Rumour Detection and Analysis on Twitter

1150027 **1114028** **1161977**

1 Introduction

Millions of people access and share information through social media such as Twitter, and such information can be quickly disseminated to a wider users without being verified. After the outbreak of COVID-19, related rumours increased through social media like Twitter, causing fear and anxiety. Soroush et al. (2018) found rumours to be more novel or distasteful/fearful than factual content, thus leading to rumours with high responsiveness and propagation. So the purpose of this report is to get a rumour detection model by comparing different models using Twitter data. And the best model is used to identify rumours from an unlabeled COVID-19 tweets, and to give an analysis of these rumours.

In Section 3, the training, development, and testing data used in this report will be explained.

In Section 4, statistical models and five neural network models based on different pre-trained models will be compared.

In Section 5, the rumours among COVID-19 tweets detected by the previous models will be characterized, including topic classification, sentiment analysis, etc.

2 Related Work

As early as 2011, Castillo et al. (2011) proposed an automatic rumour classifier based on machine learning, and this classifier mainly uses user information, retweet patterns and other features. Ma et al. (2015) proposed a method to identify rumours by focusing on the time series characteristics of tweets. Still, this approach spends a lot of effort on data pre-processing and feature engineering. After 2016, scholars started to use deep learning on text classification such as CNN, Bi-GRU (Riduan et al., 2021). In terms of rumour detection, Ma et al. (2016) were the first to implement RNN, and they used LSTM and GRU to get good performance on microblogs. Yu et al. (2017) argued that chronological models are not suitable for detecting rumours at an early step. Therefore, CNN is used in this paper to extract the characteristics of the rumour itself. Besides, Zhou et al. (2019) used a combination of RNN and reinforcement learning to detect rumours early. After Transformer (Vaswani et al., 2017) and BERT (Devlin et al., 2019) came out, Kaliyar et al. (2021) proposed a transfer learning using the BERT model combined with CNN, and in their experiments, the accuracy of text classification was as high as 98% .

The model architectures explored in this report are also based on the BERT and other pre-trained models for transfer learning.

3 Data and Problem definition

3.1 Raw Data

In this experiment, we crawled a total of 21,828 tweets through the Twitter crawler, of which the training and validation sets were 14,909 and 6,919, respectively. After stitching the main tweets with the replies in time series, we obtained 1277 training and 535 validation data, as shown in Table 1, with non-rumour tags accounting for about 80% of both the training and validation sets. Using unprocessed raw data, we counted the distribution of features in the training sample (Figure 1). We observed that non-Rumour tweets and replies generally had fewer characters and words and carried fewer URLs, Mentions and Stopwords. However, the distribution of Hashtags taken by non-Rumour and Rumour tweets is almost the same.

3.2 Preprocessing Method

The presence of slang, misspelled words, emojis, hashtags, and URLs in Twitter tweets creates a lot of noise in extracting text information from tweets. We pre-processed each tweet and its reply tweets, including 'cleaning', 'normalisation' and 'splicing', which we expect to positively impact the effective-

ness of our classification and sentiment analysis tasks(Pano and Kashef, 2020). The 'cleaning' manages the unwanted texts, numbers, mentions, URLs, and emoji in Twitter tweets, where mention and URL are defined as starting with '@' and 'HTTP '. In addition, emoji symbols, stopwords provided by NLTK, hashtags that start with '#', words that are too short, and words that do not contain the alphabet were removed in pre-processing.
We used WordNetLemmatizer, a standard text normalisation method provided by NLTK, to reduce the noise effect of different word forms in sentiment extraction by removing word affixes through word form reduction and extracting the central part of the word(Gupta and Joshi, 2017).

