COMP90042 Rumour Detection and Analysis on Twitter

Student IDs: 1067992, 1234032, 1237539

1 Introduction

With the popularity of the internet and social media in recent years, social media has become an indispensable part of people's lives. More and more people are using social media not only to connect with family and friends, but also to keep up to date with what is currently happening. The role of the user has also changed from that of a receiver of information to that of a creator. Many news outlets publish news on social media, and ordinary users can also express their views and opinions. However, the growth of social media has accelerated the spread of information and has also brought about a proliferation of false rumours. Twitter, one of the most popular social media, has generated a large number of rumours that are difficult to verify and are spreading rapidly among users. Therefore, it is crucial to identify whether a message is a rumour or not.

2 Related Works

The topic of rumour identification and analysis has attracted a great deal of attention in recent years.

Tian et al. (2020) transformed the rumour detection task into a supervised classification problem and trained the model using a set of labelled source tweets and comments. They found that comments responded to different user attitudes towards rumours and non-rumours, and based on this decomposed the problem into rumour detection for detecting positions and tweets from comments. Convolutional neural networks (CNN) and BERT neural network language models were used for the experiments. The article further proposes a rumour detection model based on a combination of these two models with significant advantages for early rumour detection.

Some studies analysed whether a tweet is a rumour by using features other than the content (message of a tweet). Kwon et al. (2013) detected rumours by analysing time, structure and language. They found that non-rumours usually had only one significant spike, while rumours tended to have multiple periodic spikes. They also found that rumours and non-rumours differed significantly in the user diffusion network. A new time series fitting model and network structure are used to train the data. This article shows a more accurate result of identifying rumors from other type of information.

Wu et al. (2015) examined the problem of false rumours on the popular Chinese social network Weibo. In addition to studying traditional semantic features such as topic-based and sentiment-based features, propagation patterns were also investigated through a hybrid Support Vector Machine (SVM) classifier based on a graph-kernel. They found that the false rumor is first posted by a normal user, then reposted and supported by some opinion leaders and finally reposted by a large number of normal users. On the contrary, the normal message is posted by an opinion leader and reposted directly by many normal users. Based on these findings, the model can be used to detect false rumours early, and 90% of rumours can be detected just one day after their initial broadcast.

3 Dataset

In this project a model was trained based on the given training data, optimised on the development data, tested on the test data, and ultimately applied to determine whether tweet events related to COVID-19 are rumours. All the datasets contain events, each having tweet IDs of a source tweets and its replies. When crawling for the tweets by their IDs using the Twitter API, some tweets could not be retrieved as they have either been deleted or the user accounts have been deactivated. These tweets have been ignored in this project.

Before building the rumour analysis model, the tweet data was analysed.

Dataset	Percentage
train-rumour	0.2216
train-nonrumour	0.7784
dev-rumour	0.2199
dev-nonrumour	0.7801

Table 1: Class distribution in the datasets

Dataset	Median	Mean
train-rumour	14	30.2
train-nonrumour	5	10.7
dev-rumour	12	26.8
dev-nonrumour	5	12.6

Table 2: Number of replies

4 Task 1

4.1 Obtaining Tweet Object Data

The objective for task 1 is to develop a rumour classifier that is able to detect rumours from tweets. To do this we need to first use the Twitter API to obtain tweet object data based on the tweet IDs provided. To do this, the /search/tweets interface provided by Twitter was used through a written script. According to the Twitter API documentation, the interface can be requested up to 200 Tweet IDs at a time, so the tweet IDs were split into to a 2D arrays of 200 to minimise the number of requests needed.

4.2 Data Preprocessing

Only the texts of tweets were used in our model as it is the central aspect in classifying whether a tweet is a rumour or not, and the introduction of other many features can also increase the complexity in model fine tuning.

From these texts, mentions (@'s), URLs, hyperlinks, and newline (\n) and retweet symbols (\r) were stripped. For hastags (#), however, only the hash symbol was removed as the word in the hashtag could be part of a sentence in some tweets.

