Rumour Detection and Analysis on Twitter

1235216 1215907 1195789

1 Introduction

Since the introduction of social media, misinformation has drawn wide concern in its impact on social and digital life. However, such influence has been amplified and accelerated since the covid-19 global pandemic, a period when people are increasingly relied on social media and spending more time on it due to lockdowns and traffic restrictions. In this regard, rumour detection systems and rumour analyses demonstrate pivotal importance on various aspects.

To achieve the aim of developing an effective rumour detection system and conducting further rumour analyses, this report will first compare different NLP classification models' performance on rumour detection over covid-related tweets. Then, the classifier with the best performance will predict labels of covid-19 related tweets and analyse the characteristics of the rumours and non rumours.

2 Dataset and Data Pre-processing

The data set used for rumour detection is crawled by utilising the provided tweet IDs. For the training set, there are a total of 1563 tweet "events" after crawling the data, which include both source and reply tweets.

Dataset	Provided events	Crawled events	Rumour	Non Rumour
Train	1895	1563	346	1217
Dev	632	536	112	424
test	558	N/A	N/A	N/A

Table 1: Rumour Distribution Among Train, Dev And Test

As demonstrated in the table above, the ratio of training set and dev set is 65:35. And the events labelled rumour and non-rumour are severely unbalanced.

The data pre-processing can be divided into several parts.

1. Concatenate each tweet event: Two ways are used. The first is a topic concatenated event, using DFS to traverse and combine the tree structure of the tweet event. The source tweet is the root, the node is the tweet with replies, and the leaf is the reply. The tweets/replies about the same topic will be linked in this case. The second is timely concatenate event by sorting the entire tweet event as the time it was created.

2. Feature selection: The crawled tweet is in the form of the JSON, which contains many features. Therefore, feature selection from the JSON is needed. The most signification feature is the tweet's text; apart from this, the valuable information 'retweet_count' is also selected for training the model.

3. Text pre-processing: The "@people", emojis, URLs, punctuation and the "#" sign in hashtags are removed. Since hashtags often provide important information that assists the model in determining if the whole tweet event is rumour or not.

After preprocessing, the train, dev, and test dataset outputs are three CSV files that contain the texts, total retweet counts, and provided labels (excluded from test output)

3 Models

3.1 LSTM, GRU, Bi-LSTM Modelling

Because Recurrent Neural Networks have a short-term memory problem and are unable to grasp the long-term dependencies that are presented in texts. To overcome the limitations of the vanishing gradient problem, Bi-LSTM, LSTM, and GRU networks (Chung et al., 2014; Graves & Schmidhuber, 2005; Hochreiter & Schmidhuber, 1997) were used. These three models are implemented using the Keras model. Sequential from TensorFlow to build the network sequentially. For example, in the

LSTM model, the first layer is the embedding layer, which turns the input nth dimension one-hot vector of each word into a 300-dimensional vector. And 100 LSTM units are added with a dropout rate of 0.3. At last, a single neuron with sigmoid function, which takes the output of LSTM cells to forecast the outcomes. The Adam optimizer and binary cross entropy loss function are used to compile the model. The graph visualisation of different model properties is shown in the figure down below.