	Total	Rumour	Not Rumour
Train	1277	253	1024
Dev	535	115	420
Test	558	NaN	NaN

Table 1: Train, dev and test labels

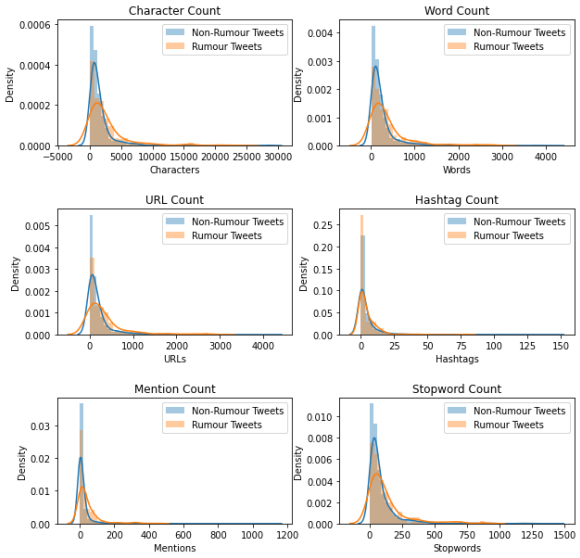

Figure 1: distribution plot of Training data

3.3 Problem Definition

We stored the pre-processed source tweet text and its retweet texts as a combination referred as an *event*. So the whole input data contains a set of events $E = \{e_1, e_2, \cdots, e_N\}$, where N is the total number of tweet combinations. In each event e_i, it contains n tweets c_j, as $e_i = \{c_0, c_1, c_2, \cdots, c_n\}$, where c_0 is a source tweet, and rest of them are ordered chronologically to retain semantic order. In each c_j, it contains m tokens t, as $c_j = \{t_{j1}, t_{j2} \cdots t_{jm}\}$. We concatenate all tweet texts c_j in to one text "$c_1 c_2 \cdots c_n$". And it will be the initial input data format among all the models.

4 Model Architectures

4.1 Baseline Approach

I. Bernoulli Navie Bayes with TF-IDF
Using TF-IDF to get the weight of each word to represent the importance of the word in $c_1 c_2 \cdots c_n$, and this weight will replace the number of occurrences of each word as the input to the Bernoulli Naive Bayes model. This model unexpectedly has high accuracy and precision but low recall and F1 score.
II. BERT with One Fully Connected Layer
To make the input data conform to BERT's format, we add special tokens $[CLS]$, $[SEP]$ and $[PAD]$ to "$c_1 c_2 \cdots c_n$". And set max length of the input to 256, since the number of tokens in e_i which less than 256 accounted for 88%.
The vector of $[CLS]$ token of the BERT is extracted to do the classification since it can represent the whole input (Devlin et al., 2019). Classification based on pre-trained models significantly improves performance compared to the previous statistical models.

4.2 GRU

The core idea of this method is that since the data is chronological, using the sequential model to take every word embedding as an input for each time step should have a better result. Besides, changing a suitable pre-trained model may also be beneficial. As a result, this and the following models in the report will use the latest RoBERTa fine-tuned with 128M tweet data (Liu et al., 2019) as the basis for transfer learning, which has better performance than BERT on emotion and sentiment (Loureiro et al., 2022).
We choose Gated Recurrent Unit (GRU) as the sequential model. GRU replaces the forget and input gates in LSTM with update gates and combines cell and hidden states to reduce the complexity of the architecture. The performance difference between the two is often not significant (Chung et al., 2014). The RoBERTa with GRUs model's diagram is shown in Figure 2:
In Figure 2, t_{ij} means j^{th} token in c_i and V_{ij} is word embedding corresponding to it. Special tokens <s> and </s> are corresponding to $[CLS]$ and $[SEP]$ in BERT.

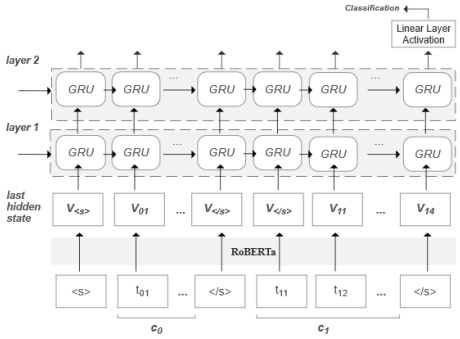

Figure 2: RoBERTa with two layers GRUs

4.3 Vectors Representing Sentences with GRU (SGRU)

Reimers and Gurevych (2019) and Yang et al. (2016) both proposed methods to extract sentence level embedding. Based on their ideas, this method is to extract the special token </s> vector of the input and put them into a sequential model GRU. However, considering the different number of tweets c in each event e, the input lengths are a problem when passing the data into GRU as a batch. GRU Cell in PyTorch is used to solve this problem, which is a model that expands the original recursive input into separative input (Figure 3).