Some of the tweet's texts end up being empty after filtering. In these cases, those are ignored and we proceed to the next reply tweet for the event. However, some events are standalone tweets and when those kind of tweets are empty they are labelled as 'invalid' and added to the list. They cannot be ignored as this would mess up the number of corresponding labels/predictions.

Finally, as shown in *Table 1*, there is a large imbalance in the instances of rumour and non-rumour events. To address this issue, the rumour class was oversampled so that the proportion of data in both classes would approximately the same.

4.3 Model Architecture

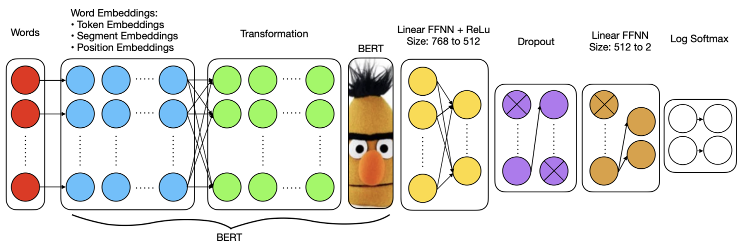

Figure 1: Architecture of the Used BERT Model.

The initial model consists of the following layers: the pre-trained BERT forms the Embedding layer and the Transformation model layer, followed by the main BERT layer and two layers of fully-connected feed-forward neural networks (FFNN) (first layer: size 768 to 512, second layer: size 512 to 2) with a ReLu and Dropout layer in between them, and at the end we use Log-Softmax to output predictions. The end product of size 2 represents the two classes (nonrumour: '0' and 'rumour':'1'). The loss function used during training and evaluation is the Negative Log-Likelihood Loss (NLLL) function to complement the Log-Softmax layer. In the back-propagation phase the model uses AdamW as our optimiser, which improves the generalisation ability of Adam optimiser by decoupling the weight decay from the gradient-based update.

The first two layers of the model, the Embedding and Transformation layer, is concerned with tokenizing the input sentence and transforming each token into a fixed dimensional vector representation. The obvious advantage of using BERT Embeddings over traditional methods of extracting information from raw text, such as tf-idf or Bag-of-Words, is that since BERT uses sub-word level embeddings rather than word level embeddings the probability of out-of-vocabulary decreases. This is especially helpful when dealing with data from online platforms like Twitter, where the use of words tend to be more casual and informal.

The main layer consists of a pretrained BERT model. This model has already been trained in a self-supervised manner on a large English corpus, which not only saves on time but also, being given a training set with around 20,000 tweets, ensures that the model will be able to identify more information when used for the downstream rumour classification task.

The ReLu and Dropout functions transform the data from the first FFNN layer by 'activating' certain neurons and randomly zeroing out some of the elements respectively. ReLu is used as it is regarded as one of the best activation functions due to its lack of drawbacks, such as the 'Vanishing Gradient' problem (Goswami (2020)). Dropout is used as normalisation as it forces the model not to rely on the data too much and motivates learning the actual relationship between the data and the labels.

Log Softmax is used instead of normal softmax as more it has better optimisation and higher penalties for incorrect classes. With the high complexity of the model it is best to have a strict learning process to induce a more accurate model.

4.4 Development Phase

Certain hyperparameters within the different parts of the initial model were modified by evaluating on the development dataset and recording the losses and accuracies for different values/methods. The hyperparameters considered for further development through evaluation include i) dropout rate, ii) regularisation (L1 and L2), and iii) learning rate of the AdamW optimiser.

The model was first trained using the training set and carried out 10 epochs. After each epoch it calculates the total loss from evaluating the development set and if the loss was lower the previously lowest recorded loss, it takes saves the model weights of that epoch as the model's "best state". The initial hyperparameters used prior to the development phase were i) dropout rate = 0.1, ii) no regularisation, and iii) AdamW learning rate = 2e^{-5}.

