Task 1: Rumour Detection

This task aims to detect misinformation on Twitter data. With social media being used widely misinformation detection is very essential to stop spreading rumours. Several attempts (Kaimin, et al., 2019) (Arkaitz, et al., 2018) have been made in the past to detect rumours in social media. In this paper, we have analysed the state of the art Recurrent Neural Network techniques like LSTM and BERT on textual data to detect rumours.

**1.1 The Dataset**

The dataset is sourced from the Twitter API and labelled to classify between rumour and non-rumour classes. The dataset is divided into training, development, and test sets to create the model. The training and development set are provided with labels whereas the test set labels are masked to evaluate the performance in the Codalab class competition. Each training instance consists of a source tweet along with comments and retweets. The tweets consist of various information including the text of the tweet, user information, geolocation, etc.

**1.2 The methodology**

Considering the data available in the scenario, the fact of a tweet being considered as misinformation can be derived mainly through the comments and retweets. As the main focus of this research is on NLP, textual data alone is considered in the following two way.

* Base tweet
* Base Tweet + Sub Tweets

Having said that, different pre-processing techniques and machine learning models have been administered for further evaluation.

**1.3 Pre-Processing**

Pre-processing step here involves the processing of textual information of the tweet. The text of the tweet needs to be converted into vectors so that the machine learning algorithms can directly take them as input. There are several techniques in which text can be handled and so the following text-processing methods are inspected as given below.

**1.3.1 Text Processing**

The text of the tweet is first lowered and stop words are removed from the document as these do not convey any real meaning to the tweet. Once this base processing is done, the tweets are then processed in one of the ways given below.

**1.3.1.1 Bag of Words**

The bag of words is one of the classic techniques used in text processing where the text is broken down into individual words and parsed into a matrix form using binary numbers. *Tokenizer.texts\_to\_matrix* method from the *Keras* package of *TensorFlow* library allows us to convert text into a *NumPy* matrix of binary numbers. *Tf-Idf* (Aizawa, 2003) mode is chosen to achieve this process as it calculates the importance of a word to a document. Using this technique enables to achieve the best training accuracy.

**1.3.1.2 Word Sequences**

Unlike the Bag of Words representation of text, the word sequences technique (Grobelnik, 1998) preserves the sequence of the words to provide more context for the classifier to predict. The order of words is taken into the context which enables RNN classifiers like LSTM and BERT to leverage this to better understand the training instances. *texts\_to\_sequences* method from the *Keras* package of the *TensorFlow* library allows us to convert a piece of text into continuous vector space. These sequences are then padded with trailing zero to normalise variable sentence length. *pad\_sequences* method from *keras.preprocessing* when supplied with the max length of the sentence enables us to achieve this feat.

**1.4 Machine Learning Models**

**1.4.1 Logistic Regression**

A model which considers just the base tweet and using classic logistic regression is considered as the baseline model. This can be used to compare more advanced algorithms which use different ways to incorporate all the subtweets in a training instance. The text data is converted into vectors using the bag of words technique as discussed above. The *LogisticRegression* model from SciPy package is then used with the best setting (solver="lbfgs", penalty="l2", C=1.0) found using cross-validation technique. This setting enables us to achieve a test F-Score of 0.83330.

**1.4.2 Feed-forward Neural Networks**

When dealing with a text classification problem, the sequence of text input gives a better context to machine learning algorithms. The feedforward neural network is set up with one intermediate layer having 10 neurons and a *Relu* activation function which has fewer vanishing gradient problems (Xavier, et al., 2011). The input dimension for this layer is set to the vocabulary size of the entire corpus. As it is a binary classification task the output layer is constructed with a single neuron with a sigmoid activation function which produces binary output. The entire modelling has been done using the *Sequential* method from *keras.models* using *Adam* optimizer with *binary\_crossentropy* loss function.

**1.4.3 LSTM**

The Long Short Term Memory Neural Networks is an advanced version of Recurrent Neural Networks that has a persistent memory to overcome the Vanishing Gradient Problem. The LSTM can be modelled to consider bidirectional input to consider words on either side. The LSTM model is designed with 64 bidirectional LSTM nodes in the intermediate layer and 1 dense layer in the output which incorporates sigmoid activation. This is again implemented using the *LSTM and Bidirectional* method from *keras.layers* package.