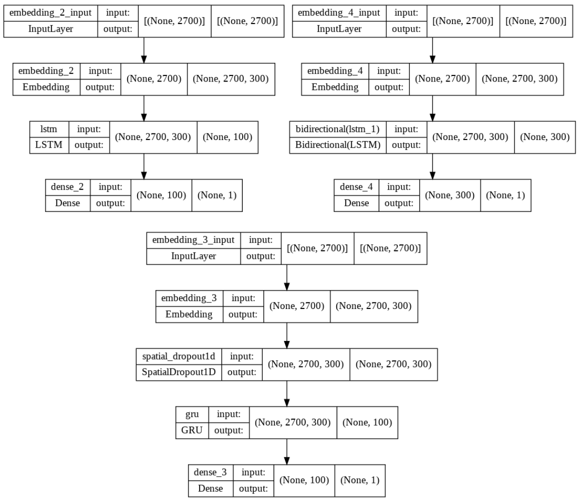
embedding_2_input | input: | output: | InputLayer | [(None, 2700)] | [(None, 2700)]
embedding_2 | Embedding | input: | output: | (None, 2700) | (None, 2700, 300)
lstm | LSTM | input: | output: | (None, 2700, 300) | (None, 100)
dense_2 | Dense | input: | output: | (None, 100) | (None, 1)
embedding_4_input | InputLayer | input: | output: | [(None, 2700)] | [(None, 2700)]
embedding_4 | Embedding | input: | output: | (None, 2700) | (None, 2700, 300)
bidirectional(lstm_1) | Bidirectional(LSTM) | input: | output: | (None, 2700, 300) | (None, 300)
dense_4 | Dense | input: | output: | (None, 300) | (None, 1)
embedding_3_input | InputLayer | input: | output: | [(None, 2700)] | [(None, 2700)]
embedding_3 | Embedding | input: | output: | (None, 2700) | (None, 2700, 300)
spatial_dropout1d | SpatialDropout1D | input: | output: | (None, 2700, 300) | (None, 2700, 300)
gru | GRU | input: | output: | (None, 2700, 300) | (None, 100)
dense_3 | Dense | input: | output: | (None, 100) | (None, 1)

Figure 1: Graph Visualisation of LSTM, Bi-LSTM, GRU Neural Networks

As for the further improvements, adjusting the word embeddings to pretrained embeddings such as Stanford Glove embeddings and adding extra layers can be implemented.

3.2 Bidirectional Encoder Representations from Transformers (BERT)

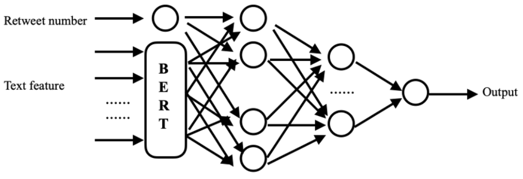

Figure 2: BERT Classifier Topology

Unlike the other language representation model above, the BERT model has a pre-train representation from the massive text (Devlin et al., 2019). Thus, fine-tuning the pre-trained model with an extra output layer could create a great performance classification model.

This research uses a BERT pre-trained model combined with a two-layer feed-forward neural network with one hidden layer as the classifier. The model's topology is shown in the figure above; one extra feature (retweet number) is also associated with the BERT language representation feature and fed to the classifier.

For the BERT pre-trained model, the "BERT-base-uncased" AutoTokenizer is selected, which eliminates the need for lemmatisation/stemming and stop-word removal. By using the BERT tokenizer, special tokens [CLS] and [SEP] are added to the beginning and end of each sentence. To fully expand the text, the encoded text representation is padded to the MAX_LEN with 512.

Input ids (index of each token in BERT-base vocabulary), attention mask (1 for input tokens, 0 for padding), and token type ids (using 1s and 0s to distinguish two different sentences) are selected from the output of the auto tokenizer. Along with the "retweet_count", those four features are fed into the classifier.

As for the classifier, a two-layer neural network (NN) is used. Compared with the one layer classifier, NN with a hidden layer can take advantage of any decision boundary of any accuracy that can be represented (Heaton, 2017). As for the activation function used for the NN classifier. The tanh function is used in the hidden layer. Compared with the Relu function, it can have a fast learning speed but can lead to a dead Relu issue in our dataset. Also, this classifier uses the back-propagation to train the model. The loss function uses the BCEWithLogitsLoss, which has the advantage of being numerically stable (Ruby et al., 2020) and the Adam optimizer with the learning rate 2e-5, which is computationally efficient (Zhang, 2018). The number of the hidden neutrons takes 1/3 of the input feature with the number 256.

For the training step, the batch size tested ranges from 4, 8, 16. And the batch size 8 is chosen with the best performance, which could avoid training stops at the local minimum with suitable memory capacity. Also, the training epoch is 6 as the training can overfit from 7 epochs.

