EmotionGIF2020 Technical Repot

Anonymous ACL submission

Abstract

This document is the technical report of final project by NCTU Team Yellow. We fine-tune the pre-trained BERT model and applied on the multi-label classification task with 43 labels.

1 Introduction

Jacob Devlin *et.al.*(Devlin et al., 2018) has introduced a new language model named Bidirectional Encoder Representation from Transformer, as know as BERT based on the "Self-Attention Mechanism"(Vaswani et al., 2017). BERT is a strong model which can handle varies of tasks by fine tuning the pre-trained model.

In order to train the language model more efficient, we choose to fine tune a BERT pre-trained model to applied on the specific task. Refer to works from Kaggle's Toxic Comment Classification Challenge, we consider our task as multi-label classification task, and fine tune the BERT pre-train model to fit our goal.

The rest of the report is organized as follows: Section II introduced the base model(pre-train model) we chosen, and the original BERT tokenizer we used. The data pre-processing including BERT features extraction are illustrated in Section III. We further discuss the training and prediction results in Section IV, and draw some conclusions at the end of this report in Section V.

2 Proposed Architecture

As shown in Figure.1, the pre-trained model we chose is "Uncased, L-12, H-768, A-12", in which **the model has 12 layers, 768 hidden and 12 heads.** The model has 110M parameters, and could train 4 days using 16 TPU chips. Therefore, instead of proceed this step, we just download the open source model provided by the authors and fine tune the model to fit our task.

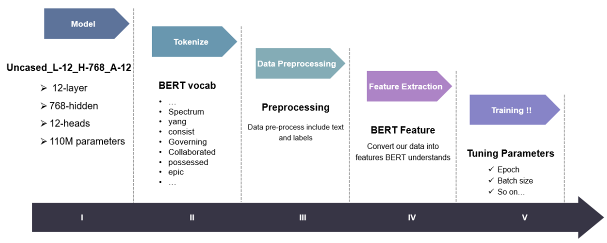


Figure 1: System Architecture

In tokenize step, we just applied the original BERT tokenizer, which will execute the following:

- **Text Normalization :** Convert all the characters to lower case in our case.
  - **Origin :** "John Johanson's,"
  - **Tokenized :** "john johanson's,"
- **Punctuation Splitting :** Add a space in front of and after a punctuation.
  - **Origin :** "John Johanson's,"
  - **Tokenized :** "john johanson ' s , "
- **WordPiece Tokenization :** Convert all the words according to the BERT vocabulary.
  - **Origin :** "John Johanson's,"
  - **Tokenized :** "john johan ##son ' s , "

3 Data Pre-processing

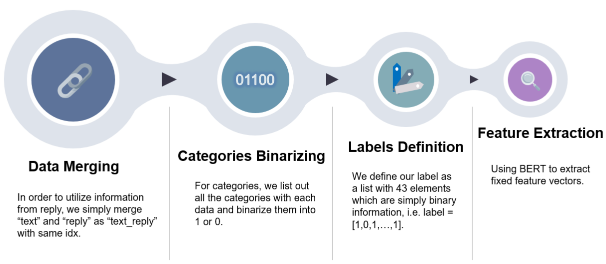


Figure 2: Data Pre-processing

During data pre-processing, we can split the procedure into four parts shown as Figure.2, details will be described in following:

Data Merging: For the purpose of taking both text messages and reply messages into consideration, we simply concatenate two messages with same idx into a single message. Take data with idx 0 as example below:

- Origin :
  - **text :** ”we can all agree that any song by Niall Horan.”
  - **reply :** ”oui oui.”
- Merged :
  - **text-reply :** ”we can all agree that any song by Niall Horan. oui oui.”

Categories Binarizing: In order to convert the categories into vectors that machine can understand, we converted categories for each data into binary form refers to one-hot embedding. We first sort all 43 categories according their letters. At each data, we scan all 43 categories to see if it shows in the data, and convert it into 1 and 0. Again, we take data with idx 0 as example below:

- Origin :
  - **categories :** ”yes”
- Binarized :
  - **categories vector :** [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]

Labels Definition: In our model, we define our labels as a vector with 43 dimensions corresponding to 43 different categories, which is obtained in previous step. As a result, we will get a 1x43 vector in prediction output, in which each element is a probability corresponding to each categories between 0 and 1.

Feature Extraction: The last step in data preprocessing is to convert all data into features BERT understand, we simply separate the process into five steps described as following:

- Zero Padding : In order to parallelize operations, we need to fill each input sequence in the batch with zero padding to ensure that its length is consistent.
- Converting Special Token :
  - **[CLS]** : Label token
  - **[PAD]** : Zero padding mask
  - **[UNK]** : Unknown word
  - **[SEP]** : Sentence separate
- Masking : To distinguish the range of self-attention, 1 means to pay close attention, 0 means no.
- Position Embedding : Embedded positional information.
- Segment Embedding : Embedded the separation of sentences, 0 for first sentence, 1 for second one. In our work, we consider **”text-reply”** as a single sentence, so it will be only 0 in our case.