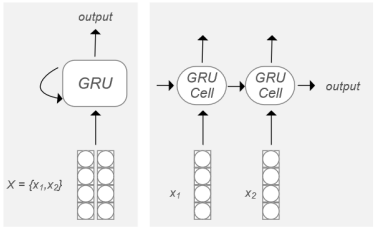

Figure 3: Left: Original GRU; Right: GRU Cell

Like the tokenization method mentioned in section 4.1, we directly selected every </s> vector in the input data as the representation for each c_i. Equally, the sentence vector can also be obtained by summing or averaging the vectors of tokens within that sentence (Figure 4). Then we use each extracted sentence vector as the input to GRU Cell.

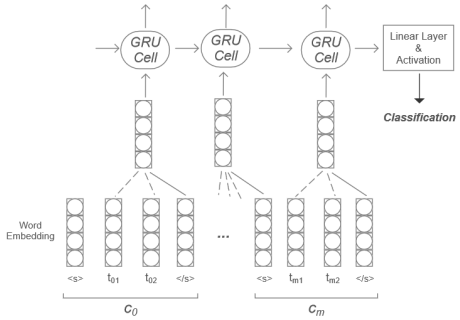

Figure 4: Diagram of vectors representing sentences

4.4 CNN

Since previous models are all related to sequence, the main idea of this method is to compress the input using CNNs and find the focused information in the input. This method uses three CNNs with kernel 3, 4, 5 and 100 filters each to extract the important information at different levels after word embedding. The three results are maxpooled and concatenate into a vector that is used as the input for the final discriminant function (Figure 5).

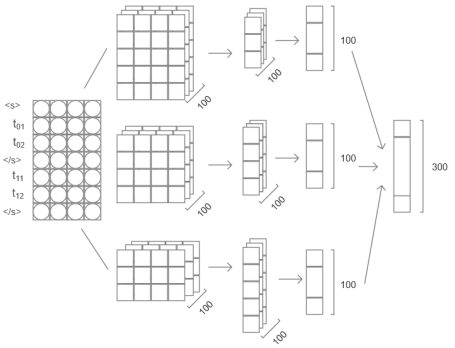

Figure 5: Multiple CNNs

4.5 Sentence Pairing (SPair)

The idea of this method is mainly to mimic the thinking process of humans when determining whether a tweet is a rumour. When a human knows nothing about a tweet, he/she cannot judge it from the source tweet alone. However, when there is a retweet with a different viewpoint from the source tweet, humans may wonder if the tweet is real. Therefore, we propose a sentence pairing algorithm, which focuses on finding out whether a contradiction exists between the source tweet and each of its retweets by concatenate them together. The architecture of the model is shown in Figure 6.

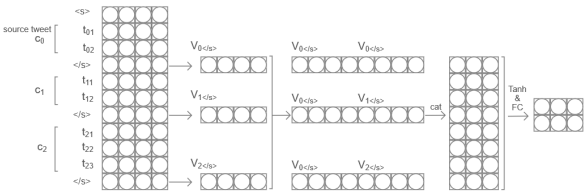

Figure 6: Sentence pairing model

4.6 Experiments and Results

The experimental results on the development dataset are shown in Table 2, and the results on partial test dataset are shown in Table 3. Vectors representing sentences with GRU has the best result, which achieve F_1 0.892 on development dataset. Sentence pairing model got the best F_1 0.894 on partial test dataset.

Model	Accuracy	Precision	Recall	F_1
MNB	0.869	**1.0**	0.391	0.563
BERT	0.938	0.868	0.839	0.854
GRU	0.935	0.866	0.845	0.857
SGRU	**0.953**	0.881	**0.904**	**0.892**
CNN	0.944	0.912	0.817	0.862
SPair	0.946	0.906	0.835	0.869

Table 2: Results on Twitter Development Dataset

Naive Bayes achieves precision 1.0, which means that when the model judges a rumour, it must be a rumour. This indicates that a certain type of words in a rumour always has a higher weight. Single BERT performed very well. However, the overall performance of GRU is poor, which may due to the long time steps, resulting in the source tweet having been forgotten in the final output.