**1.4.4 BERT**

Diagram

Description automatically generatedThe Bidirectional Encoder Representations from Transformers is a pre-trained model which can be tuned to any NLP task. Bert uses a bidirectional model which enables it to learn from both left and right context of words. Adding a final output layer to this powerful pre-trained model produces a state of art NLP algorithm.

Figure 1

Owing to the high computing resources required by full-sized BERT, a lighter version of it called *DistilBERT* (Victor, et al., 2020), which has almost a similar performance is being used here. This is coupled up with a Logistic Regression model from *Scikit* to make the final prediction. As described in *figure 1*, the model takes in the sentence as input and pre-processes it before passing it onto the *DistilBERT* model. The pre-processing involves tokenization of the text and transforming them into vector embeddings using the Word2Vec technique. These vectors are then padded with special tokens and attention mask for the *DistilBERT* model to interpret. These pre-processed sentences are passed through *DistilBERT* and Logistic Regression to get the final prediction.

**1.5 Results and Conclusion**

|  |  |  |  |
| --- | --- | --- | --- |
|  | F-Score | Precision | Recall |
| LG1 | 0.8333 | 0.8721 | 0.7979 |
| FF-NN2 | 0.7485 | 0.8312 | 0.6809 |
| LSTM3 | 0.7952 | 0.7021 | 0.7458 |
| BERT4 | 0.7386 | 0.6915 | 0.7143 |

Table 1: Test Performance Results on Root + Sub Tweets

|  |  |  |
| --- | --- | --- |
|  | Root Tweet | Root + Sub Tweets |
| LG1 + BOW | 0.8879 | 0.8017 |
| FF-NN2 + BOG | 0.8690 | 0.8448 |
| FF-NN2 + Word Sequences | 0.8500 | 0.8672 |
| LSTM3 | 0.8276 | 0.8259 |
| BERT4 | 0.8664 | - |

Table 2: Comparison of different textual techniques

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*1 – Logistic Regression, 2 – Feed Forward Neural Network, 3 – Long Short-Term Memory, 4 - Bidirectional Encoder Representations from Transformers*

From *Table 1* and *Table 2*, it is evident that the Logistic Regression model with just the base tweet performs the best with a validation accuracy of 0.8879. This can be attributed to the fact that logistic regression models perform best with binary classification task. Moreover, the neural Text

Description automatically generatednetworks require a large amount of training data which is not available in this scenario.

It is to be noted that maximum accuracy was produced when the bag of words pre-processing technique and just the base tweet are taken into consideration. On the other hand, when sub tweet texts are considered using the word sequences technique, models like RNN which can leverage the contextual meaning produced the best accuracy than just using the base tweets. BERT which uses an advanced two-layer architecture produced considerably better accuracy than traditional RNN models.

**1.6 Future Directions**

Although the full version of BERT will theoretically produce better accuracy, hardware limitations hindered us from being implementing it. The metadata of the tweets such as user information, retweets and several other attributes can be utilized in tandem with textual data to create a better predicting model. In those hybrid model, the functional API architecture of the Keras library can be leveraged to take heterogeneous data as input.

Task 2: Rumour Analysis

The best model found in task 1 which is logistic regression using base tweets is used to detect rumours in COVID-19 tweets. The classified rumour and non-rumour tweets are taken down for further analysis to produce some useful insights.

**2.1 Topic Analysis**

Topic analysis has been performed based on the entire pre-processed text corpus of all the root tweets to form word clouds using the *wordcloud*.

**2.1.1 Unigram topics**

Text

Description automatically generatedFigure 3

Unigram topics for rumour and non-rumour tweets have been displayed in *Figure 2* and *Figure 3* respectively. The rumour tweets contain topics like “breaking”, “UK”, and “Death” which is not so popular in non-rumour tweets. This indicates that most of the rumours are revolving around breaking news claiming about deaths which was a common problem to tackle as per social media reports. Countries like the UK which had a new variant of the virus had more rumour tweets.

