samples, while the lowest minority class 😁 includes 2525 (2.6%) samples.

	Subtask-1	Subtask-2
Train	489660	98506
Develop	50000	10000
Test	50000	10000

Table 1: Datasets of Task2.

Pre-trained Twitter embeddings for English and Spanish were offered in the task. We validated these by using another pre-trained embedding from GloVe[3] (Pennington et al., 2014), and the English embedding offered in the task presented approximately the same performance in our test.

3 System Description

As these two subtasks were rather similar, with the exception of different emoji labels, we used exactly the same thought to train every system of the respective languages. In order to choose the best one as our base model, several models were tested(such as CNN, CNN+LSTM), and superior performance was achieved by using the multi-ensemble methods as the following subsection 3.2.

3.1 Base Model

We created our base model with Bi-GRU (Bahdanau et al., 2014) rather than Bi-LSTM, owing to its faster, efficient, and superior performance. Furthermore, the bi-directional model can concatenate the sentence matrix vector forward and backward at each time step to obtain full sentence information (Irsoy and Cardie, 2014). The attention mechanism was also involved in our model, the model architecture of which is illustrated in Fig. 1, where h_t denotes the hidden vector at each time step, and t means the time step in the input sequence. Vectors in hidden state sequence h_t are fed into the learnable function to produce attention weight α_t. A single vector is computed as the weighted average of h_t.

[1] The crawler and extractor for this task can be downloaded from https://github.com/fvancesco/Semeval2018-Task2-Emoji-Detection/tree/master/dataset

[2] https://apps.twitter.com/

[3] glove.840B.300d from https://nlp.stanford.edu/projects/glove/

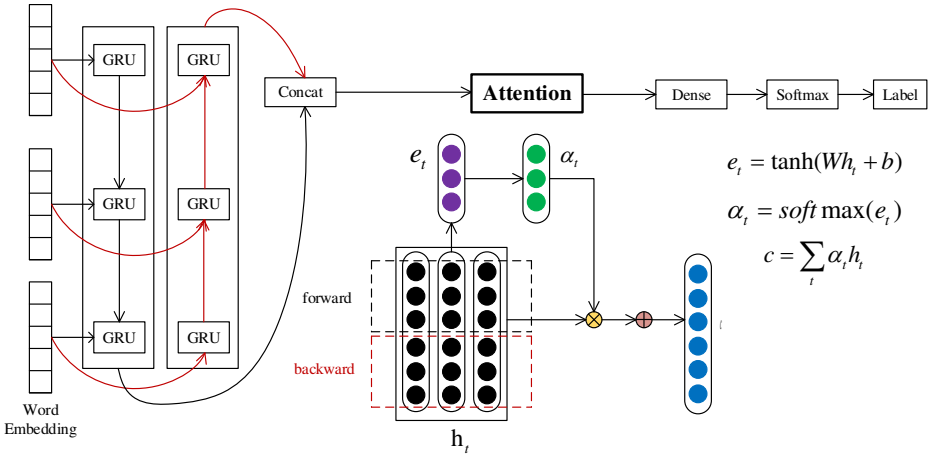

Figure 1: Architecture of Bi-GRU Model with Attention Mechanism.

The characteristic advantage of the attention mechanism over other reduction operations is that it takes an auxiliary context vector as input. The context vector is crucial because it indicates which information to discard, and a summary vector is therefore tailored to the network using it.

3.2 Ensemble

The ensemble classifier is a type of algorithm in which multiple learners are used to improve the individual classification performance by combining hypotheses. In this task, the multiple ensemble methods of boosting and voting were used to achieve the optimal result.

3.2.1 Boosting

The concept of boosting involves iteratively training the base classifier with the re-weighting method, which assigns a new list of sample weights to each round of training data samples. The weight distribution of each sample round depends on the results of the previous round classifier. Simple weights to instances are based on how arduous they are to the classification; thus, the boost classifier provides additional weights to those mis-classified instances following a round of classification in order to boost the following classifier result. Finally, the sequence-based linear weighting method was used to combine the classifiers. Although boosting is not intended for class imbalance problems, owing to this characteristic, it has become an ideal method for class imbalance problems.

3.2.2 Voting

The voting classifier can take a series of machine learning classification algorithms (possibly conceptually different) and average them to obtain final predictions. Two voting methods are available:

Soft voting: Each model outputs a probability vector for all classes, and the voting model is average-weighted to obtain a final probability vector for classification.

Hard voting: Each model outputs what it believes to be the most likely category, from which the voting model selects the category with the largest number of voting models as the final classification.