For the dropout rate, rates 0.2, 0.3, 0.4, and 0.5 were also considered. It can be seen that from *Figure 2* that rates 0.1, 0.2, and 0.3 overall have the lowest losses. However, amongst them, 0.1's lowest loss did not produce its highest accuracy, and between 0.2 and 0.3, 0.3's lowest loss is lower than 0.2's, leading to its slightly higher accuracy. Therefore, **0.3** has been chosen as the ideal dropout rate.

As for regularisation, L1 and L2 regularisation were considered along with λ values of 0.001, 0.0001, 0.00001. Along with this, the ideal dropout rate, 0.3, was also used. From *Figure 2*, it is difficult to analyse the losses each regularisation

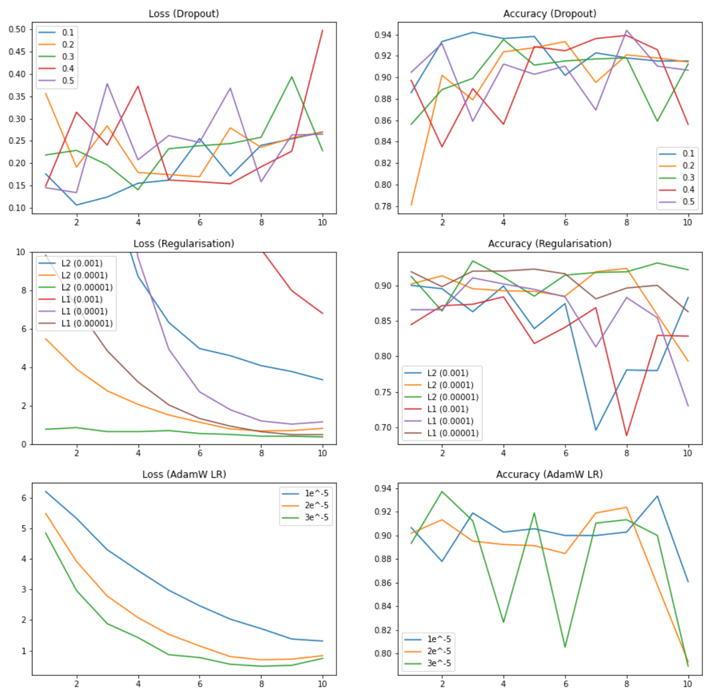

Figure 2: Losses and Accuracies for Dropout Rate, Regularisation, and AdamW Learning Rate.

method as a higher value of lambda makes a big impact. Both L1 and L2's accuracy with $\lambda = 0.001$ had the worst performing accuracies, and L2's accuracy for both $\lambda = 0.0001$ and 0.00001 beat L1's correlating accuracies. It can be seen that in the epoch that L2 0.00001 had its lowest loss (10) its accuracy is less than that of L2 0.0001's lowest losing epoch (8). Therefore, the ideal regularisation method is **L2 0.0001**.

For AdamW's Learning Rate (LR), only 1e^{-5} and 3e^{-5} were considered. Here the ideal dropout rate, 0.3, and regularisation method, L2 0.0001, were used. By observing *Figure 2*, at LR 1e^{-5}'s lowest loss epoch (10) it has the worst accuracy. At LR 3e^{-5}'s lowest loss epoch (9) its accuracy is lower than LR 2e^{-5}'s lowest loss epoch's (8) accuracy. Therefore, we retain 2e^{-5} as the ideal LR for AdamW.

In the end, the model with these ideal hyperparameters resulted in an average loss of 2.00 (3 sf) and an average accuracy of 88.7% (3 sf).

5 Task 2

5.1 Data

There are 17,458 events in the COVID-19 tweets dataset. 2,564 of them are classified as 'rumour' and 14894 of them are classified as 'non-rumour'. Most rumours and non-rumours have a low number of replies. Although the tweet that had the highest number of replies were non-rumours, it was clear that rumours generally had more replies than non-rumours as shown in *Figure 3* and *Figure 4*.