4 Performance and Analysis

As the dataset used for training and testing is unbalanced, thus the evaluation matrix of the model is shown below under the development dataset. The most meaningful evaluation matrix are precision, which denotes the percentage of correctly detected as a rumour over all datasets.

Table 2 below demonstrates that in comparison

	Precision (Dev)	Recall (Dev)	F1-score (Dev)	Accuracy (Test/Kaggle score)
LSTM (using timely concatenate event)	0.82	0.72	0.76	0.73
GRU (using timely concatenate event)	0.77	0.62	0.66	0.70
Bi-LSTM (using timely concatenate event)	0.78	0.85	0.81	0.74
BERT (using timely concatenate event)	0.88	0.97	0.92	0.91
BERT (using topic concatenate event)	0.86	0.96	0.91	0.89

Table 2: Results of the performance of different models

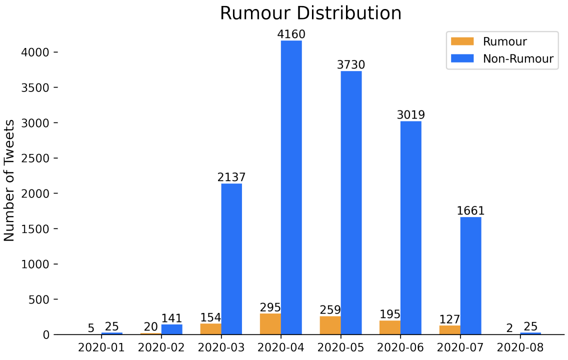

Figure 3: Rumour Distribution Over Time

with the performance of simple RNN Variants with a pre-trained BERT model.The experimental results show that BERT model can achieve significantly higher results than the different simple RNN Variants. This can be interpreted into two main points. The first one is that the pretrained embeddings, which can capture the semantic and syntactic meaning of a word to further boost model performance, are not employed. And the second one could be that the simple RNN Variants treat the whole tweet event text equally, while the attention-based BERT could lead it to focus more on specific meaningful terms within a lengthy text. For the BERT model, the one using the timely concatenate event can have the best performance with a precision of 0.88, recall of 0.97, F1 score of 0.92, and the highest Kaggle score of 0.91. And the BERT classifier outperforms the other three models substantially. The BERT using the topic concatenate event has the worse performance. The valuable reply topic could cause this in the event can be too far away from the source tweet.

5 Application – Rumour Analysis

5.1 Dataset

The dataset used for rumour analysis is based on the data retrieved from covid.data.txt that contains a list of tweet IDs of source tweets and their replies. Since most of the tweets are posted in 2020, only part of them are successfully retrieved (15,963 out of 17,458). After applying the rumour classifier built in the first task, 1,057 source tweets are labelled as rumours, 14,898 source tweets are labelled as non-rumours. The following analyses are based on fields like "text", "created_at", "public_metrics", and hashtags on these tweets.

5.2 Analysis and Results

5.2.1 Rumour Distribution Over Time

Through plotting the count of rumours and non-rumours within each month (see Figure 3), correlation between them is discovered. Both types peaked in April 2020 then slowly decreased in the coming months, where non-rumours are normally around 10 times the count of rumours. The surge of covid tweets in March 2020 may because WHO's announcement of covid-19 outbreak a global pandemic on March 11, 2020 (Cucinotta & Vanelli, 2020).

5.2.2 Topic Modelling

To extract the topics in rumour and non-rumour tweets, non-negative matrix factorisation (NMF) is applied with tf-idf vectorisation. Table 3 demonstrates the top five words of each topic (10 topics are extracted from each data set).

The topic modelling results show some degree of difference between rumour and non-rumour tweets. For instance, the word "Trump" appeared several times in rumour topics, as well as "America" and "American". Whereas the word "Test" only appeared in non-rumour tweets. Overall, it is observed that non rumour tweets are more fact based while rumours are more emotional in forms of mentioning countries and political people.

