CS 579 - Online Social Network Analysis

Spring 2022

Project II

Team members : *Reda CHAGUER (A20497223)*

Ismail ELOMARI ALAOUI (A20497221)

Supervisor : *Dr. Kai SHU*

04/27/2022

Introduction

The task of this project is fake news classification using Machine Learning algorithms: Given the title of a fake news article A and the title of a coming news article B, the task is to classify B into one of the three categories:

- **agreed**: B talks about the same fake news as A.
- **disagreed**: B refutes the fake news in A.
- **unrelated**: B is unrelated to A.

1 DataSet

The figure 1 shows the dataset that we used for this project. Each row is composed of two texts, A and B, their ids, and the label indicating the relationship between them.

	id	tid1	tid2	title1_en	title2_en	label
0	195611	0	1	There are two new old-age insurance benefits f...	Police disprove "bird's nest congress each per...	unrelated
1	191474	2	3	"If you do not come to Shenzhen, sooner or lat...	Shenzhen's GDP outstrips Hong Kong? Shenzhen S...	unrelated
2	25300	2	4	"If you do not come to Shenzhen, sooner or lat...	The GDP overtopped Hong Kong? Shenzhen clarifi...	unrelated
3	123757	2	8	"If you do not come to Shenzhen, sooner or lat...	Shenzhen's GDP overtakes Hong Kong? Bureau of ...	unrelated
4	141761	2	11	"If you do not come to Shenzhen, sooner or lat...	Shenzhen's GDP outpaces Hong Kong? Defending R...	unrelated

Figure 1: Dataset samples

As we can see in 2 figure that the dataset is not evenly distributed, which makes the classification slightly harder.

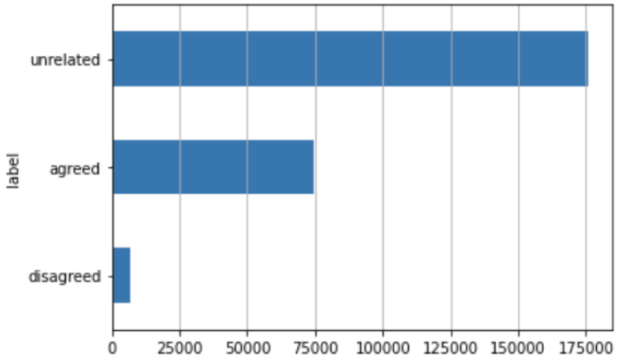


Figure 2: Label distribution

The dataset was split to training and validation set.

2 Data Processing

Before we can feed the dataset to our model, we need to perform some preprocessing, so that we can have more representative features and adapt to our model input.
With pretrained BERT tokenizer's `batch_encode_plus` function, we created batchs of both the sentences encoded together and separated by [SEP] token.
We got the following features :

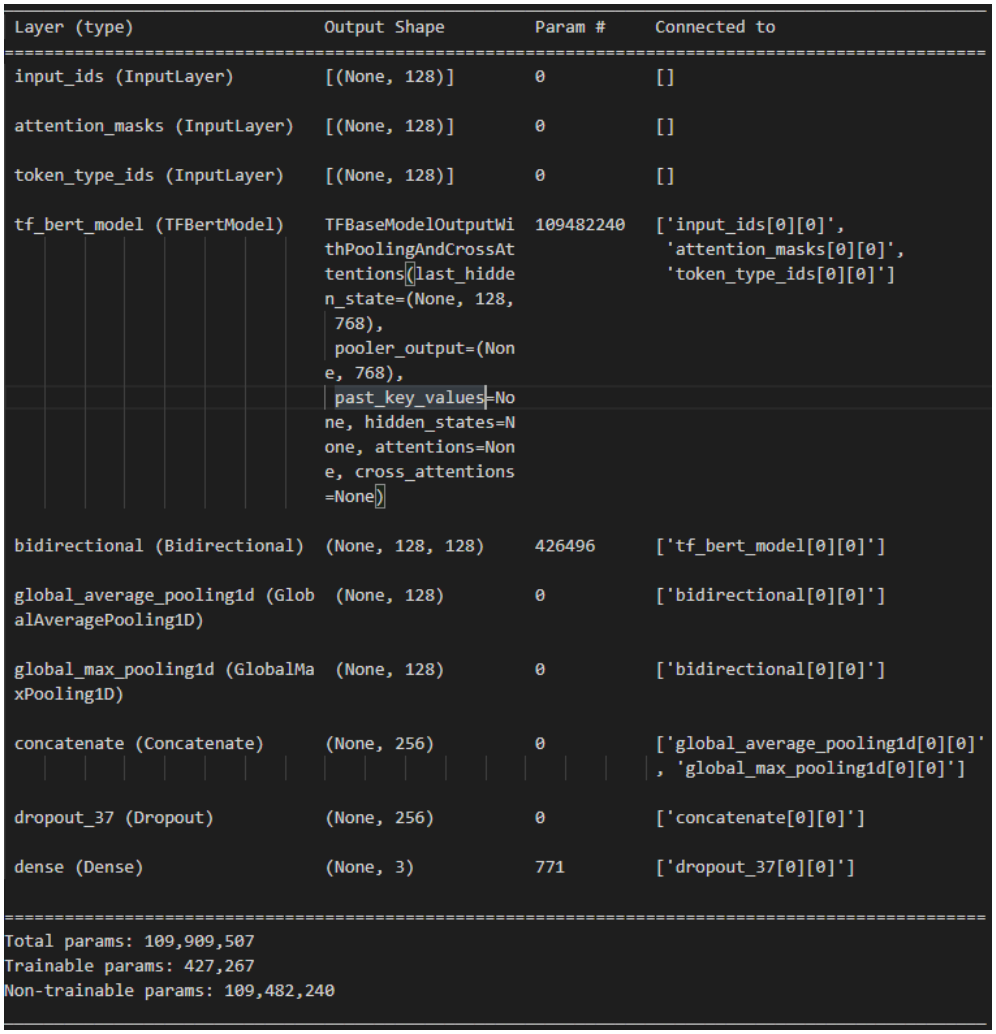
- Encoded token ids from BERT tokenizer.
- Attention masks indicates to the model which tokens should be attended to.
- Token type ids are binary masks identifying different sequences in the model.

