COVID-19 Fake News Detection

Disaster Management Project

Team 1:

Ruthvik Kodati (2019101035)

Avlok Gupta (2018111017)

Arshad Mohammed (2018900056)

Division of Work:

1) Ruthvik - Worked on the fake news model, made a few suggestions to the web page, and made the report

2) Avlok - Worked on the fake news model, made a few suggestions to the web page, and made the report

3) Arshad - Worked on the web page and made suggestions for the fake news modelling

Introduction:

There has been an increasing interest in fake news on social media. Detecting misinformation on social media is important and challenging. Since humans cannot accurately distinguish false from true news mainly because it involves tedious evidence collection as well as careful fact-checking, this poses a difficulty. With the advent of technology and the ever-increasing propagation of fake articles on social media, it has become really essential to come up with automated frameworks for fake news identification. In this paper, we describe our system which performs a binary classification on tweets from social media and assigns a *score* to them that helps us identify whether they are “real” or “fake”. We have also used fine-tuning in our approach as it has proven to be extremely effective in text classification tasks.

A web page was made as a project for our Disaster Management course and helps the public distinguish between rumours and facts regarding the coronavirus pandemic. We acquired a dataset of COVID-19 news and trained a model which assists in detecting whether the data is false or not. The page includes a search bar in which a statement can be entered and a score is returned that delineates whether it is more likely to be true/false. Moreover, it also includes a few examples of facts and rumours presented in a table.

Background:

In fake news identification problems, traditional machine learning approaches have been quite successful. The problem was then approached as a binary classification problem where these features were fed into conventional Machine Learning classifiers like K-Nearest Neighbor (KNN), Random Forest (RF), etc. which yielded results that were quite favourable [1]. Most of the current state-of-the-art language models are based on Transformers and they have proven to be highly effective in text classification problems. They provide superior results when compared to previous state-of-the-art approaches using techniques [1]. The introduction of the BERT architecture has transformed the capability of transfer learning in Natural Language Processing. It has been able to achieve state-of-the-art results on downstream tasks like *text classification*.

Our project utilised *TensorFlow* and the *BERT Language Model* to create an accurate model which detects whether the news is fake or not.

Materials and Methodology:

Dataset Information -

The dataset used in our project was collected from various social media and fact-checking websites. The “real” news items were collected from verified sources that give useful information about COVID-19, while the “fake” ones were collected from tweets, posts and articles which make speculations about COVID-19 that are verified to be false.

Dataset link - https://github.com/diptamath/covid_fake_news/tree/main/data

This dataset was obtained from a competition that was focused on detecting fake news in English.

Competition link - https://competitions.codalab.org/competitions/26655

Methodology -

We have approached this task as a text classification problem. Each news item needs to
be classified into two distinct categories: “real” or “fake”. Our proposed method consists of four main parts: (a) Understanding the data, (b) Text Preprocessing, (c) Training the Classification Model, and (d) Evaluation.

Understanding the Data -

The data from the dataset link is passed and classified into “real” or “fake” data.

Positive Samples (Real Data):

1. The CDC currently reports 99031 deaths. In general the discrepancies in death counts between different sources are small and explicable. The death toll stands at roughly 100000 people today.
2. States reported 1121 deaths a small rise from last Tuesday. Southern states reported 640 of those deaths.

https://t.co/YASGRTT4ux

3. #IndiaFightsCorona: We have 1524 #COVID testing laboratories in India and as on 25th August 2020 36827520 tests have been done : @ProfBhargava DG @ICMRDELHI #StaySafe #IndiaWillWin

https://t.co/Yh3ZxknnhZ

4.Populous states can generate large case counts but if you look at the new cases per million today 9 smaller states are showing more cases per million than California or Texas: AL AR ID KS KY LA MS NV and SC.

https://t.co/1pYW6cWRaS

5.Covid Act Now found "on average each person in Illinois with COVID-19 is infecting 1.11 other people. Data shows that the infection growth rate has declined over time this factors in the stay-at-home order and other restrictions put in place."

https://t.co/hhigDd24fE

Negative Samples (Fake Data):

