EMOTION DETECTION FROM TEXT

Author : RAHUL MALLAH

Mail ID : rahulmallah785671@gmail.com

University Name : NATIONAL INSTITUTE OF TECHNOLOGY SILCHAR

INTRODUCTION

The recent century has brought to humanity many innovations, which expanded the way we interact within society and with the universe. Many of these innovations are double edged swords, like the discovery of modern explosives by Albert Nobel, which allowed quicker extraction of priceless resources below the surface of Earth. In this paper, we investigate a different-faced aspect of information posted on Twitter. We aim to investigate a way how we can effectively categorize tweets. Natural language processing (NLP) is a branch of computer science and linguistics that focuses on the analysis and processing of human language. It deals with natural language, a human-generated data highly distinguished from other kinds of data like images. The challenging part of NLP is the ambiguity and complexity of human language. With the popularization of social media, the amount of content grows corresponding to their popularity.

In the era of a customer-focused industry, companies are coming up with new ways to understand their consumers. Detecting emotions accurately from the reviews, chats, tweets, blogs, posts, etc. is one such method without explicitly asking the customers. With the advent of new algorithms and increasing computing power, Natural Language Processing (NLP) has enabled us to detect emotions from written text & take action accordingly.

'Emotion detection' is one level deeper than the typical sentiment classification problem in the NLP world. In sentiment analysis, polarity (positive, negative, or neutral) is the primary concern, whereas, in emotion detection, the emotional or psychological state is detected. Sentiment analysis is highly subjective, whereas emotion detection is more objective and precise.

METHODOLOGY .

What is BERT?

We've heard about BERT and how incredible it is, and how it's potentially changing the NLP landscape. But what is BERT in the first place?

Here's how the research team behind BERT describes the NLP framework:

"BERT stands for Bidirectional Encoder Representations from Transformers. It is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of NLP tasks."

That sounds way too complex as a starting point. But it does summarize what BERT does pretty well so let's break it down.

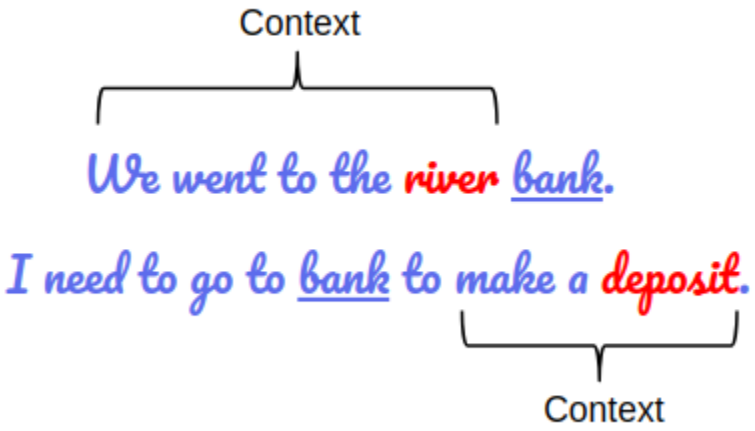
First, it's easy to see that BERT stands for Bidirectional Encoder Representations from Transformers For now, the key takeaway from this line is – BERT is based on the Transformer architecture.

Second, BERT is pre-trained on a large corpus of unlabelled text including the entire Wikipedia(that's 2,500 million words!) and Book Corpus (800 million words).