Figure.3 shows the result of our data pre-processing steps.

```
Token ID :
[101, 1030, 2402, 3207, 4478, 1035, 1045, 2228, 2065, 1057, 5831, 1998, 11122, 2072, 7510, 6928, 2017, 10657, 4530, 13451, 24185, 4430, 2036, 102, 0, 0, 0, 0, 0
Mask :
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
Segment embedding :
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
Label ID :
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
```

Figure 3: Feature Extraction Result

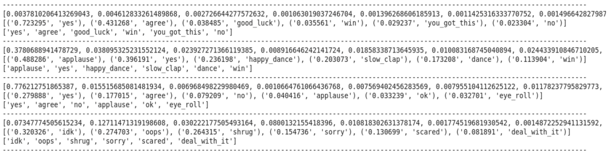
4 Experiments

Our training parameters are describes as following:

- Batch Size : 16
- Epoch Number : 8
- Loss Function : Binary cross-entropy with logits
- Learning Rate : 3e-5
- Output Layer : Sigmoid

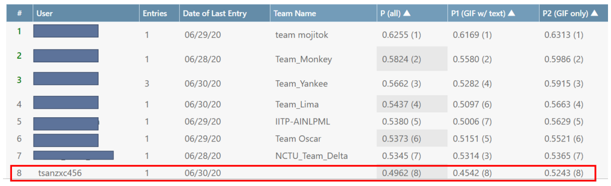
We spend about 4 hours to finish our training, and the final loss is **0.13366494**.

Figure.4 and Figure.5 shows the final prediction result and our score in main track round 2. In Figure.4, first row shows the raw output in prediction, second row shows the 6 categories with highest probabilities, and the third row shows the final prediction result.



```
------------------------------------------------------------------------------------------------------------------------
[0.0037810206413269043, 0.004612833261489868, 0.002726644277572632, 0.001063019037246704, 0.001396268606185913, 0.0011425316333770752, 0.001496642827987
[('0.723295', 'yes'), ('0.431268', 'agree'), ('0.038485', 'good_luck'), ('0.035561', 'win'), ('0.029237', 'you_got_this'), ('0.023304', 'no')]
['yes', 'agree', 'good_luck', 'win', 'you_got_this', 'no']
------------------------------------------------------------------------------------------------------------------------
[0.3780688941478729, 0.038095325231552124, 0.023927271366119385, 0.008916646242141724, 0.01858338713645935, 0.010083168745040894, 0.024433910846710205,
[('0.488286', 'applause'), ('0.396191', 'yes'), ('0.236198', 'happy_dance'), ('0.203073', 'slow_clap'), ('0.173208', 'dance'), ('0.113904', 'win')]
['applause', 'yes', 'happy_dance', 'slow_clap', 'dance', 'win']
------------------------------------------------------------------------------------------------------------------------
[0.776212751865387, 0.015515685081481934, 0.006968498229980469, 0.0010664761066436768, 0.007569402456283569, 0.007955104112625122, 0.01178237795829773,
[('0.279888', 'yes'), ('0.177015', 'agree'), ('0.079209', 'no'), ('0.040416', 'applause'), ('0.033239', 'ok'), ('0.032701', 'eye_roll')]
['yes', 'agree', 'no', 'applause', 'ok', 'eye_roll']
------------------------------------------------------------------------------------------------------------------------
[0.07347774505615234, 0.12711471319198608, 0.030222177505493164, 0.0800132155418396, 0.010818302631378174, 0.001774519681930542, 0.0014872252941131592,
[('0.320326', 'idk'), ('0.274703', 'oops'), ('0.264315', 'shrug'), ('0.154736', 'sorry'), ('0.130699', 'scared'), ('0.081891', 'deal_with_it')]
['idk', 'oops', 'shrug', 'sorry', 'scared', 'deal_with_it']
------------------------------------------------------------------------------------------------------------------------
```

Figure 4: Prediction Result



#	User	Entries	Date of Last Entry	Team Name	P (all) ▲	P1 (GIF w/ text) ▲	P2 (GIF only) ▲
1	[redacted]	1	06/29/20	team mojitok	0.6255 (1)	0.6169 (1)	0.6313 (1)
2	[redacted]	1	06/28/20	Team_Monkey	0.5824 (2)	0.5580 (2)	0.5986 (2)
3	[redacted]	3	06/30/20	Team_Yankee	0.5662 (3)	0.5282 (4)	0.5915 (3)
4	[redacted]	1	06/30/20	Team_Lima	0.5437 (4)	0.5097 (6)	0.5663 (4)
5	[redacted]	1	06/29/20	IITP-AINLPML	0.5380 (5)	0.5006 (7)	0.5629 (5)
6	[redacted]	1	06/29/20	Team Oscar	0.5373 (6)	0.5151 (5)	0.5521 (6)
7	[redacted]	1	06/28/20	NCTU_Team_Delta	0.5345 (7)	0.5314 (3)	0.5365 (7)
8	tsanzxc456	1	06/30/20		0.4962 (8)	0.4542 (8)	0.5243 (8)

Figure 5: Score

5 Conclusion and Discussion

As shows in Figure.5, we can tell that our model only perform in averages. The main reason may be that our data pre-processing is not as consummate as other teams, such as processing for emoji or the tokenize step. In the future, we will try to add some more data pre-processing tricks to improve our model.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.