EmotionGIF-Delta: Ensemble Method for Twitter GIF Recommendation

Ming-Han Huang, Wei-Lun Fong and **Sz-Yu Chen**
National Chiao Tung University, HsinChu, Taiwan
{hannnnk.iie08g, warrenpig.cs08g, broodfish.cs08g}@nctu.edu.tw

Abstract

This paper describes our approach to the EmotionGIF-2020, the shared task of SocialNLP 2019. To recommend GIF categories for tweet responses, we ensemble our LSTM model with three state-of-the-art pre-trained models: BERT, XLNet and RoBERTa. These public models are well-trained on a very large data set, which is a mission-impossible for most of people with only one or two GPU. We can benefit from them and easily achieve our target with very good performance by just fine-tuning them. The presented method achieve 53.45% in the test set and 53.14%, 53.65% in GIF with text and GIF only respectively.

1 Introduction

Twitter is a very popular communication platform which enables users share their information with their friends or the pubic by tweeting a message. Another user can give a response to the tweets or retweet a tweet. It's getting popular to reply with a GIF instead of texts since it is more convenience and more lively to express their reaction to the tweet. The GIFs are categorized into several categories (*e.g.* hug, thank you), which makes users find the desired GIF quickly. In this paper, we try to recommend the user at most 6 categories of GIF according the tweet they are going to reply and the replying texts they've typed (if exists).

Learning from text has drawn a great attention in recent years. There are more and more algorithms or models, such as CNN and RNN, can help us dealing with the natural language processing problem (Chollet, 2017). As the rise of big data and open sourcing, there are lots of state-of-the-art public pre-trained models can help us get a very good result by just fine-turning them. We apply transfer learning on three pre-trained models: BERT (Devlin et al., 2018), XLNet (Yang et al., 2019) and roBERTa (Liu et al., 2019), and ensemble them with our own LSTM model to achieve a better result.

The rest of paper is organized as follows: Section 2 describes our system architecture, including the preprocessing procedural, model we used for this task and the ensemble method. The experiment result is in Section 3, and Section 4 conclude the paper and the future work.

2 System Architecture

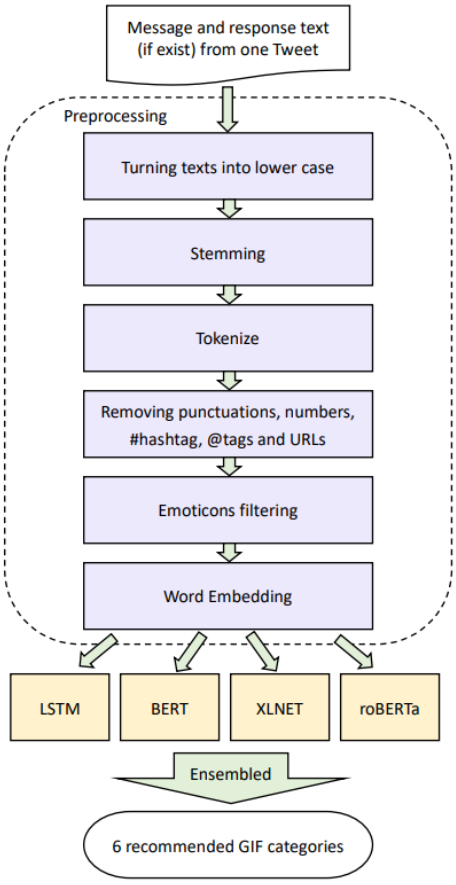

Figure 1: Framework of recommending 6 GIF categories for responding a tweet

Figure 1 shows the framework of our system for recommending 6 GIF categories for a tweet. A preprocessing pipeline is designed to deal with noise in Twitter messages and turning texts into word vectors for each model. Then weight results from each model, averaging them into the final 6 categories.

2.1 Preprocessing

In the pre-processing pipeline, we first convert all the texts into the lower case for better embedding result and prediction. Then, since the abbreviation and the internet slang words, which are widely used in Twitter messages, are mostly out-of-vocabulary words for the embedding system (Wu et al., 2010), we restore the abbreviation such as "I've" or "you're" back into their original form "I have" and "you are" manually. Also, we build up a lookup table based on Glove's vocabulary book to find all the invalid tokens. We grab 40 tokens according to their frequency and restore them into valid words, such as "lmao"(laugh my ass off) or "yyyeeessss"(yes).

We apply the TweetTokenizer from the NLTK toolkit, which can best deal with the Twitter texts. Then, benefit from this tokenizer, we can easily remove the punctuation, numbers, hashtags, tags and URLs, those we think are useless and noisy for our prediction, from the text.

Emoticons frequently appear in Twitter messages. We found that some of them is highly related to the prediction, and the others are not. Also, this relation depends on the place of the text they are. A emoji appears in a response text may highly related to our prediction but it may not when it appears in the main text. Hence, we remain some emoticons for the text and the response text respectively and remove the others. Figure 2 shows the emoticons we remain.

In the end of the preprocessing pipeline, we apply different word embedding technologies for different models, and apply zero padding to get the fixed length input. We will explain the details of word embedding specifically in the next subsection respectively.

2.2 Models

2.2.1 LSTM

Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture used in the field of deep learning (Liu et al., 2016). Figure 3 shows our LSTM architecture. The first layer of the model is an embedding layer, which can help the model deal with encoded input. We use GloVe (Pennington et al., 2014) with the Twitter word vectors(uncased, 100d) to encode the tokens and build the embedding matrix. However, emoticons are not contained in the GloVe word vectors. We fine-tune another word embedding model emoji2vec (Eisner et al., 2016), which is an emoticon specific word vector, to work with GloVe.