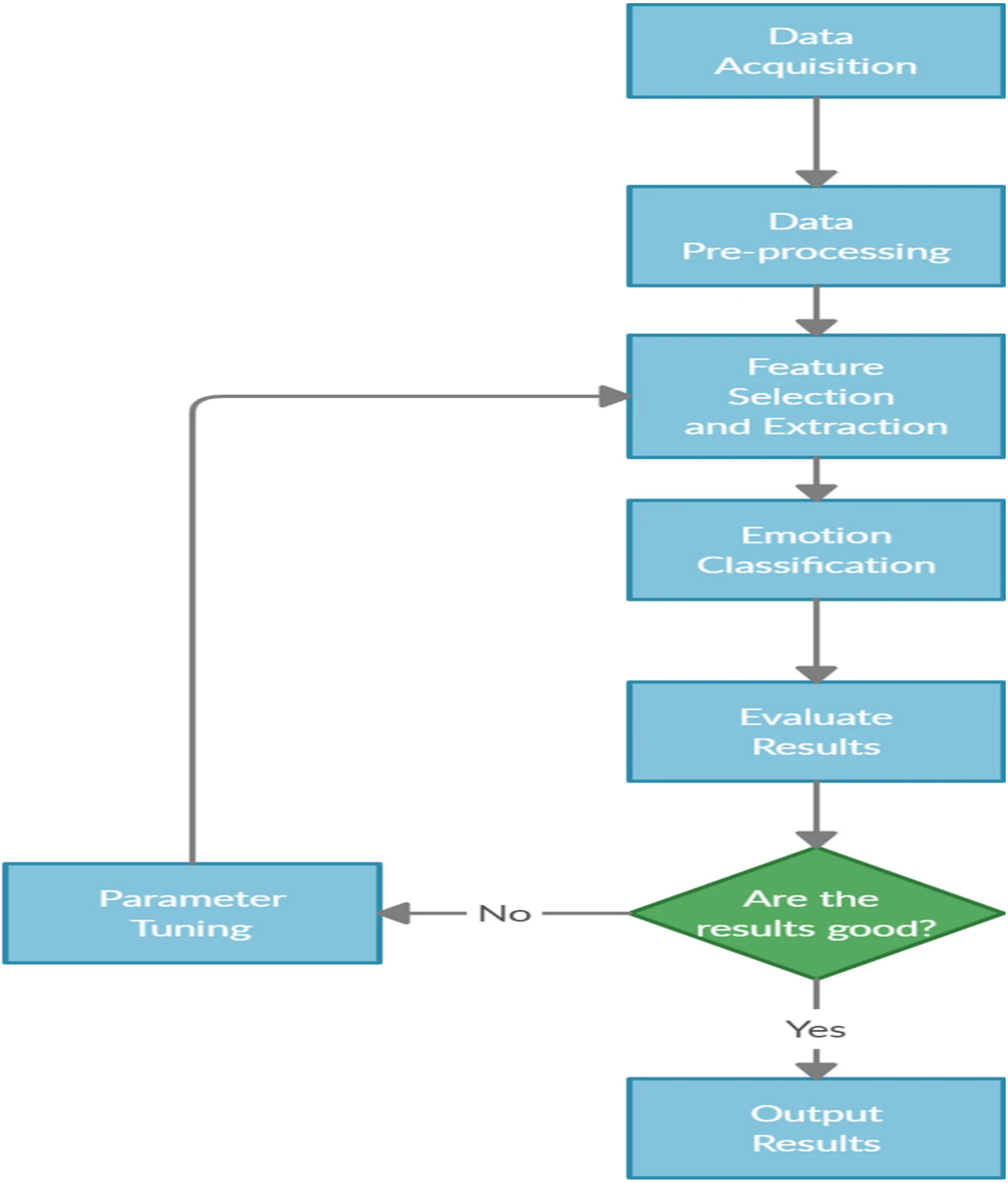
This pre-training step is half the magic behind BERT's success. This is because as we train a model on a large text corpus, our model starts to pick up the deeper and intimate understandings of how the language works. This knowledge is the swiss army knife that is useful for almost any NLP task.

Third, BERT is a “deeply bidirectional” model. Bidirectional means that BERT learns information from both the left and the right side of a token’s context during the training phase.