5.2.3 Most Common Words

In addition to topic modelling, most common word analysis is another way to understand the themes in the dataset. With the use of TweetTokenizer from

Topic	Rumours	Non Rumours
1	coronavirus, trump, time, new, twitter	coronavirus, pandemic, china, outbreak, news
2	trump, americans, coronavirus, did, died	covid, 19, patients, 2020, hospital
3	covid, 19, says, man, florida	cases, new, deaths, coronavirus, states
4	deaths, 000, flu, covid19, cases	covid19, lockdown, 2020, today, read
5	covid19, trump, amp, help, house	health, pandemic, need, care, crisis
6	china, virus, pandemic, flu, chinese	positive, tested, coronavirus, test, tests
7	biden, american, country, realdonaldtrump, president	trump, president, house, white, administration
8	don, people, coronavirus, going, know	amp, today, time, america, great
9	just, trump, say, mask, right	people, just, 000, death, died
10	president, coronavirus, flu, kung, rate	says, spread, new, home, mask

Table 3: Topic Modelling

the NLTK library, and removing the stop words as well as applying lemmatisation, the count of words appearing in each of the rumour and non-rumour tweets is recorded.

As figure 4 displays, "trump" is the most frequent word appearing in the tweets labelled as rumours, words like "president" and "American" are also highly ranked. In comparison, apart from the word "trump" ranked as the 7th highest, all other words in the top 10 words in non-rumour tweets are all covid related words.

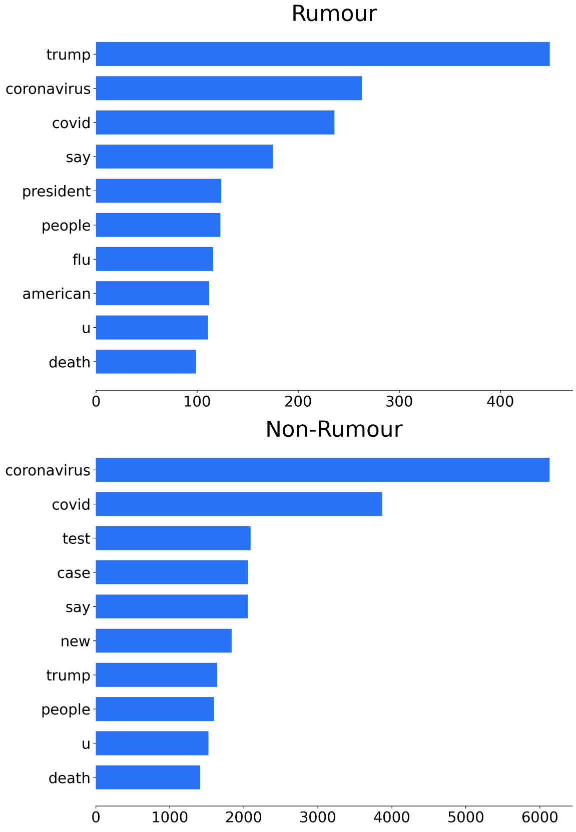

Figure 4: Top 10 Most Common Words

5.2.4 Hashtag Analysis

Another aspect to inspect the tweet data is through hashtags. As a label manually chosen by the users, hashtags can reflect the theme of one tweet to some degree, and helps others to locate information with ease. By using regular expressions to match the hashtags in each tweet, the count of hashtags in rumour and non-rumour tweets is collected (see Table 4). One interesting difference is that rumour tweets seemed prefer using #coronavirus much more than non-rumour tweets. One other hashtag #recallkatebrown also distinguished rumour and non-rumour tweets.