1. Politically Correct Woman (Almost) Uses Pandemic as Excuse Not to Reuse Plastic Bag https://t.co/thF8GuNFPe #coronavirus #nashville
2. Obama Calls Trump’s Coronavirus Response A Chaotic Disaster https://t.co/DeDqZEhAsB
3. ???Clearly, the Obama administration did not leave any kind of game plan for something like this.??
4. Retraction—Hydroxychloroquine or chloroquine with or without a macrolide for treatment of COVID-19: a multinational registry analysis - The Lancet https://t.co/L5V2x6G9or
5. The NBA is poised to restart this month. In March we reported on how the Utah Jazz got 58 coronavirus tests in a matter of hours at a time when U.S. testing was sluggish. https://t.co/I8YjjrNoTh https://t.co/o0Nk6gpyos

The graph below depicts the training data distribution:

It is class-wise balanced with 52.34% of the samples consisting of real news, and 47.66% of fake samples.

Data Preprocessing -

Some social media items are mostly written in informal language. Also, they contain various other information like usernames, URLs, emojis, etc. [1] We have filtered out such attributes (ex. retweets, links, extra spaces, emoticons, symbols, etc.) from the given data as a basic preprocessing step, before training the classification model. We have used *regex* to filter out such noisy information from the data. Text inputs need to be transformed to numeric token IDs and arranged in several Tensors before being input to the BERT model. TensorFlow Hub provides a matching preprocessing model for the BERT model [3].

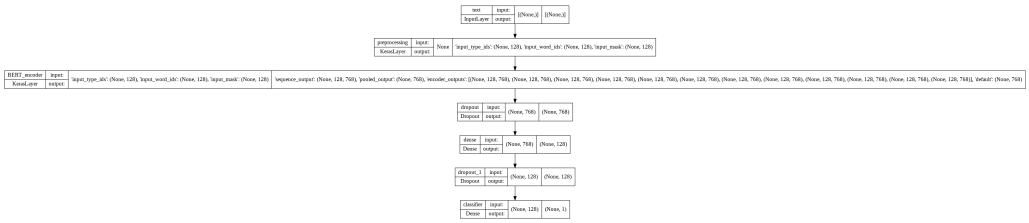
Training Classification Model -

We utilised the BERT Language Model for this project. It was implemented as shown in the code below:

```
def build_classifier_model():
  text_input = tf.keras.layers.Input(shape=(), dtype=tf.string, name='text')
  preprocessing_layer = hub.KerasLayer("https://tfhub.dev/tensorflow/bert_en_uncased_preprocess/3", name='preprocessing')
  encoder_inputs = preprocessing_layer(text_input)
  encoder = hub.KerasLayer("https://tfhub.dev/tensorflow/bert_en_uncased_L-12_H-768_A-12/3", trainable=True, name='BERT_encoder')
  outputs = encoder(encoder_inputs)
  dense_input = tf.keras.layers.Dropout(0.1)(outputs['pooled_output'])
  dense_output = tf.keras.layers.Dense(128, activation='relu', name='dense')(dense_input)
  classifier_input = tf.keras.layers.Dropout(0.1)(dense_output)
  classifier_output = tf.keras.layers.Dense(1, activation=None, name='classifier')(classifier_input)
  return tf.keras.Model(text_input, classifier_output)

covid_fake_news_model = build_classifier_model()

epochs = 5
step_size = 32
steps_per_epoch = len(train_df)/step_size
num_train_steps = steps_per_epoch * epochs
num_warmup_steps = int(0.1*num_train_steps)

init_lr = 3e-5
optimizer = nlp.optimization.create_optimizer(init_lr=init_lr,
                                          num_train_steps=num_train_steps,
                                          num_warmup_steps=num_warmup_steps,
                                          optimizer_type='adamw')

covid_fake_news_model.compile(optimizer=optimizer, loss=tf.keras.losses.BinaryCrossentropy(from_logits=True), metrics=tf.metrics.BinaryAccu
```

The dense layer is just a regular layer of neurons in a neural network. Each neuron receives input from all the neurons in the previous layer, thus densely connected.

The dropout layer is used to tackle *overfitting*. The Dropout method in keras.layers module takes in a float between 0 and 1, which is the fraction of the neurons to drop.



As seen in the model's structure, after passing through the various layers, we obtain the classifier which is significant in training the model.

