# COMP90042 Project 2021:
# Rumour Detection and Analysis on Twitter

**889835**

## 1 Introduction

Rumour detection is an important tool to have in today's digital age when moderating social media platforms. This is due to the increase in misinformation coupled with the ability for information to spread at an exponential rate before false information can be taken down.

This report aims to evaluate several models that can be used to classify tweets as rumours or non-rumours. The report also aims to evaluate the model on a set of tweets related to COVID and to gain insight into possible relationships between features of the tweet set and its label.

## 2 Data Exploration

Prior to building the rumour analysis model, the tweet data was explored to identify relationship between the label of a tweet and the features of it's replies. The features explored were the number of replies and sentiment of replies to the source tweet.

The NLTK Vader sentiment analyzer (Loper and Bird, 2002) was implemented to provide a sentiment score for each tweet. The analyzer was used due to the simplicity of its implementation.

As seen in Table 1 there seems to be a relationship between the number of replies a source tweet has and its class in the training set, with rumour source tweets having fewer replies compared to their non-rumour counterparts. However, with reference to the dev set, there does not seem to be a relationship. As seen in Figure 1, in both training and development sets, replies to tweets classified as rumours had a more negative sentiment score compared to replies to tweets classified as non-rumours. This could indicate a possible relationship between a source tweets class and the sentiment of its replies.

| Label | Mean | Median |
|---|---|---|
| Rumour (Training) | 14.47 | 12.0 |
| Rumour (Dev) | 16.65 | 12.0 |
| Non-Rumour (Training) | 17.52 | 14.0 |
| Non-Rumour (Dev) | 17.44 | 15.0 |

Table 1: Number of Replies to Source Tweet

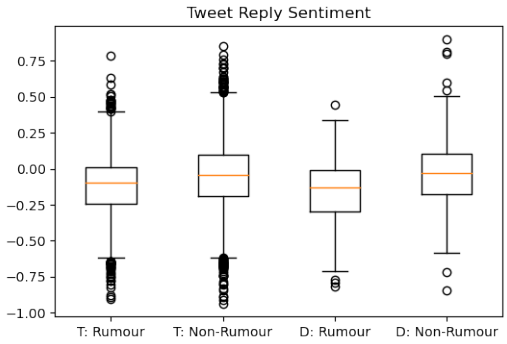


Figure 1: Tweet Reply Sentiment Scores

## 3 Task 1: Rumour Detection

### 3.1 Bag of Words

A bag of words model was chosen as a baseline to get a benchmark performance metric for comparisons with more complex models. Logistic regression was used due to the appropriate use of the sigmoid function for the binary rumour detection classification problem. Multinomial Naïve-Bayes was used due to its simplicity and its synergy with textual data.

The model was created by creating a bag of words vector from training tweet text and fitting the vectors into the classifier. Logistic regression and multinomial naïve-bayes classifiers were used to model the probability of whether a tweet was a rumour or non-rumour. This was implemented with the sklearn library (Pedregosa et al., 2011).

An attempt to add a rumour class bias to tweets that had replies with negative sentiment. This was

done by adding 0.2 to the probability that the tweet would be classified as a rumour if the mean sentiment score of the tweet replies is less than -0.1. However, this made no change to the predictions made on the development set. The results of the logistic regression and multinomial naïve-bayes classifiers can be seen in Table 2.

### 3.2 BERT

A BERT model was then chosen to classify the tweets as the BERT model improves upon language models that are constrained by unidirectionality such as RNN models. It does this by training a bidirectional model that uses transformers, an attention mechanism (Vaswani and Polosukhin, 2017), that accounts for left and right contexts in a sentence (Devlin et al., 2018). The model is appropriate in the context of rumour detection as the model is trained to learn contextual representations rather than to generate language. The BERT model was also selected due to its performance.

Pre-trained BERT models, "bert-base-cased" and "bert-base-cased-finetuned-mrpc", were pulled from the transformers library (Wolf et al., 2019) and implemented using PyTorch (Paszke and Desmaison, 2019). A classification layer with the labelled training data was added on top of the transformer output to fine-tune the pre-trained models. The "bert-base-cased" model was chosen to benchmark the performance of the model compared to the bag of words model.