The bidirectionality of a model is important for truly understanding the meaning of a language. Let’s see an example to illustrate this. There are two sentences in this example and both of them involve the word “bank”:



Bert captures both right and left contexts



Simple emotion-detection process

IMPLEMENTATION

Libraries Used:

We have implemented the pre-trained BERT model in Python for prediction. Following is the list of libraries that have assisted us in the code-

```
import pandas as pd
import numpy as np
import text_hammer as th
from tqdm._tqdm_notebook import tqdm_notebook
tqdm_notebook.pandas()
from sklearn.model_selection import train_test_split
from transformers import AutoTokenizer,TFBertModel
from transformers import BertTokenizer, TFBertModel, BertConfig,TFDistilBertModel,DistilBertTokenizer,DistilBertConfig
import shutil
import tensorflow as tf
from tensorflow.keras.layers import Input, Dense
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import EarlyStopping
from tensorflow.keras.initializers import TruncatedNormal
from tensorflow.keras.losses import CategoricalCrossentropy
from tensorflow.keras.metrics import CategoricalAccuracy
from tensorflow.keras.utils import to_categorical
from sklearn.metrics import classification_report,confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
%config Completer.use_jedi = False # if autocompletion doesnot work in kaggle notebook | hit tab
```

Pandas : Pandas is used to read a csv and strongly used in data wrangling.
Numpy : Numpy is a mathematical computational library
Text_hammer : Text hammer is used for text preprocessing
Tqdm : Tqdm creates hassle-free progress bars
Scikit Learn : It provides selection of efficient tools for Machine Learning/ Deep learning/ NLP
Transformers : Used to load pre-trained bert model

Shutil : Used to save the model for future use
#Tensorflow : Deep learning library
Matplotlib/ Seaborn : To plot data

Loading pre-trained bert model:

[18]:
```
tokenizer = AutoTokenizer.from_pretrained('bert-base-cased')
bert = TFBertModel.from_pretrained('bert-base-cased')
```

```
Downloading: 100% 29.0/29.0 [00:00<00:00, 1.00kB/s]
Downloading: 100% 570/570 [00:00<00:00, 17.0kB/s]
Downloading: 100% 208k/208k [00:00<00:00, 338kB/s]
Downloading: 100% 426k/426k [00:00<00:00, 536kB/s]
Downloading: 100% 502M/502M [00:11<00:00, 40.1MB/s]
```

Building model:

[24]:
```
max_len = 70

input_ids = Input(shape=(max_len,), dtype=tf.int32, name="input_ids")
input_mask = Input(shape=(max_len,), dtype=tf.int32, name="attention_mask")

embeddings = bert(input_ids,attention_mask = input_mask)[0] #(0 is the last hidden states,1 means pooler_output)
out = tf.keras.layers.GlobalMaxPool1D()(embeddings)
out = Dense(128, activation='relu')(out)
out = tf.keras.layers.Dropout(0.1)(out)
out = Dense(32,activation = 'relu')(out)

y = Dense(6,activation = 'sigmoid')(out)

model = tf.keras.Model(inputs=[input_ids, input_mask], outputs=y)
model.layers[2].trainable = True
```

Model architecture:

[26]:
```
model.summary()
```

```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
input_ids (InputLayer)       [(None, 70)]              0         
_________________________________________________________________
attention_mask (InputLayer)  [(None, 70)]              0         
_________________________________________________________________
tf_bert_model (TFBertModel)  TFBaseModelOutputWit 108310272   
_________________________________________________________________
global_max_pooling1d (GlobalMax (None, 768)            0         
_________________________________________________________________
dense (Dense)                (None, 128)               98432     
_________________________________________________________________
dropout_37 (Dropout)         (None, 128)               0         
_________________________________________________________________
dense_1 (Dense)              (None, 32)                4128      
_________________________________________________________________
dense_2 (Dense)              (None, 6)                 198       
=================================================================
Total params: 108,413,030
Trainable params: 108,413,030
```

4. Model fitting and then evaluation

```
train_history = model.fit(
    x ={'input_ids':x_train['input_ids'],'attention_mask':x_train['attention_mask']} ,
    y = to_categorical(data_train['label']),
    validation_data = (
    {'input_ids':x_test['input_ids'],'attention_mask':x_test['attention_mask']}, to_categorical(data_test['label'])
    ),
  epochs=1,
    batch_size=36
)
```

```
2022-09-07 15:30:25.613943: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:185] None of the MLIR Optimization Passes are enabled (re
 76/389 [====>.....................] - ETA: 1:19:34 - loss: 1.3852 - balanced_accuracy: 0.4894
```