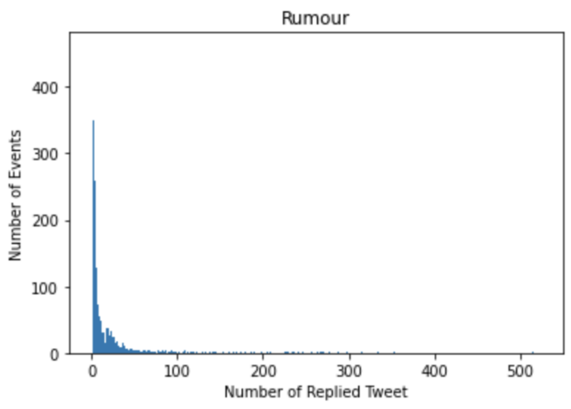

Figure 3: The number of replies to rumours.

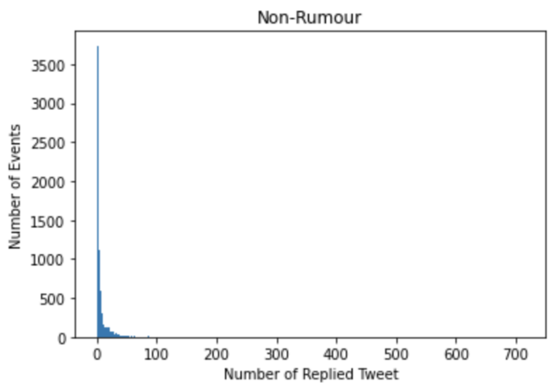

Figure 4: The number of replies to non-rumours.

5.2 Hashtags

After obtaining the hashtags from the tweets, some hashtags have the same wording but different case, which can interfere with the analysis of the hashtag. Therefore, all hashtags were lowercased and the number of times they were used was recalculated.

The five most popular hashtags for both rumours and non-rumours are "covid19", "coronavirus", "trump", "coronaviruspandemic", and "trumpvirus" (*Figure 5* and *Figure 6*).

By comparing the popular hashtags for rumours and non-rumours, hashtags used in rumours were also used in non-rumours. Only a small portion of hashtags are different for rumours and non-rumours, for example, "biden2020", "trumpisanationaldisgrace", and "blacklivesmatter" are three hashtags that are commonly used by non-rumors and not by rumors. "stayhome", "coronavirusoutbreak", and "lockdown" are three hashtags that

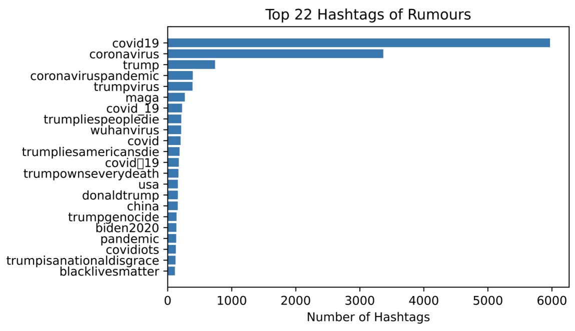

Figure 5: Hashtags for rumours.

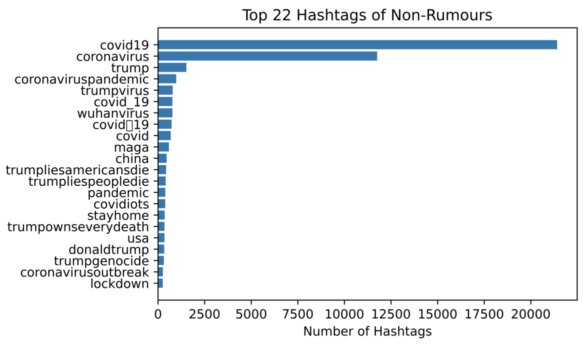

Figure 6: Hashtags for non-rumours.

are commonly used by rumours and not by non-rumours.