Since this is a binary classification problem and the model outputs a probability (a single-unit layer), the `losses.BinaryCrossentropy` loss function was used. The loss function tells how good the model is in predictions. If the model predictions are closer to the actual values the loss will be minimum and if the predictions are totally away from the original values the loss value will be the maximum [3].

For fine-tuning, we used the same optimizer that BERT was originally trained with: the "Adaptive Moments" (Adam). This optimizer minimizes the prediction loss and does regularization by weight decay (not using moments), which is also known as AdamW [3].

Why we used the BERT Model -

The use of the bidirectional training of Transformer, a popular attention model, to language modelling was seen BERT's fundamental technical breakthrough. In contrast to earlier research, which looked at a text sequence from left to right or a combination of left-to-right and right-to-left training, this study looked at a text sequence from left to right. **Bidirectionally trained language models can have a better grasp of language context and flow than single-direction language models.** The Transformer encoder reads the complete sequence of words at once, unlike directional models that read the text input sequentially (left-to-right or right-to-left).

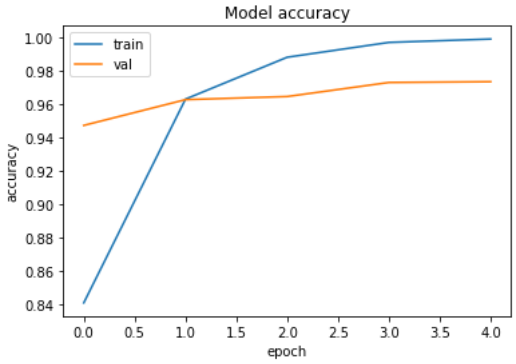
The BERT model which we used - https://tfhub.dev/tensorflow/bert_en_uncased_L-12_H-768_A-12/4

Evaluation -

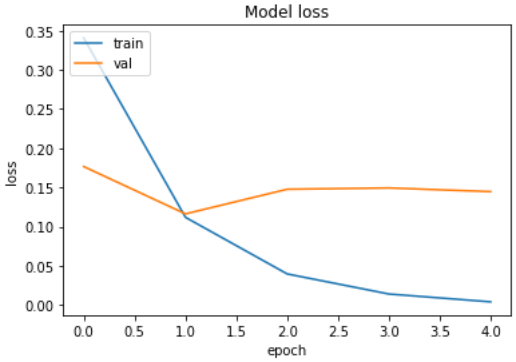
After each epoch, it is evident that the binary accuracy for the training data set increased and the validation accuracy increased (validation data set is also present in the github dataset link above).

```
Epoch 1/5
201/201 [==============================] - 431s 2s/step - loss: 0.3407 - binary_accuracy: 0.8410 - val_loss: 0.1766 - val_binary_accuracy: 0.9472
Epoch 2/5
201/201 [==============================] - 407s 2s/step - loss: 0.1118 - binary_accuracy: 0.9629 - val_loss: 0.1162 - val_binary_accuracy: 0.9626
Epoch 3/5
201/201 [==============================] - 407s 2s/step - loss: 0.0396 - binary_accuracy: 0.9880 - val_loss: 0.1477 - val_binary_accuracy: 0.9645
Epoch 4/5
201/201 [==============================] - 407s 2s/step - loss: 0.0139 - binary_accuracy: 0.9969 - val_loss: 0.1493 - val_binary_accuracy: 0.9729
Epoch 5/5
201/201 [==============================] - 409s 2s/step - loss: 0.0040 - binary_accuracy: 0.9989 - val_loss: 0.1446 - val_binary_accuracy: 0.9734
```

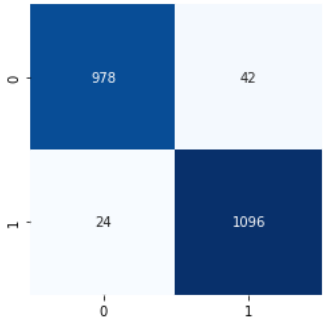
We ran the model on 5 epochs with the model accuracy increasing after every epoch. Our final accuracy on the validation data set was as high as 97.34% with the accuracy on training data being almost close to perfect.



We see the loss (a number which represents the error, lower values are better) increasing after one epoch which tells us that the model is slightly *overfitted*. However, this did not pose a problem for us as we were able to achieve very high accuracy on the validation set.