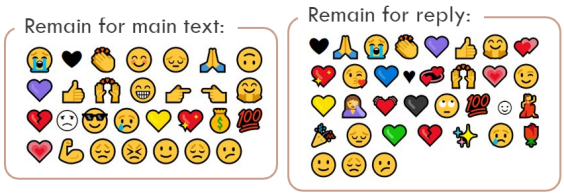

Figure 2: Remaining emoticons for the main text and the reply text respectively

Figure 3: LSTM architecture

2.2.2 BERT

Bidirectional Encoder Representations from Transformers (BERT) is a language representation model. The pre-trained BERT model can be fine-tuned with just one additional output layer for our tasks (Devlin et al., 2018). The embedding consists of three parts: token embedding, attention mask and token type. For token embedding, we must put [CLS] token in front of the sentence and use [SEP] token to separate sentences. For example:

text: "I love NLP!", reply: "Me too!"

becomes

[CLS] i love nlp [SEP] me too [SEP]

Since the emoticons is out-of-vocabulary for BERT model, we also demojize the emoticons into the texts. We use the base uncased pre-trained BERT

Model	P (all)	P1(GIF w/ text)	P2 (GIF only)
LSTM	50.5%	45.92%	**53.5%**
BERT	52.75%	**53.68%**	52.15%
XLNet	52.23%	52.41%	52.11%
RoBERTa	52.4%	52.85%	52.1%
Ensemble	53.96%	53.95%	53.97%
Weighted Ensemble	**54.41%**	**54.45%**	**54.39%**

Table 1: Model performances on the development set

Model	epochs	batch size	optimizer	learning rate
LSTM	20	32	RMSprop	1e-2
Pre-Trained Models	5	16	Adam	1e-2

Table 2: Configurations for four models.

model, which is trained on lower-cased English text, for our task.

2.2.3 XLNet

XLNet is a generalized autoregressive pretraining method that enables learning bidirectional contexts and overcomes the limitations of BERT (Yang et al., 2019). The embedding procedure is similar to the BERT model. We use the base pre-train XLNet model and fine-tune the model with one dense output layer.

2.2.4 RoBERTa

A robustly optimized BERT pretraining approach (RoBERTa) is a state-of-the-art technique (Liu et al., 2019). Different from BERT model, the token embedding no longer needs a [CLS] token in front of the texts and instead, it uses <s >and </s >to separate sentences. For example:

text: "I love NLP!", reply: "Me too!"

becomes

<s >i love nlp </s ><s >me too </s >

We use the base pre-train RoBERTa model and fine-tune the model with one dense output layer for our task.

2.3 Ensemble

Ensemble learning helps improve the predictive performance by combining multiple learning algorithms. The reason why this method can succeed is because with more models, the more aspects we can see and thus closer to the whole picture.

The most naive way to do the ensemble is getting the new result by taking the average of the results from models. However, from the Table 1, we can see that each model has its own strengths, we should not treat them equally. Also, since LSTM performs very worse in the GIF w/ text task, we only use the GIF only part of LSTM. Thus, we conduct grid search to try different weights on the development set.

$$res_{ensemble} = \Sigma w_i res_i \quad (1)$$

3 Experiment

3.1 Experiment Setup

The experiment is based on the development set, 4000 unlabeled samples used for practice. Four models are trained with different configurations, which can be found at Table 2. Since this competition is a multi-label task instead of a multi-class task, the activation function of the output dense layer is sigmoid, which give us probabilities of each category. With evaluating metric, mean recall at 6 (MR@6), we ensemble the results with weights and select the top 6 categories as our prediction.

3.2 Experiment Result

Table 1 shows our result on the development set at the practice & development phase. It shows that each model has its own strengths, the BERT model outperform others in the GIF w/ text and the LSTM model outperform in GIF only. Also, the weighted ensemble method outperform the naive ensemble method in both tasks. Table 3 and Table 4 show our final result on the test set.

4 Summary

In this work, we show how to use LSTM and conduct transfer learning on three state-of-the-art mod-

P (all)	P1(GIF w/ text)	P2 (GIF only)
50.2%	44.76%	53.82%

Table 3: Round 1 result on the test set use only LSTM

P (all)	P1(GIF w/ text)	P2 (GIF only)
53.45%	53.14%	53.65%

Table 4: Round 2 result on the test set use ensemble method

els such as BERT, XLNet and RoBERTa to deal with a multi-label natural language processing task. Also, we show their based model performance and the power of ensemble.

In the future, since three pre-trained model we used in the task are very similar (they are based on each other), we plan to try more algorithms such as TF-IDF or FastText and ensemble them with our current model to get better result. Also, there are still larger pre-trained model for BERT, XLNet and RoBERTa. We will definitly try them as we get a better GPU or TPU.

References

Francois Chollet. 2017. *Deep Learning with Python*, 1st edition. Manning Publications Co., USA.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

Ben Eisner, Tim Rocktäschel, Isabelle Augenstein, Matko Bošnjak, and Sebastian Riedel. 2016. emoji2vec: Learning emoji representations from their description.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

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, Doha, Qatar. Association for Computational Linguistics.

Wei Wu, Bin Zhang, and Mari Ostendorf. 2010. Automatic generation of personalized annotation tags for twitter users. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, page 689–692, USA. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding.