EmoNLP at SemEval-2018 Task 2: English Emoji Prediction with Gradient Boosting Regression Tree Method and Bidirectional LSTM

Man Liu

liumanlalala@gmail.com

Abstract

This paper describes our system used in the English Emoji Prediction Task 2 (Barbieri et al., 2018) at the SemEval-2018. Our system is based on two supervised machine learning algorithms: Gradient Boosting Regression Tree Method (GBM) and Bidirectional Long Short-term Memory Network (BLSTM). Besides the common features, we extract various lexicon and syntactic features from external resources. After comparing the results of two algorithms, GBM is chosen for the final evaluation.

1 Introduction

Short text messages from social media websites such as Twitter and Facebook have become an important communication channel in our daily life. Although the writing styles of such short text messages are extremely diverse, the usage of emojis are generally shared. Emojis are ideograms and smiles that can be electronic expressions of natural emotions. Genres of emojis vary from facial expression, places, types of weather to animals. Emojis are used every day, rapidly changing the communication way in the social network. Due to the importance of emojis, investigations about emojis have been performed in recent years. For example, the previous work (Barbieri et al., 2017) has shown that there are relations between words and emojis, in other words, emojis are predictable given its surrounding words. Task 2 of SemEval 2018 provides a platform for the further prediction of emojis on tweets.

Our system addresses the first subtask: English emoji prediction. Note that all the emojis in the training data are removed. We models this problem as a multiclass classification problem. Specifically, we leverage on semantic and syntactic resources to extract varieties of features. After feature engineering, two models are adopted: gradient boosting regression tree method (GBM) and bi-directional long short-term memory network (BLSTM). After comparing results of these two models, GBM is selected for the final evaluation.

The reminder of this paper is structured as follows. In section 2, we describe our system in detail, including the feature description and approaches. In section 3, results of 5-fold experiments and feature ablation are presented. Finally, section 4 summarizes our work.

2 System Description

In this section, we present the details of our English emoji prediction system. In the dataset, each tweet corresponds to one label indicating one type of emojis. There are 20 types of emojis, most of which are emotions.

We treat the problem as a multiclass prediction task and extract a variety of features. For the model GBM, besides the common features such as word *n-gram* features, we utilize extensive external resources to build diverse word clusters, lexicon and syntactic features. For the model BLSTM, we adopt the pre-trained word embedding GloVe (Pennington et al., 2014).

2.1 Preprocessing

As the first step, we perform preprocessing for tweets tokenization and normalization. The tokenization of all tweets are performed using **tweetokenize**[1]. In addition, we normalize tweets by replacing all the URLs (*e.g. https://t.co/bihPimeeV9*) with "*_url_*" and all the mentions (*e.g. @Preston Hall*) with "*@mention*".

[1] https://github.com/jaredks/tweetokenize

Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018), pages 390–394
New Orleans, Louisiana, June 5–6, 2018.

2.2 Features

This section briefly describes features employed in our two models. GBM takes advantage of all the features shown in this section while BLSTM only utilizes pre-trained GloVe as the word embedding. For GBM, each tweet is represented as a feature vector consisting of all the following features. Since most of the target emojis are related to emotions, we employ diversiform lexicon features as well as emotional word (e.g. ":)") features to exploit the sentimental information in the sentence.

Character ngram: This feature represents the presence or absence of contiguous sequence of 3, 4 and 5 characters to capture the morphological information hidden in the words.

POS: The POS tag presents the information about the lexical type of the word. We part-of-speech tag the tweets with the Carnegie Mellon University (CMU) tool (Gimpel et al., 2011). This tool is designed specifically for tweets pos tagging with the capability to deal with the non-standard words. For example, it can tag "ikr" ("I know, right?") as an interjection.

Cluster: This feature is induced from CMU pos-tagging tool which provides the word cluster using the Brown clustering algorithm. These pre-trained 1,000 clusters serve as alternative representations of each tweet. This feature illustrates the presence and absence of tokens from the 1,000 clusters.

Negation: This negation feature is generated followed by Mohammad's work (Mohammad et al., 2013), where a negated context is defined from a negation word (e.g. no, none) to one of the punctuation marks:",", ".", ":", ":", "!", "?". Each word in the negated context is added with the suffix *"_NEG"* .

Word ngram: The word ngram (*n=1, 2, 3, 4*) features are captured after negation processing and can explore additional context information.

Counting feature: This feature is inspired by Mohammad's work (Mohammad et al., 2013) and developed by combining all the number of special symbols (e.g. mentions) in each tweet.

- the number of hashtags;
- the number of words with all characters in upper case;
- the number of contiguous sequences of question marks, exclamation marks, and both of them;
- whether the last token contains an question mark or exclamation;
- the number of mentions;
- the number of URLs;
- the number of words which have repeated characters (e.g. "coooool");
- presence or absence of positive or negative emoticons in the tweet. The positive and negative emoticons are defined in (Mohammad et al., 2013).

SSWE feature: SSWE (sentiment-specific word embedding) are learned on 10 million tweets using customized neural network (Tang et al., 2014). SSWE features can capture the sentiment information of sentences as well as the syntactic context of words.

Lexicon feature: This feature is created following the method produced by (Mohammad et al., 2013). We investigate the number of sentiment words, the total sentiment score, the score of last sentiment words and the maximal sentiment score for each lexicon. Taking advantage of extensive external lexicons [2], this feature can interpret the sentimental information in tweets comprehensively.

2.3 Model

Two models are used in our system: GBM and BLSTM. We compare performances of these two models and finally use the GBM result for final evaluation.

GBM: We construct GBM model by using all the features mentioned in section 2.2. For each tweet, the concatenated feature vector can be used as the model input. Outputs of the model are the multiclassification results. The gradient boosting regression tree method generates base models sequentially and at each step updates the base model by minimizing the loss function value. The base model is a single regression tree which fits a set of features by partitioning the feature space into

[2]NRC Emotion Lexicon; NRC Hashtag Sentiment Lexicon; MaxDiff Twitter Lexicon; MPQA Effect Lexicon; MPQA Subjectivity Lexicon; Harvard Inquirer Lexicon; Bing Liu Lexicon; Loughran McDonald Lexicon; Amazon Laptop Review Lexicon; Sentiment140 Lexicon.

different regions. With additional regression trees added to the model, the fitted model may achieve a small training error. In other words, the gradient boosting method sequentially fits the training data by correcting base models at each step to strategically yield the best combination of trees. Therefore, the gradient boosting method is potential to produce more accurate predictions results (Zhang and Haghani, 2015). The tool we used to build GBM model is lightGBM[3]. We tune the hyperparameters on the training set by grid search. Because of the time constraints, it is impossible to tune all the hyperparameters in the GBM model. We choose two hyperparameters to tune and we set learning rate to 0.1 and minimal number of data in one leaf to 20. Table 1 summarizes the final settings and the search space of hyperparameters.

Parameters	Setting	Search space
number of leaves per tree	64	16, 32, 64, 128
number of trees	300	100, 300, 700, 1000

Table 1: Tuned hyperparameter values and search space for GBM.

BLSTM: We experiment BLSTM with published word embedding, namely Stanford's GloVe embedding[4] trained on 6 billion words from Google and Web text. Instead of a traditional feedforward network, we use the bi-directional long-short term memory network. LSTM (Hochreiter and Schmidhuber, 1997) is a powerful connectionist model that can capture time dynamics and it has special capability to cope with these gradient vanishing problems compared with the traditional recurrent neural network (RNN). However, LSTM only has access to process one directional information in the sequence which is contradictory with most of the practical situations where the bi-directional information is both beneficial. BLSTM is designed to deal with this problem with the basic idea to process the sequence backward and forward and feed the output into two separate hidden states to catch the past and future information (Ma and Hovy, 2016).

We exploit this BLSTM transforming word embedding into classification results. Figure 1 shows the network in detail. We also tuned the hyperparameters in BLSTM to achieve the best result. The tuned parameter values used are illustrated in table 2.

[3]https://lightgbm.readthedocs.io/en/latest/

[4]https://nlp.stanford.edu/projects/glove/

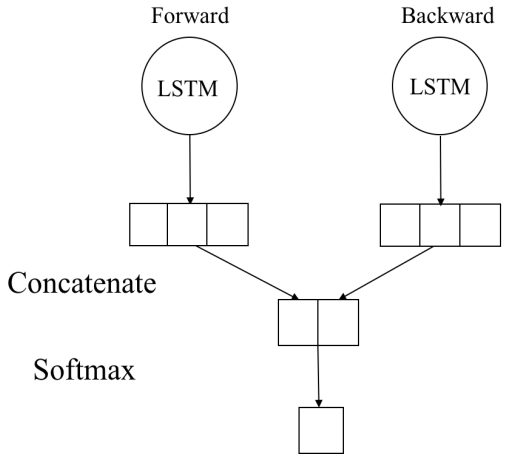

Figure 1: Architecture of BLSTM. The word embeddings of each word are fed into forward LSTMs and backward LSTMs. The outputs of each network are concatenated and decoded by a softmax function into probability for each category.

Parameters	Setting
LSTM units	64
LSTM dropout	0.2
Recurrent_dropout	0.2
Optimizer	rmsprop
Loss function	categorical_crossentropy

Table 2: Tuned hyperparameter values for BLSTM.