In several tasks, soft voting may obtain superior results over hard voting. In this case, both voting methods were used. We assembled our strong boosting classifier and weighted Bi-GRUATT by means of soft voting.

4 Experiment and Results

4.1 Pre-processing

Twitter messages include a great deal of irrelevant information that is useless for capturing text message features. Prior to building our model, we utilized a series of operations to normalize tweet messages, divided into the following steps.

Lowercase. We integrated all text words by converting them to lowercase.

Replace Emoticons. We began by pre-processing tweet messages with emoticon replacement. Typographic emoticon symbols that appear sideways, resembling facial expressions, such as :), :-), and (: all mean “smiley face”, while :-(, :(, and):

mean “frowny face”. In spite of the fact that these emoticons could barely be recognized in the pre-embedding vocabulary, they visibly advanced or changed the text message interpretation. We identified these emoticons and replaced them with the words “smile” or “sad” in the English task, and “sonrełr” and “triste” in the Spanish task.

Remove marks. With Twitter as a large social communication platform, we detected user operations in order to remove them:

- remove URL link;
- do nothing with “@” and “@user”;
- remove hashtag mark and retweet mark, such as “#”, “rt”, “&amp”;
- remove numbers and other irrelevant punctuation marks, such as question mark “?”, exclamatory mark “!”, and quotation.

Revise elongated words. Incorrect words remained in the training samples,, elongated with a single letter, such as “Hellooooooo”. We searched for words containing repeated letters consecutively more than three times, and revised the number of letter replications to one.

Abbreviation. People are likely to use abbreviations of certain phrase to create Twitter message, and we changed these abbreviations back into original phrases in the English task. Above 60 common abbreviations of twitter were replaced in the pre-processing. For example, “thx” was changed back to “thanks” and “ASAP” to “as soon as possible”. We determined to skip this step in the Spanish task because we were not familiar with Spanish or its abbreviations.

4.2 Implementation

Following preprocessing, we used a multi-language tokenizer tool, Unitok[4], for tokenization of each task, and then calculated the token number of the longest sentences in the training datasets. All tokens that could be recognized in the provided word embedding were converted into a 300-dimension vector, which would be filled in with zero if it was still unrecognized with the prefix “#”. All training samples were padded out to the same length L, which is 40 in English and 38 in Spanish.

We implemented our model using the Python Keras library with a TensorFlow backend. The properties of our Bi-GRUATT model were as follows.

1. The GRU dropout was set to 0.2, which was the same as the dropout layer following the attention layer.
2. The active function of the dense layer was $tanh$ with a length of 100, and $softmax$ was used to output the prediction label.
3. Instead of using $Sequential.fit$ to train the model, the $Sequential.fit\_generator$ was used for huge training samples in order to solve the memory problem.
4. The optimizer $Adam$ and loss function cross entropy were used to deal with this multi-labeled task.
5. The training epoch was dependent on the early stopping monitor. If the training loss had not improved in the latest 10 epochs, the training process stopped. Both training processes in task 2 exhibited extremely slow improvement in every epoch. In English, the training epoch was approximately 180 to 230, and 280 to 400 in Spanish.

As the ratio of the majority to minority classes could be as high as 10:1, a simple means of enhancing the minority class performance is a cost-sensitive method that applies different costs for mis-classification errors to each class. Usually, there is a high cost for the minority class and a low cost for the majority class. We balanced the training samples with various class weights, and used these to train the cost-sensitive models in the English and Spanish tasks.

Boosting and voting were implemented using the AdaBoosting and Voting classifiers of scikit-learn (Pedregosa et al., 2011) in Python. During the boosting processing, we applied the Bi-GRUATT model with an equal sample weight of $1/N$ (where N is the total number of training samples) as the base estimator, and then iteratively trained a stronger classifier by paying more attention to mis-classified samples. The boosting learning rate was set to 0.001 and the number of estimators was dependent on the result of macro F1. The boosting model would be stopped if it already had three weak classifiers. Only the strong classifier in the boosting process would be used in the voting ensemble, and weak classifiers were ignored.

[4] http://corpus.tools/wiki/Unitok

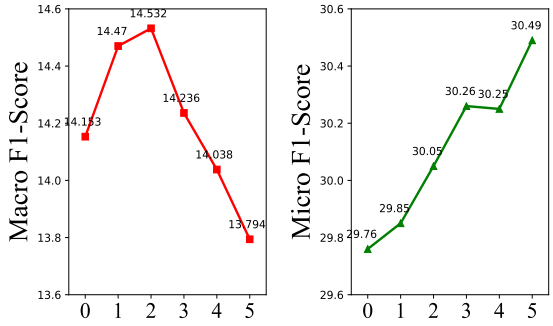

Figure 2: Macro F1 and Micro F1 of boosting classifiers in Spanish. (X-axis means the number of boosting iteration, and number '0' refers to the initial model which was trained with equal sample weight.)