Vector representing sentence with GRU reduces the length of time steps, so there is a great improvement. While the effect in CNN is not obvious. It indicates that the information extracted by CNN is more ambiguous compared to the one extracted by the sentence vector directly. The performance of the sentence pair algorithm will be slightly higher than that of CNN. The reason may be that some important features of the rumour are intercepted when RoBERTa is processing the sentence length.

Taking CNN and Sentence Pair as examples, the models based on pre-trained are suffering overfitting. Although the penalty factor has been increased, it still cannot be solved (Figure 7). It is very necessary to fine tune the hyper-parameters and sample the data later, but this report is mainly for the comparative analysis of the models.

	MNB	BERT	GRU	SGRU	CNN	SPair
F_1	0.571	0.821	0.877	0.860	0.835	**0.894**

Table 3: Results on Twitter Test Dataset

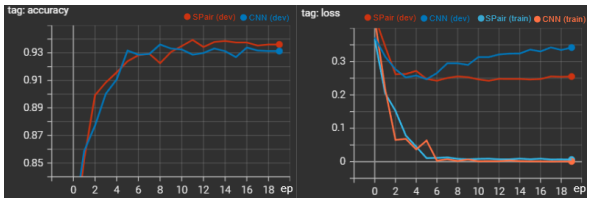

Figure 7: Loss and Accuracy of CNN / Sentence Pair

5 COVID-19 Rumour Analysis

5.1 COVID-19 Classification and Sentiment

We use the Sentence Pairing Model which got the highest F_1 in Kaggle to do the classification. And we got 1911 rumour tweets and 14045 non-rumour ones.

0.96% of rumour tweets have non-neutral sentiment, and 0.36% of non-rumours are non-neutral. Of these non-neutral tweets, rumour tweets accounted for 76.92% of negative sentiment, but only 62.26% of non-rumour tweets. This indicates that more negative sentiment will be present in rumours.

5.2 Text WordCloud and Top Words

WordCloud Python library is used to generate the COVID-19 rumour and non-rumour tweets' wordcloud images (Figure 8, Figure 9). WordCloud can visually display keywords that frequently appear in the text, but filter out the low-quality text information(Heimerl et al., 2014).

We counted the top 10 frequent words of rumour and non-rumour corpus respectively (Figure 10). By working with WordCloud and the top 10 most frequent words, we can find that there are a lot of similar words in the two corpora, such as coronavirus, covid-19, Trump, say, case, new, test, case, people. These frequently-occurring words obtained can help us consider these words as few as possible as representative words for the generated topic in the following modelling analysis. It is worth mentioning that when counting the top 10 words of the two corpora, it is found that there is a unique word "pandemic" in the top 10 words of rumour.

