DM_kaggle_Report

Class	
Created	@Dec 7, 2020 9:19 PM
Type	
課程網頁	

Submission score

Submission	Submitted	score
result_1.csv	@Nov 25, 2020	0.43066
result_2.csv	@Nov 26, 2020	0.43395
IMDB_result_1.csv	@Dec 2, 2020	0.5569
bert_result_1.csv	@Dec 3, 2020	0.53266
IMDB_result_2.csv	@Dec 3, 2020	0.5569
IMDB_result_3.csv	@Dec 4, 2020	0.5569
IMDB_result_4.csv	@Dec 4, 2020	0.5569
IMDB_triangular.csv	@Dec 8, 2020	0.055295

1. Data preparation and observation 🔎

1. I use Pandas DataFrame to maintain the data, including 'tweet_id', 'text', 'emotion' and 'ident' columns. I deleted the '_index', '_crawldate' and '_type' columns because they're all the same (I mean the text in every row of each column are the same), and in this model I did not use 'hashtags' attribute, I only use the 'text' attribute.
2. After some observations, I found that the '_score' column did not help much because all of the scores are between 0 to 1024 and the average score is about 500 in every emotion category.
3. I also use the code below to know that the count of every emotion (I've also checked if there's duplicated or missing value in the data). As the result shows, most of the posts are 'joy', 'anticipation' or 'trust' (total 66%) so maybe if we let all of the test emotion to be one of 'joy', 'anticipation' and 'trust', the accuracy may be ok, too. (I did not try this method this time though.)

```
train_df = X.query("ident=='train'")
train_df['emotion'].value_counts()

output:
        joy             516017
        anticipation    248935
        trust           205478
        sadness         193437
        disgust         139101
        fear             63999
        surprise         48729
        anger            39867
        Name: emotion, dtype: int64
```

2. Models I've tried

1. First, I use the model taught in class(Keras) to be my deep learning framework, and follow the Model (functional API) to build a Deep Neural Network (DNN) model. Because this is maybe the easiest way I know, besides I can review what I learned on class.
   After training (I split the training set to 2 part: train, validation set (8:1) , epoch:15) and making prediction, I tried my first submission and got the score: 0.43066 , then I add one more hidden layer and split the training and testing set to

9:1 which let me got an moderate improvement. (I use TFIDF instead of bag-of-words)

2. Then I try to use ktrain to build the model that can do sentiment classification task (ktrain is a library to help build, train, debug, and deploy neural networks in the deep learning software framework, Keras.). With ktrain, I can employ fast and easy to use pre-canned models(BERT) for text classification. (See the code below for more details about training setting.) And I got a great improvement which make me take the first place at that time.(0.55690). I've tried setting the learning rate 2e-5 or 3e-5 and epochs from 2 to 3 but does not see any improvement.

3. I'm not satisfied with that score so I tried to use BERT directly to do text-classification. For simplicity, I changed the label in 'run_classifier.py' from [0, 1] to [0, 1, 2, 3, 4, 5, 6, 7] rather than ['joy', 'anger', ...] because I'm not doing binary classification like positive or negative (See the 'emotion_dict' below for the realation between integer 0~7 and emotion). I also convert the dataset into cola format (There are 4 classes that can be used for sequence classification tasks and I use Cola) which can be read by 'run_classifier.py' as input.

```
emotion_dict = { 'anger' : 0,
                 'anticipation' : 1,
                 'disgust' : 2,
                 'fear' : 3,
                 'sadness' : 4,
                 'surprise' : 5,
                 'trust' : 6,
                 'joy' : 7 }
```

DataFrame format and three input files for BERT :

1. train.tsv (no header)
2. dev.tsv (evaluation, no header)
3. test.tsv (with header)

train.tsv and dev.tsv format (guid, label, text_b, text_a, see here for more input format information):

0x376b20 1 a "People who post ""add me on #Snapchat"" must be dehydrated. Cuz man.... that's <LH>"
0x2d5350 4 a @brianklaas As we see, Trump is dangerous to #freepress around the world. What a <LH> <LH> #TrumpLegacy. #CNN
0x1cd5b0 3 a Now ISSA is stalking Tasha 😂😂😂 <LH>

test.tsv format:

tweet_id text
0x28b412 Confident of your obedience, I write to you, knowing that you will do even more than I ask. (Philemon 1:21) 3/4 #bibleverse <LH> <LH>
0x2de201 """Trust is not the same as faith. A friend is someone you trust. Putting faith in anyone is a mistake."" ~ Christopher Hitchens <LH> <LH>"

Fine-tuned (training and predicting) :

```
CUDA_VISIBLE_DEVICES=0 python3 run_classifier.py
--task_name=cola --do_train=true --do_eval=true --data_dir=dataset
--vocab_file=model/vocab.txt --bert_config_file=model/bert_config.json
--init_checkpoint=model/bert_model.ckpt --max_seq_length=64
--train_batch_size=64 --learning_rate=2e-5 --num_train_epochs=2.0
--output_dir=bert_output/ --do_lower_case=False --save_checkpoints_steps
3000

CUDA_VISIBLE_DEVICES=0 python run_classifier.py --task_name=cola
--do_predict=true --data_dir=./dataset --vocab_file=model/vocab.txt
--bert_config_file=model/bert_config.json --
init_checkpoint=bert_output/model.ckpt-40937
--max_seq_length=64 --output_dir=./bert_output/
```