Results:



The *confusion matrix* shows us that our model predicts fake news with an accuracy of 95.8% (978 out of 1020 correct) and real news with an accuracy of 97.8% (1096 out of 1120 correct) on the test dataset. Overall the accuracy of our model is 96.91%.

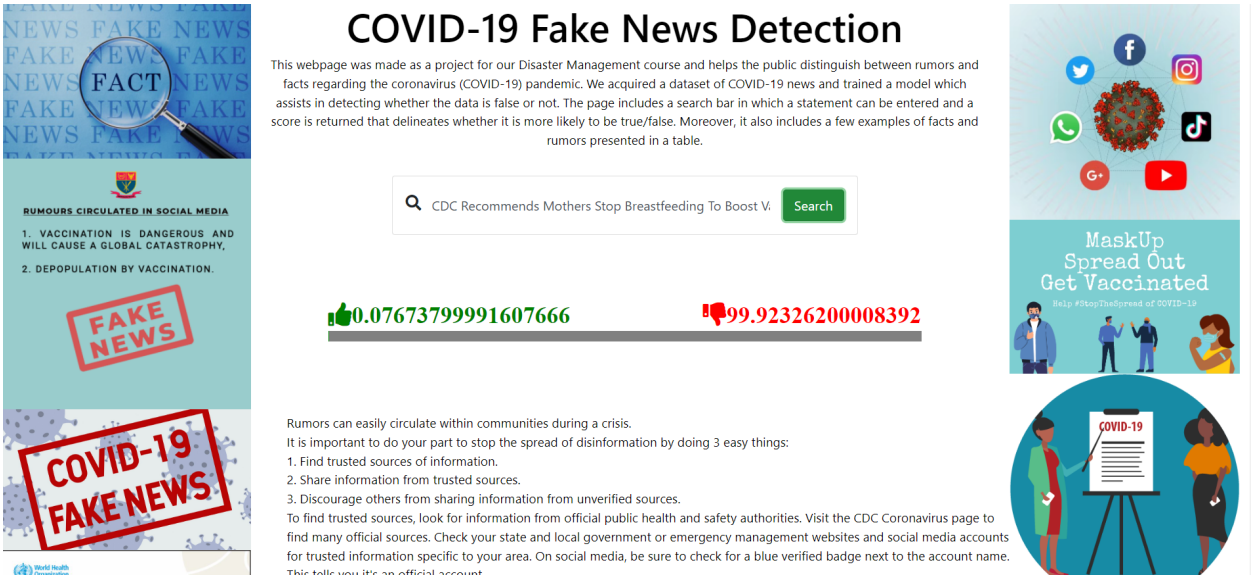
We obtained real data regarding COVID-19 and tested it on our web page:

Real data - Protect yourself and others from #COVID19 when using public transportation. Practice social distancing avoid touching surfaces and practice hand hygiene. Learn more: https://t.co/0vhHD4uFv9. https://t.co/D8YSeE3vXv

As shown on the web page, the data received a 85.76 percent positive score which means it is more likely to be real.

We obtained fake data regarding COVID-19 and tested it on our web page:

Fake data - CDC Recommends Mothers Stop Breastfeeding To Boost Vaccine Efficacy



As shown on the web page, the data received a 99.92 percent negative score which means it is more likely to be fake.

Conclusion:

Our goal was to train a model which could detect whether COVID-19 related news was real or fake. We utilised the BERT language model and have provided all of the materials and methodology in this report. The results delineate that our model was 96.91% accurate and we have seen successful results after testing the model with the testing dataset (included in the github dataset above). Through this project, we have created a model that could be extended for other disasters as well. We strictly adhered to the COVID-19 disaster because we wanted to focus on ensuring accurate results and were unsure if we could maintain that scope for multiple disasters. As a team, we learnt a lot about the significance of detecting fake data and how it should not be spread. We hope that our project serves as a good example for fake data detection and our ideas can be utilised for further ventures.

Github repository link to code and data - https://github.com/avlokgupta1313/Disaster-Management-Project

References:

[1] https://arxiv.org/pdf/2101.03545.pdf

[2] https://arxiv.org/pdf/1810.04805.pdf

[3] https://www.tensorflow.org/text/tutorials/classify_text_with_bert