Rumours	Non Rumours
covid19, coronavirus, recallkatebrown, lockdown, covid, coronaviruspandemic, covid-19, china, pandemic, stayhome	covid19, coronavirus, breaking, covid, stayhome, china, covid-19, covid_19, coronaviruspandemic, cdnpoli

Table 4: Hashtag Analysis

5.2.5 Sentiment Analysis

Sentiment analysis provides a perspective beyond word-level analyses of tweets. Through using SentimentIntensityAnalyzer from NLTK, the positive, negative, and neutral scores of each source tweet is calculated and aggregated together to give an overall view. As a result, rumour tweets have 6.7% positive, 12.3% negative, and 81.0% neutral sentiment. Non-rumour tweets have 7.9% positive, 8.2% negative, and 83.9% neutral sentiment. It is observed that non-rumour tweets have similar proportions of positive and negative sentiment, while rumour tweets have double the amount of negative sentiment than positive ones. This is in line with the traits of rumours, they are misinformation and deliberately false information that normally convey negative emotions (Qazvinian et al., 2011).

5.2.6 User Engagement

Broadly defined as measurement of comments, likes, and shares (Dolan et al., 2016), user engagement is a solid metric to evaluate one's social media success. The "public_metrics" field contains this information of each source tweet. Proportionally, rumour tweets have 19.4% retweet, 7.0% reply, 69.3% like, and 4.3% quote. Non-rumour tweets have 19.0% retweet, 6.7% reply, 70.5% like, and 3.8% quote. The large amount of tweets in the dataset may explain this similarity. Though, it is still differentiable that rumour tweets have less proportion of likes than non-rumours because less people agree with them. Meanwhile, rumour tweets also have a higher proportion of replies and retweets since there may exist more debates in rumour tweets and rumours normally spread wider than usual information.

6 Conclusion

This report started with a brief introduction to the influence of rumours on social media and the importance to differentiate it from non-rumours. It then introduced the dataset used in training and testing the performance of the classifier proposed and corresponding experiments. Next, the details of data preprocessing and data engineering were descripted as well. The next section walked through LSTM, GRU, Bi-LSTM, and BERT modelling methods, as well as the neural network classifier adopted. Regarding performance, it is discovered that BERT models have the best performance over all other models.

Moreover, the classifier was applied on a set of unlabelled covid data to detect rumours. Analyses such as topic modelling, most common words, hashtag analysis, sentiment analysis, and user engagement were conducted, with the conclusion that rumour and non rumour covid tweets do have different characteristics.

For future directions, it is recommended that more features related to the tweets can be added, due to the limitation, many features can not be crawled from Twitter, such as the follower of the resource tweet publisher and the like number of the tweets. Moreover, the outputs from the BERT model can be fed to a convolutional Neural network which is able to learn more features from the inputs, instead of a linear classifier.

References

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *arXiv:1412.3555 [cs]*. ArXiv: 1412.3555.

Domenico Cucinotta and Maurizio Vanelli. 2020. WHO Declares COVID-19 a Pandemic. *Acta Bio-Medica: Atenei Parmensis*, 91(1):157–160.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]*. ArXiv: 1810.04805.

Rebecca Dolan, Jodie Conduit, John Fahy, and Steve Goodman. 2016. Social media engagement behaviour: a uses and gratifications perspective. *Journal of Strategic Marketing*, 24(3-4):261–277. Publisher: Routledge _eprint: https://doi.org/10.1080/0965254X.2015.1095222.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5):602–610.

Jeff Heaton. 2017. The Number of Hidden Layers.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780. Conference Name: Neural Computation.

Vahed Qazvinian, Emily Rosengren, Dragomir R Radev, and Qiaozhu Mei. 2011. Rumor has it: Identifying Misinformation in Microblogs. page 11.

Usha Ruby, Prasannavenkatesan Theerthagiri, Jeena Jacob, and Yendapalli Vamsidhar. 2020. Binary cross entropy with deep learning technique for Image classification. *International Journal of Advanced Trends in Computer Science and Engineering*, 9(4):5393–5397.

Zijun Zhang. 2018. Improved Adam Optimizer for Deep Neural Networks. In *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*, pages 1–2. ISSN: 1548-615X.

7 Team Contribution

1235216:	33.3%
1215907:	33.3%
1195789:	33.3%