The "bert-base-cased-finetuned-mrpc" model was chosen to increase the performance of the model as it had been trained using the Microsoft Research Paraphrase Corpus. The tokenizers used in the model were also pulled from the transformers library. An Adam optimizer was chosen to be used with the BERT model due to its simplicity in its implementation, efficiency and ability to work well with noisy and sparse gradients. The Adam optimizer is seen to perform well compared to other stochastic optimization methods (Kingma and Ba, 2014). Weight decay when using the optimizer was also adjusted between 0.0 and 0.3 to prevent overfitting and to optimize model weights (Loshchilov and Hutter, 2017). A batch size of 16 was used and 10 epochs were used when training the model.

### 3.3 Analysis of Models

The metrics obtained from evaluating the model using the dev and test tweet sets can be seen in Table 2.

#### 3.3.1 Bag of Words: Logistic Regression & Multinomial Naive-Bayes

When evaluating the model on both dev and test sets, the model performed better than expected with an F1 score of 0.7921 on the dev set and 0.8066 on the test set. The model, however, had a higher precision than recall, indicating that the quality of predictions was higher than the quantity of correct predictions. When compared to the logistic regression model, the multinomial naïve-bayes model did not perform as well with lower precision and recall scores.

#### 3.3.2 BERT-Base: Learning Rate: 2e-5, Weight Decay: 0.0

The F1 score of the model is marginally higher when compared to the logistic regression classifier in both dev and test evaluations, indicating that the BERT model provided a small increase in performance when performing the predictions.

#### 3.3.3 BERT-MRPC: Learning Rate: 2e-5, Weight Decay: 0.0

When evaluating the model on both the dev and test set, the higher precision and recall metrics seen for this model indicates that the MRPC dataset used to train the model had allowed the model to classify rumour tweets more accurately than the BERT base model and the logistic regression classifier. However, the model also misses some actual rumour tweets. This indicates that there are other features that have not been used to train the model that could result in a lower recall score. The final evaluation indicates that the model performs well, with similar precision and recall as the test evaluation.

#### 3.3.4 BERT-MRPC: Learning Rate: 2e-5, Weight Decay: 0.3

When compared to the BERT MRPC model with a weight decay of 0, the model had a marginally lower F1 score, slightly lower precision, and higher recall in both the dev and test evaluations. This indicates that the higher weight decay value did aid in increasing the performance of the model.

#### 3.3.5 BERT-MRPC: Learning Rate: 1e-5 & 3e-5, Weight Decay: 0.0

When compared to the BERT MRPC model with a lower learning rate, the difference between precision and recall is higher in both the dev and test evaluations for both changes in learning rate. This indicates that the learning rates could have resulted

| Model | D: Precision | D: Recall | D: F1 | T: Precision | T: Recall | T: F1 |
|---|---|---|---|---|---|---|
| BoW: Logistic Regression | 0.8343 | 0.75401 | **0.7921** | 0.8391 | 0.7766 | **0.8066** |
| BoW: Multinomial NB | 0.7725 | 0.7807 | **0.7766** | 0.7475 | 0.7872 | **0.7668** |
| BERT-Base: LR:2e-5 WD:0.0 | 0.7905 | 0.8074 | **0.7989** | 0.8242 | 0.7979 | **0.8108** |
| BERT-MRPC: LR:2e-5 WD:0.0 | 0.8100 | 0.7754 | **0.7923** | 0.8851 | 0.8191 | **0.8508** |
| BERT-MRPC: LR:2e-5 WD:0.3 | 0.7860 | 0.7860 | **0.7860** | 0.8152 | 0.7979 | **0.8065** |
| BERT-MRPC: LR:1e-5 WD:0.0 | 0.8102 | 0.7513 | **0.7796** | 0.7549 | 0.8191 | **0.7857** |
| BERT-MRPC: LR:3e-5 WD:0.0 | 0.8046 | 0.7486 | **0.7756** | 0.7614 | 0.7128 | **0.7363** |
| **Model** | **F: Precision** | **F: Recall** | **F: F1** | | | |
| BERT-MRPC: LR:2e-5 WD:0.0 | 0.8571 | 0.8317 | **0.8442** | | | |

Table 2: Rumour Prediction Metrics (Dev(D), Test(T) & Final(F) Evaluation)

in the model moving too slowly or quickly, respectively, towards the optimal weights, possibly missing the optimal solution.