2.4 Results

Our system is trained on two models. With the fine-tuned hyperparameters exhibited in table 1 and 2, we train the two separate models to determine the final classifier for evaluation.

To find out the optimum settings, we explore all the training data and conduct 5-fold cross-validation experiments. Table 3 shows the 5-fold cross-validation performances on the two models. Comparing the 5-fold cross-validation results, we observe that there is no significant difference across these 5 experiments.

By comparing results of the two models, GBM outperforms in the official macro F1 score. We finally submit the evaluation result of GBM. The precision, recall and macro F1 score of the final evaluation are 39.426, 33.695 and 33.665 respec-

	Micro F1	Macro F1
GBM	47.16 ± 0.157	33.90 ± 0.119
BLSTM	21.73 ± 0.114	1.78 ± 0.007

Table 3: 5-fold cross-validation results of GBM and BLSTM.

Emo	P	R	F1	%
❤️	90.55	84.14	87.23	21.6
😍	27.43	56.02	36.83	9.66
😂	34.24	62.51	44.24	9.07
💕	26.88	26.91	26.89	5.21
🔥	56.31	51.99	54.06	7.43
😊	14.5	12.9	13.65	3.23
😎	26.64	16.93	20.7	3.99
✨	40.01	24.99	30.77	5.5
💙	35.51	14.72	20.81	3.1
😘	22.61	17.11	19.48	2.35
📷	28.61	51.61	36.81	2.86
🇺🇸	67.46	62.24	64.75	3.9
☀️	74.35	54.07	62.61	2.53
💜	47.85	7.0	12.22	2.23
😉	18.62	4.98	7.85	2.61
💯	37.56	20.5	26.52	2.49
😁	16.58	2.69	4.63	2.31
🎄	67.31	81.17	73.59	3.09
📸	40.49	19.65	26.46	4.83
😜	15.0	1.78	3.19	2.02

Table 4: Precision, Recall, F-measure and percentage of occurrences in the test set of each emoji.

tively. Our system achieves the best accuracy in this task. In addition, the precision, recall, F1 and percentage of occurrences in the test set of each emoji are shown in table 4.

2.5 Further Analysis of Feature Engineering

Table 5 shows the F1 score and loss on the test set resulting from training with each group of feature removed. The experiment performance reveals that all features except SSWE in our system are helpful. The performance drops after adding SSWE features. This observation is probably caused by overfitting on the training set because of the curse of dimension.

Feature	F1 loss	F1 score
Character ngram	1.161	32.74
POS	0.167	33.73
Cluster	0.15	33.75
Word ngram	0.503	33.40
Counting	0.176	33.72
SSWE	-0.235	34.14
Lexicon	0.094	33.81

Table 5: Feature ablation study using the GBM model. The quantity is the F1 loss and score resulting from the removal of each feature group.

3 Conclusion

In this paper, we present the systems used in SemEval 2018 task 2 for English emoji prediction. Our effort focuses on putting forward two models to improve the multi emojis classification. By leveraging on general features (i.e. word ngram feature, character ngram feature and counting features), external resources (i.e. a variety of manual constructed lexicons, CMU brown cluster), feature selection and hyperparameters fine-tuning, GBM achieves better performance than BLSTM. This observation is attributed to the extensive usage of sentimental and syntactic features. Due to most of the target emojis are related emotions, these sentimental features can reveal the relation from words with the emotional emojis. In future, we hope to improve our BLSTM model by taking advantage of more features and incorporating more effective architecture.

References

Francesco Barbieri, Miguel Ballesteros, and Horacio Saggion. 2017. Are emojis predictable? In *Lapata M, Blunsom P, Koller A, editors. 15th Conference of the European Chapter of the Association for Computational Linguistics; 2017 Apr 3-7; Valencia, Spain. Stroudsburg (PA): ACL; 2017. p. 105-11.*. ACL (Association for Computational Linguistics).

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)*. Association for Computational Linguistics, New Orleans, LA, United States.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein,

Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A Smith. 2011. Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*. Association for Computational Linguistics, pages 42–47.

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

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. volume 1, pages 1064–1074.

Saif Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013. Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets. In *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*. volume 2, pages 321–327.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. pages 1532–1543.

Duyu Tang, Furu Wei, Bing Qin, Ting Liu, and Ming Zhou. 2014. Coooolll: A deep learning system for twitter sentiment classification. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*. pages 208–212.

Yanru Zhang and Ali Haghani. 2015. A gradient boosting method to improve travel time prediction. *Transportation Research Part C: Emerging Technologies* 58:308–324.