+ Code + Markdown

RESULT/OUTPUT

Balanced accuracy score and classification report:

```
accuracy_score(data_test['label'],y_predicted)
```

```
[43]: 0.9275
```

+ Code + Markdown

```
[33]: print(classification_report(data_test['label'], y_predicted))
```

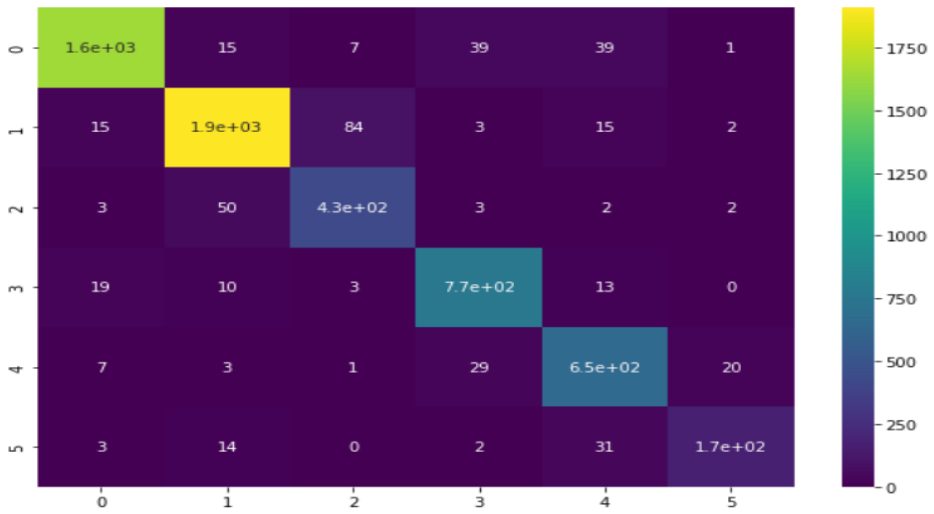
```
              precision    recall  f1-score   support

           0       0.97      0.94      0.96      1739
           1       0.95      0.94      0.95      2028
           2       0.82      0.88      0.85       492
           3       0.91      0.94      0.93       813
           4       0.87      0.92      0.89       712
           5       0.87      0.77      0.82       216

    accuracy                           0.93      6000
   macro avg       0.90      0.90      0.90      6000
weighted avg       0.93      0.93      0.93      6000
```

We have got an accuracy score of approx. 93% and presion, recall, f1_score are shown above.

Heatmap:



We have also drawn the heatmap of confusion_matrix for all the emotions i.e. anger(0), fear(1), joy(2), love(3), sadness(4) and surprise(5).

Results:

1. Average Precision : 93%
2. Average Recall : 93%
3. Average F1_score : 93%

CONCLUSION

The potential for further investigation is high considering the number of comments not only on Twitter but also on other social media platforms. Knowing the correct emotions of a person is not only for marketing and gaining more and more profits it's the role that the internet, particularly social media, might have in suicide-related behaviour is a topic of growing interest and debate. Through Machine learning/ Deep learning/ NLP there is a possibility to detect emotions of a person, detect and stop them from such activities. Regarding emotion analysis, further research is especially desirable in the evaluation of various services and products.

Furthermore, the imperfect inter-annotator score did not hinder our results, as we never used manually annotated data. However, we rely solely on the quality of the provided labels. Thus, it is hard to imagine it would be possible to create such a piece of software which would reliably annotate data with 100% accuracy. But, It's not impossible.

Reference websites

1. https://huggingface.co/bert-base-uncased
2. https://www.tensorflow.org/resources/libraries-extensions
3. https://scikit-learn.org/stable/
4. What is BERT | BERT For Text Classification (analyticsvidhya.com)
5. https://link.springer.com/article/10.1007/s10462-021-09958-2
6. https://file.techscience.com/ueditor/files/cmc/TSP_CMC-71-2/TSP_CMC_21671/TSP_CMC_21671.pdf