## 4 Task 2: Rumour Analysis

### 4.1 Topic Modelling

Topic modelling using the Latent Dirichlet Allocation model (Blei et al., 2003) implemented using the Gensim python module (Rehurek and Sojka, 2011). Topics were modelled to get an understanding of how topics differed between rumour and non-rumour tweets and also, how topics developed over time during the pandemic in 2020.

When comparing rumour and non-rumour tweets obtained using the LDA topic model, both tweet sets have similar topics with most topics related to COVID. However, many non-rumour tweets had topics with "@realdonaldtrump" as part of the topic. When evaluating the difference in topics obtained using the LDA model over time between 01-2020 and 08-2020 and splitting the data by month posted, in both rumour and non-rumour tweet sets, there were no topics that referred to death. However, from 03-2020 to 08-2020 most topics had referred to death.

### 4.2 Sentiment Analysis

#### 4.2.1 Parent - Child Comparison

NLTK's Vader sentiment Analyzer (Loper and Bird, 2002) was used to gain insight into any potential relationships between tweets classified as rumours or non-rumours and the tweet's sentiment. The analyzer was also used to investigate the relationship between the sentiment of a parent tweet (source) and its replies (child).

When comparing rumour or non-rumour tweets, the sentiment distribution for both classes is similar and are both negative as seen in Figure 2. When comparing parent tweets and child tweets, child tweets are seen to have a more negative distribution compared to source tweets in both rumour and non-rumour tweet datasets as seen in Figure 3 and Figure 4. This indicates a possible relationship between negative sentiment and tweet replies.

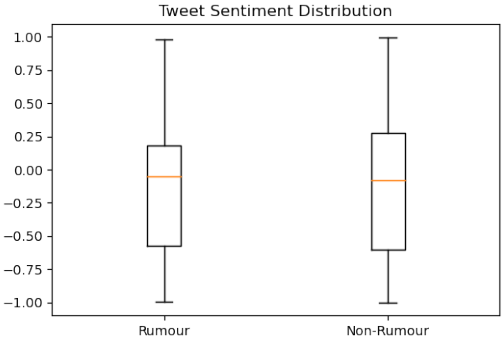


Figure 2: Tweet Sentiment Distribution


Figure 3: Rumours: Parent-Child Sentiment Distribution

#### 4.2.2 Tweet Sentiment Over Time

Sentiment analysis using NLTK's Vader sentiment analyser (Loper and Bird, 2002) was used to gain insight into a possible relationship between tweet sentiment over time. This was done by calculating the sentiment score for each tweet in the dataset and plotting the scores over the period 02-2020

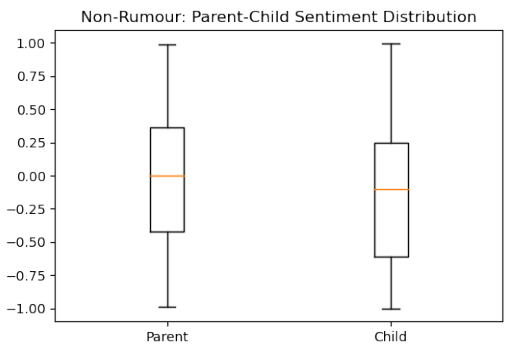


Figure 4: Non-Rumours: Parent-Child Sentiment Distribution

to 08-2020. As seen in Figure 5 and Figure 6, both classes of tweets have similar sentiment over the time period, indicating no clear relationship between tweet sentiment over time.