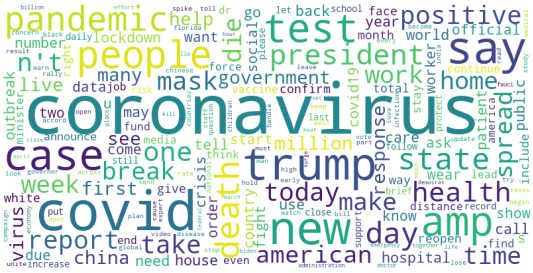

Figure 8: COVID-19 rumour corpus

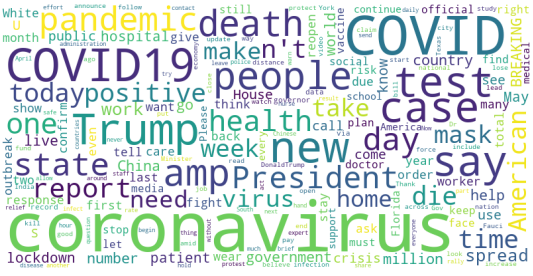

Figure 9: COVID-19 non-rumour corpus

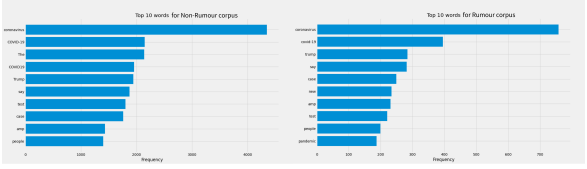

Figure 10: Top 10 words frequent for COVID-19 corpus

5.3 COVID-19 N-Gram models

We used an n-gram model to count the frequency of various combinations of words in general and covid tweets. We analyzed the difference between covid-related and general tweets by looking at the top 20 occurrences of the words. In Figure 11 bigram model, we can observe ron-rumour tweets in general tweets, with about 10 out of 20 tweets about covid, including some prevention and treatment-related tweets, while in the Top 20 rumour tweets, we can observe: among other miscellaneous tweets, that 'Florida woman' is also represented in other entries. For bigram's covid tweets, non-rumour and rumour's top 20 entries are very similar in terms of distribution and ranking. In addition to the usual covid-related prevention, treatment and intelligence, government-related entries also rank relatively high, such as 'president trump ', 'white house', 'donald trump' and 'unite state'. In addition, the rumour tweets for 'death toll', 'spread coronavirus', 'trump say' are also relatively more frequent.

The trigram model(figure 11) behaves similarly to the bigram model. some keywords such as 'trust n't trust' and 'chinese communist party' in the Covid-related rumour trigram model can also help us better understand the purpose and semantics of the rumour tweets.

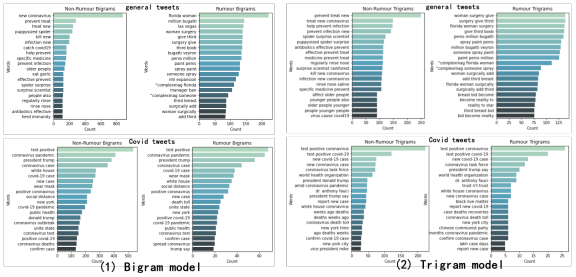

Figure 11: Bigram(1) and Trigram(2) model for General tweets and Covid tweets

5.4 Topic Modeling

For the topic modelling part, we adopt the Latent Dirichlet Allocation (LDA) model (Blei et al., 2003) on the rumour and non-rumour COVID-19 corpus respectively. The LDA model is a classic topic modelling model. It is generally used to analyse topics of large corpora and a large amount of document information. LDA is an unsupervised learning algorithm, so it does not require the time-consuming manual labelling of data. Besides, LDA has been used in many Twitter-related pieces of research and made a significant contribution. For model building, we use the Gensim Python library. We set the parameters of the model to num_topics=10, passes=10.

It is worth mentioning that before the text is sent to the LDA model, the TF-IDF algorithm is used to convert the input one-hot data and then send it to the LDA model. Figure 12) is the results of topic modeling.

Rumour	
Topics	**Keywords**
COVID-19 and job related topic	coronavirus, job, crore, people, years, day, go, work
Mask and health related topic	mask, wear, face, health, public, hospital, medical, without
Fail and president related topic	Trump, say, fail, coronavirus, president, spread, curve, new, leave, lockdown
Death case and total number report related topic	case, deaths, new, total, report, number, record, confirm, recoveries, daily, today, continue, death
People trust related topic	trust, say, people
Coronavirus and pandemic related topic	COVID-19, pandemic, end, live, much, case, world, global, risk
Test and positive related topic	test, amp, lakh, COVID-19, positive, get, first
Test positive and numbers of hours die at home related topic	coronavirus, test, people, say, positive, result, hours, die, home, deaths

Non-Rumour	
Topics	**Keywords**
Drug and trump and biden related topic	drug, Trump, Biden, hydroxychloroquine, virus, Joe
Bill related topic	bill, need, Americans, workers, NHS, pay
China and mask related topic	China, mask
Social distance and policy related topic	social, distance, help, policy, work
People names related topic	Pence, Florida, South, sample, Mike, lab, DeSantis
Death and numbers related topic	case, deaths, new, report, number, coronavirus, weeks, dead, record, total, ago, days, confirm
White House related topic	White, Trump, positive, House, Tulsa, Johnson, infect
Patients and ICU related topic	hospital, patients, know, die, help, ICU
United States and Vaccine related topic	United, States, President, million, people, vaccine