**2.1.2 Bigram topics**

**Text

Description automatically generated**

Figure 4

**Text

Description automatically generated**

Figure 5

The bigram counts also show that most of the topics are similar in rumour and non-rumour tweets. “Tested positive” bigram being a popular topic in rumoured tweets indicate that there were a lot of rumours about the wrong number of covid cases. Thus, both unigram and bigram topics convey the same meaning which reassures the fact.

**2.2 Sentiment Analysis**

**Logo

Description automatically generated with low confidence**

Figure 6

**A picture containing logo

Description automatically generated**

Figure 7

The sentiment analysis results of both rumour and non-rumour tweets are shown in figure 6 and figure 7, respectively. The pie charts are divided into positive and negative sentiments based on average score. The sentiment is calculated using a pre-trained sentiment classifier imported from a library called *Flair*. *Flair* uses state of an art Natural Language Processing technique implemented using *BERT* and *Elmo* embeddings. Flair takes in the sentence as input and predicts a sentiment score to classify it into positive and negative classes.

From the pie chart, it is evident that rumour tweets tend to have more negative sentiment when compared to non-rumour tweets. This can also be attributed to the fact that the most common topics of rumoured tweets are death, new cases, and coronavirus which tend to spread more negative messages.

**2.3 Propagation Analysis**

The propagation of subtweets is being analysed by comparing the date of creation of subtweets with the root tweet. This time frame is compared against the number of tweets to indicate the propagation speed of a tweet. The resultant of this analysis is plotted as a graph in Figure 6 with seconds on the horizontal axis and the number of tweets on the vertical axis.

A picture containing text

Description automatically generated

Figure 8 Average Propagation Speed

The graph indicates that the propagation is at a similar rate for both rumour and non-rumour tweets for a certain amount of time. After which, rumour tweets tend to die without any supporting retweets and comments. This can be attributed to the fact that people initially tend to react more to rumours without any verification of the information. But once the authenticity is questioned, the support for the tweet gets reduced and dies soon.

**2.4 Hashtag Analysis**

On comparing the hashtags used on rumoured and non-rumoured tweets, it is found that the frequently used hashtags are similar in both cases. The top 10 highest used hashtags of rumoured tweets are the same as that of non-rumoured tweets.

|  |  |
| --- | --- |
| Rumour Hashtags | covid19, coronavirus, breaking, cdnpoli, covid, china, wuhan, staysafeug, ccpvirus, covidー19 |
| Non-Rumour Hashtags | covid19, coronavirus, breaking, covid, china, covid\_19, covidー19, stayhome, coronaviruspandemic, lockdown |

Table 3: Top 10 Hashtags

**2.5 Retweets Count**

The count of retweets can be one of the major factors in identifying misinformation. The average number of retweets is compared for rumour and non-rumours and plotted as a graph which is displayed in *Figure 9.*

Logo

Description automatically generated

Figure 9

From the above figure, it is evident that the Non-Rumour tweets have a greater number of retweets on average. This can be attributed to the fact that authentic information will be acknowledged by a greater number of users.

References

Aizawa, A., 2003. An information-theoretic perspective of tf–idf measures. *Information Processing & Management,* 39(1), pp. 45-65.

Arkaitz, Z. et al., 2018. Detection and Resolution of Rumours in Social Media: A Survey. *ACM Computing Surveys,* 51(2).

Grobelnik, D. M. a. M., 1998. *Word Sequences as Features in Text-Learning.* s.l., CiteSeer.

Kaimin, Z., Chang, S., Binyang, L. & Jey, H. L., 2019. *Early Rumour Detection.* Minneapolis, Minnesota, Proceedings of NAACL-HLT.

Victor, S., Lysandre, D., Julien, C. & Thomas, W., 2020. *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter.* s.l., NeurIPS 2019.

Xavier, G., Antoine, B. & Yoshua, B., 2011. *Deep Sparse Rectifier Neural Networks.* Fort Lauderdale, FL, USA., 14th International Conference on Artificial Intelligence and Statistics (AISTATS).