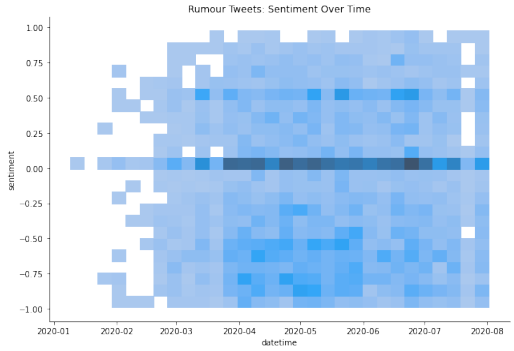


Figure 5: Rumour Tweets: Sentiment Over Time

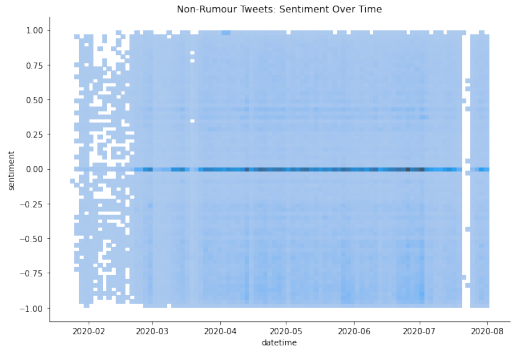


Figure 6: Non-Rumour Tweets: Sentiment Over Time

### 4.3 Popular Hashtags

The most popular hashtags were obtained by doing a count on tokens that begin with "#". This was to do see if there were any common topics that were prevalent in each category of tweets that topic modelling did not manage to pick up on. As seen in the results in Table 3, both rumour and non-rumour tweets have similar hashtags, however the main difference between the two tweet sets is that "#covid_19india" can be seen in tweets classified as rumours whereas tweets with "#trump" are classified as non-rumours. There could potentially be a bias towards these two hashtags in the classifier.

| Rumour Tweets | Non-rumour Tweets |
|---|---|
| '#covid19'(985) | '#covid19' (32596) |
| '#coronavirus'(375) | '#coronavirus' (18441) |
| '#covid_19'(74) | '#trump' (2551) |
| '#covid_19india'(68) | '#coronaviruspandemic'(1606) |

Table 3: Common Hashtags

| Label | Mean | Median |
|---|---|---|
| Rumour | 1065973.42 | 938.50 |
| Non-Rumour | 355464.19 | 305.00 |

Table 4: Follower Count Distribution

### 4.4 Follower Count

The follower counts of users were obtained by finding the mean and median of the tweet user's follower counts. This was done to determine if there was a relationship between a user's follower count and whether the user tweets rumours or non-rumours. As seen in the results in Table 4, tweets that were classified as rumours had a higher mean and median follower count. This indicates that there could be a relationship between a user's follower count and whether the user posts tweets classified as rumours.

## 5 Conclusion and Future Work

In this report, we explored the tweet data given and gained insight into possible features that could be related to a tweet being a rumour or non-rumour. We then utilised several classifiers to classify tweets into rumour and non-rumour classes. The models were chosen based on their model-specific advantages and in the case of the BERT model, it's performance in related literature. However, the results indicate that although the BERT MRPC model performed the best, the logistic regression classifier performed relatively well considering the minimal resources it takes to implement. The BERT MRPC model was then used to make predictions on a COVID-19 related tweet dataset and several insights were gained from the analysis of the predictions made. Possible future work could be done incorporating other languages into the model as the tweet set did not only contain the English language.

## References


D. M. Blei, A. Y. Ng, and M. I. Jordan. 2003. Latent dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022.

J. Devlin, M. W. Chang, K. Lee, and K. Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv:1810.04805*.

D. P. Kingma and J. Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

E. Loper and S. Bird. 2002. Nltk: The natural language toolkit. *arXiv preprint cs/0205028*.

I. Loshchilov and F. Hutter. 2017. Decoupled weight decay regularization. *arXiv:1711.05101*.

Gross S. Massa F. Lerer A. Bradbury J. Chanan G. Killeen T. Lin Z. Gimelshein N. Antiga L. Paszke, A. and A. Desmaison. 2019. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, and J. Vanderplas. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(oct):2825–2830.

R. Rehurek and P. Sojka. 2011. *Gensim–python framework for vector space modelling. NLP Centre, Faculty of Informatics*. Masaryk University, Brno, Czech Republic.

Shazeer N. Parmar N. Uszkoreit J. Jones L. Gomez A.N. Kaiser L. Vaswani, A. and I. Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. Von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. Rush. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv:1910.03771*.