Figure 12: Topic modeling results for COVID-19 corpus

6 Conclusion

In this report, we explored various text classification strategies, including GRU, SGRU, CNN and SPair, based on the baseline Bert model and Naive Bayes, focusing on the impact of different model structures on the Twitter text classification task. We classify COVID-19 related tweets using a classifier and use a variety of text models for feature extraction and analysis. In addition, we has conducted topic modelling for the COVID-19 Rumour, and Non-Rumour datasets, successfully analyzed 16 topics and generated corresponding WordCloud images for the two datasets. A certain type of word in a rumour always has a high weight, but when the topic is restricted to COVID-19, the weighted features of the text are diluted. The algorithm results often depend on how well the model extracts essential information from the sentence. The way the text data is pre-processed and the input vectors are stitched together is often crucial. Simple text stitching stretches the weight of noise and limits the model's feature extraction. In contrast, its time-series related features have great potential to improve model performance and refine feature weights. In the future, we will try to use GloVe to implement embedding layer and build Transformer architecture from scratch to implement hierarchical model. Besides, we need to have more GPU memory (not training on colab) to train and fine tune our model.

7 Contribution

1114028 explored text data pre-processing methods, EDA, feature distribution and pre-processing

of tweets, and participated in exploring baseline classification models. Completed pre-processing of Covid-related tweets, distribution analysis and n-gram models.
1161977 participated in the discussion of model selection, helped to find pre-training models, and was responsible for topic modeling, word cloud generation and top words visualization work.
1150027 pre-processed data, exploring different pre-trained models and text classification models. Mainly responsible for building, training models and making diagrams of them.

In general, each team member have made their efforts for the project.

References

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. 2011. Information credibility on twitter. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, page 675–684, New York, NY, USA. Association for Computing Machinery.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Itisha Gupta and Nisheeth Joshi. 2017. Tweet normalization: A knowledge based approach. In *2017 International Conference on Infocom Technologies and Unmanned Systems (Trends and Future Directions) (ICTUS)*, pages 157–162.

Florian Heimerl, Steffen Lohmann, Simon Lange, and Thomas Ertl. 2014. Word cloud explorer: Text analytics based on word clouds. In *2014 47th Hawaii international conference on system sciences*, pages 1833–1842. IEEE.

Rohit Kumar Kaliyar, Anurag Goswami, and Pratik Narang. 2021. FakeBERT: Fake news detection in social media with a BERT-based deep learning approach. *Multimedia Tools and Applications*, 80(8):11765–11788.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Daniel Loureiro, Francesco Barbieri, Leonardo Neves, Luis Espinosa Anke, and Jose Camacho-Collados. 2022. Timelms: Diachronic language models from twitter.

Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J. Jansen, Kam-Fai Wong, and Meeyoung Cha. 2016. Detecting rumors from microblogs with recurrent neural networks. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, page 3818–3824. AAAI Press.

Jing Ma, Wei Gao, Zhongyu Wei, Yueming Lu, and Kam-Fai Wong. 2015. Detect rumors using time series of social context information on microblogging websites. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, CIKM '15, page 1751–1754, New York, NY, USA. Association for Computing Machinery.

Toni Pano and Rasha Kashef. 2020. A complete vader-based sentiment analysis of bitcoin (btc) tweets during the era of covid-19. *Big Data and Cognitive Computing*, 4(4):33.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks.

Gusti Muhammad Riduan, Indah Soesanti, and Teguh Bharata Adji. 2021. A systematic literature review of text classification: Datasets and methods. In *2021 IEEE 5th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*, pages 71–77.

Vosoughi Soroush, Royand Deb, and Aral Sinan. 2018. The spread of true and false news online. *Science*, 359(6380):1146–1151.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, San Diego, California. Association for Computational Linguistics.

Feng Yu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2017. A convolutional approach for misinformation identification. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 3901–3907.

Kaimin Zhou, Chang Shu, Binyang Li, and Jey Han Lau. 2019. Early rumour detection. In *Proceedings of the 2019 Conference of the North American Chapter of*

the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 1614–1623, Minneapolis, Minnesota. Association for Computational Linguistics.