For labels, One hot encoding technique was used.

3 Model Architecture

In more details, as shown in figure 3, the model was based on a pretrained BERT followed with a bidirectional LSTM (Long Short Term Memory), then applying a hybrid pooling approach before getting it into a neural network.

BERT stands for Bidirectional Encoder Representations from Transformers and it is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. We opted to train using frozen weights from the pretrained BERT and a max length input size of 128 for our model.



Layer (type)	Output Shape	Param #	Connected to
input_ids (InputLayer)	[(None, 128)]	0	[]
attention_masks (InputLayer)	[(None, 128)]	0	[]
token_type_ids (InputLayer)	[(None, 128)]	0	[]
tf_bert_model (TFBertModel)	TFBaseModelOutputWithPoolingAndCrossAttentions(last_hidden_state=(None, 128, 768), pooler_output=(None, 768), past_key_values=None, hidden_states=None, attentions=None, cross_attentions=None)	109482240	['input_ids[0][0]', 'attention_masks[0][0]', 'token_type_ids[0][0]']
bidirectional (Bidirectional)	(None, 128, 128)	426496	['tf_bert_model[0][0]']
global_average_pooling1d (GlobalAveragePooling1D)	(None, 128)	0	['bidirectional[0][0]']
global_max_pooling1d (GlobalMaxPooling1D)	(None, 128)	0	['bidirectional[0][0]']
concatenate (Concatenate)	(None, 256)	0	['global_average_pooling1d[0][0]', 'global_max_pooling1d[0][0]']
dropout_37 (Dropout)	(None, 256)	0	['concatenate[0][0]']
dense (Dense)	(None, 3)	771	['dropout_37[0][0]']

Total params: 109,909,507
Trainable params: 427,267
Non-trainable params: 109,482,240

Figure 3: Model Architecture

We added a dropout layer to minimize the model overfitting.

For the loss, we used the cross entropy loss which is mostly used for the classification tasks and Adam optimizer with leaning rate of 0.0001.

4 Training

We trained the model with Kaggle using a Tesla GPU each time for around 6 hours and batch size of 512. At the end, the best validation accuracy that the model achieved was around 86%.

```
Epoch 10/10
400/400 [==============================] - 1151s 3s/step - loss: 0.2878 - acc: 0.8767 - val_loss: 0.3209 - val_acc: 0.8626
```

As shown in Figure 4, the general shape of the loss and the accuracy during the training

showed that the network was learning.

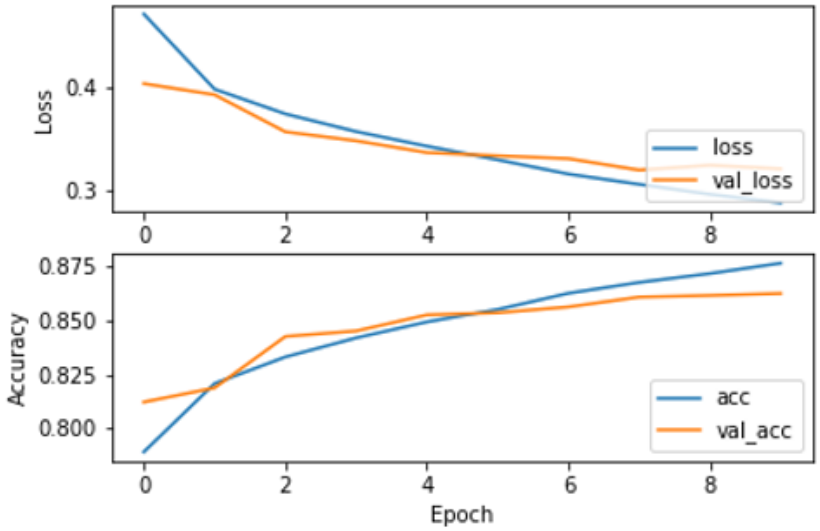


Figure 4: Training and Validation Graph

Team Efforts

We managed to split the work onto multiple tasks. Each member worked on one task and we shared our results at the end to test and verify the compatibility and integrity of our project.

Conclusion

This project was a great initiation to online social network analysis. We got to practice and brush up on some notions studied during the lectures. We also visited the world of data mining in general and precisely detecting fake news.

References

BERT: https://arxiv.org/abs/1810.04805

Keras: https://keras.io/api/

LSTM: https://machinelearningmastery.com/develop-bidirectional-lstm-sequence-classifi

Kaggle: https://www.kaggle.com/