4.3 Evaluation

For evaluation purposes, the official results were based on the macro F1 score for the two tasks. The F1 score was calculated as follows:

$$F_i = \frac{2\pi_i \rho_i}{\pi_i + \rho_i} \quad (1)$$

$$F_{macro} = \frac{\sum_{i=0}^{M} F_i}{M} \quad (2)$$

where F_i is the F1 score of the i-th class, and π_i and ρ_i denote the precision and recall of the i-th class, respectively. The macro average provides an equal weight to all the classes, regardless of how many samples belong to it. The micro average provides equal weights to all the samples, thereby favoring the performance of the majority classes. The macro F1 as well as micro F1 score were utilized to evaluate the performance of every model in our system.

4.4 Results

As the English datasets are fivefold the Spanish ones, we adjusted the system model and its parameters with the Spanish datasets in order to save time. We handled the two subtasks in the same manner; however, the results indicated that not all methods were appropriate for both the English and Spanish tasks.

Subtask-2: Spanish Emoji Prediction

Boosting had generated only six strong or weak classifiers, and the variation tendencies of the macro and micro F1 scores are displayed in Fig. 2.

Subtask-2	Macro F1	Micro F1
Boost_0	14.153	29.76
Boost_1	14.47	29.85
Boost_2	14.532	30.05
Soft Voting	**15.016**	**31.55**
Bi-GRUATT_1	13.768	18.93
Bi-GRUATT_2	14.144	18.86
Hard Voting	**16.015**	**30.22**

Table 2: Results of each model and voting ensemble in Spanish.

❤️	😍	😂	💕	😊
38.03	29.39	52.69	0	14.93
😘	💪	😉	👌	🇪🇸
24.72	35.11	13.68	8.86	28.43
😎	💙	💜	😜	💞
12.99	1.87	0	2.43	0
✨	🎶	💘	😁	
14.19	19.77	2.82	4.38	

Table 3: Macro F1 of each emoji label in Spanish.

It was found that, as the iteration times increased, the micro F1 score (accuracy) of the boosting system continuously increased, while the macro F1 score was increased in the first three iterations and then decreased in the following iterations. As the macro F1 score provides the official evaluation index, we identified the first three strong boost classifiers for the further ensemble. In the voting ensemble part, the three strong boost classifiers were used to achieve stronger classification by means of soft voting, and eventually, we integrated the soft voting result and the results of the two basic Bi-GRUATT models trained with a balanced class weight (same model but trained twice). The results of these models can be seen in Table 2. Soft voting achieved a 0.9% improvement in the macro F1 score, while the hard voting achieved 1% compared to the soft voting results.

Furthermore, hard voting, which provided the final result, demonstrated a 2% improvement in the weighted Bi-GRUATT model. Table 3 displays the macro F1 for each emoji label.

Subtask-1: English Emoji Prediction

Similar to the process of subtask 2, we obtained four boosting classifiers, the variation tendencies of which are illustrated in Fig. 3. The macro F1

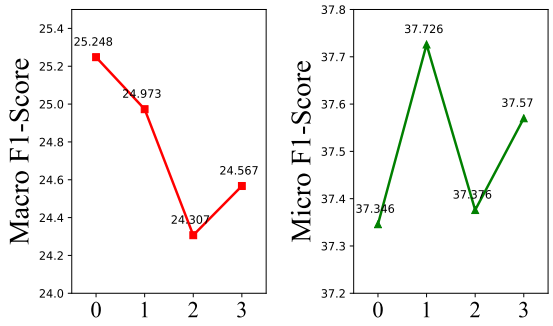

Figure 3: Macro F1 and Micro F1 of boosting classifiers in English. (X-axis means the number of boosting iteration, and number '0' refers to the initial model which was trained with equal sample weight.)

Subtask-1	Macro F1	Micro F1
Boost_0	25.248	37.346
Bi-GRUATT_1	25.656	27.556
Bi-GRUATT_2	25.552	27.28
Hard Voting	**28.112**	**35.196**

Table 4: Results of each model and voting ensemble in English.

score was decreased following each iteration. We could not conduct soft voting with only one strong classifier, but carried out the hard voting directly. Table 4 presents the results of the weighted Bi-GRUATT and hard voting. The hard voting in the English task demonstrated a 2.5% improvement in the weighted Bi-GRUATT and 2.9% improvement in the Bi-GRUATT without class weights. The macro F1 for each emoji label is displayed in Table 5.