The tweet hashtags that often appear in rumours and non-rumours are just normal tags for news or events. There is little difference between rumours and non-rumours about hashtag, probably because rumours need to be spread and disguised with the help of the same hashtags as non-rumours. Therefore, it is difficult to distinguish between rumours and non-rumours in hashtags.

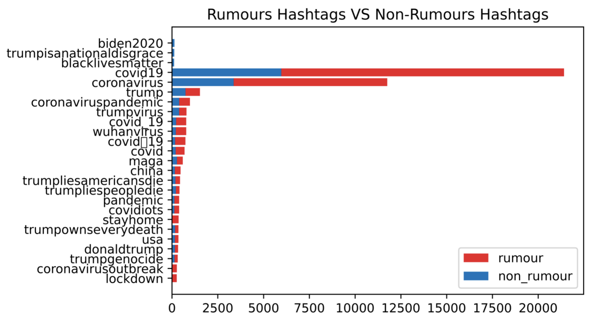

Figure 7: Hashtags for both rumours and non-rumours.

5.3 Sentiment

To analyse the semantics of rumour tweets and non-rumour tweets, vaderSentiment tool has been used. VADER is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media. VADER not only informs a Positivity and Negativity score but also expresses how positive or negative a sentiment is.

In *Table 3*, most rumours and non-rumours are neutral, with only a small proportion of positive and negative ones. However, it is clear that the proportion of negative rumour tweets is greater than the negative non-rumour tweets. Similarly, there is a higher proportion of positive non-rumour tweets than positive rumour tweets. The replies also show specific patterns *Table 4*, except that for positivity and negativity, the difference between the rumours and non-rumours decreases.

	Rumour	**Non-Rumour**
`Negative`	11.95%	8.41%
`Neutral`	80.19%	83.54%
`Positive`	7.86%	8.05%

Table 3: Sentiment Analysis

	Rumour	**Non-Rumour**
`Negative`	13.96%	12.47%
`Neutral`	77.41.19%	78.54%
`Positive`	8.63%	8.99%

Table 4: Sentiment Analysis for Replies

5.4 Users

	Rumour	**Non-Rumour**
`created time`	2010.6	2010.7
`followers`	4622051	6589871
`following`	6891	8803
`vertified`	79%	81%

Table 5: User Average Information

By collecting information on the users of source tweets, rumour users and normal users were analysed. As some characteristics could not be analysed, such as "profile image URL", only the time of creation of the user's account, the number of followers, the number of followings, and whether the account was verified were collected.

As shown in Table 5, the users of a rumour tweet were not significantly different from normal users in terms of creation time, but rumour-sourced accounts have slightly fewer followers and fans. In addition, the verification rate of source account of rumour was not as high as that of a normal user.

6 Conclusion

After the private leaderboard was released, the model obtained a private score of 85.333%. While the final model was improved in the development phase, much more could have been done to further improve it, such as including the implementation of tweet object features other than the text or experimenting with different layers. Further research could also have been made to implement a better model, such as Automated BERT Regularisation (AUBER), which due to time constraints the project team did not get into. However, that does not mean the model was a failure. Given the amount of time and other academic and personal commitments of the team through the project, the results turned out better than expected.

References

Divyang Goswami. 2020. Comparison of sigmoid, tanh and relu activation functions.

Sejeong Kwon, Meeyoung Cha, Kyomin Jung, Wei Chen, and Yajun Wang. 2013. Prominent features of rumor propagation in online social media. In *2013 IEEE 13th international conference on data mining*, pages 1103–1108. IEEE.

Lin Tian, Xiuzhen Zhang, Yan Wang, and Huan Liu. 2020. Early detection of rumours on twitter via stance transfer learning. In *European Conference on Information Retrieval*, pages 575–588. Springer.

Ke Wu, Song Yang, and Kenny Q Zhu. 2015. False rumors detection on sina weibo by propagation structures. In *2015 IEEE 31st international conference on data engineering*, pages 651–662. IEEE.