All above-mentioned results were obtained following the competition, and we changed the learning rate from 0.01 to 0.001 for increased boost classifiers. The competition submission only used two boost classifiers for ensemble in Spanish and one was utilized in English. We ranked 16th among 48 teams in the English task and 13th among 21 teams in the Spanish task[5].

As indicated in Tables 4 and 5, predictions of similar smiley face emojis (😊😉😎🤤😁 in Spanish and 😊😉😁🤤 in English) exhibited unstratified results. There were not many characteristics to distinguish these from one another, as they all meant a happy feeling, and people had a different

[5]Results of all teams can be found in https://goo.gl/P515KW

❤️	😍	😂	💕	🔥
44.53	30.6	43.98	15.79	54.17
😊	😎	✨	💙	😘
10.84	17.96	31.46	17.53	16.95
📷	🇺🇸	☀️	💜	😉
28.09	55.64	37.7	11.59	12.1
💯	😁	🎄	📸	😜
25.22	11.68	66.86	24.97	4.59

Table 5: Macro F1 of each emoji label in English.

understanding and favorite emojis, which could cause certain confusion problems. Compared to the confused smiley face emojis, the label 🎄 demonstrated the best result, although it was a minority class. It was easy to predict, owing to the label 🎄 having an obvious feature to recognize, such as "tree", "Christmas Tree" or everything relevant to Christmas.

Another confused set of emojis were series of heart emojis, particularly 💕💙💜💞💘, and in subtask 2, some of these labels even obtained a zero result. This illustrates that the model preferred to predict the highest majority class of the same series, such as ❤️ rather than its actual emoji 💕, 💙 or other heart.

5 Discussion

In this section, we briefly discuss several points that may enhance our system results. For time reasons, we have not validated these yet.

Pre-processing

1. We simply removed hashtag mark "#", however, there were numerous unrecognized hashtag words, such as "Iamhappy", "I_am_happy" or "I-Am-Happy". A preferable method is decomposing the hashtag to invert it into a sequence of words that would enrich the information of the sentence samples (Billal et al., 2016).

2. We did not deal with the mis-spelled words, and changed all elongated words into the without repeated letters that would create mistake words, like "foooood".

Model

1. As the official evaluation index is the macro F1 score, we should set the initial sample weight to be the same as the balanced class weight in the boosting process.

2. We only trained less than 10 estimators in the boosting, which may be too small to determine the boost variation tendency in this task.

3. We only used one base model to ensemble. Assembling with different model types may achieve improved results, such as CNN, a combination of CNN and LSTM, and random forest.

6 Conclusion

In this brief paper, we have presented the system we used to compete in the SemEval-2018 task 2: Emoji Prediction in English and Spanish. Our submission system was based on a Bi-GRU model with an attention mechanism, and we aimed to ensemble the base model with multi-method boosting and voting in order to achieve superior performance. In our work, multi-ensemble exhibited a 2 to 3% improvement in the task results.

In the future, we plan to validate the points in the Discussion section in order to verify their influence of them, and add useful points to our system to implement an improved model for emoji prediction.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (NSFC) under Grant No.61702443 and No.61762091, and in part by Educational Commission of Yunnan Province of China under Grant No.2017ZZX030. We would like to thank task organisers and the anonymous reviewers for their helping and constructive comments.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Francesco Barbieri, Miguel Ballesteros, and Horacio Saggion. 2017. Are emojis predictable? *CoRR*, abs/1702.07285.

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 (SemEval-2018)*, New Orleans, LA, United States. Association for Computational Linguistics.

Belainine Billal, Alexsandro Fonseca, and Fatiha Sadat. 2016. Named entity recognition and hashtag decomposition to improve the classification of tweets. In *Proceedings of COLING*, pages 102–111.

KyungHyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *CoRR*, abs/1409.1259.

Mathieu Cliche. 2017. Bb_twtr at semeval-2017 task 4: Twitter sentiment analysis with cnns and lstms. In *Proceedings of SemEval*, pages 573–580.

Ronan Collobert, Jason Weston, Lon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

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

Petra Kralj Novak, Jasmina Smailović, Borut Sluban, and Igor Mozetič. 2015. Sentiment of emojis. *CoRR*, abs/1509.07761.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, and Vincent Dubourg. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct):2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing(EMNLP)*, volume 14, pages 1532–1543.

Colin Raffel and Daniel P. W. Ellis. 2015. Feed-forward networks with attention can solve some long-term memory problems. *CoRR*, abs/1512.08756.