B. TECH. PROJECT REPORT

On

Rumour Detection in Social Network

BY

Nimish Bansal **Shivesh Dave**

DISCIPLINE OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

November 2022

Rumour Detection in Social Network

A PROJECT REPORT

Submitted in partial fulfillment of the

*requirements for the award of the degrees**of***

BACHELOR OF TECHNOLOGY

in

DISCIPLINE OF COMPUTER SCIENCE AND ENGINEERING

Submitted by:

NIMISH BANSAL **190001040**

SHIVESH DAVE **190001057**

Guided by:

Dr. Abhishek Srivastava, Professor, Department of Computer Science and Engineering

INDIAN INSTITUTE OF TECHNOLOGY INDORE

November 2022

Candidate’s Declaration

We hereby declare that the project entitled “RUMOUR DETECTION IN SOCIAL NETWORK” submitted in partial fulfillment for the award of the degree of Bachelor of Technology in Computer Science and Engineering completed under the supervision of Dr. Abhishek Srivastava is an authentic work.

Further, we declare that we have not submitted this work for the award of any other degree elsewhere.

Nimish Bansal	Shivesh Dave
190001040	190001057
28/11/2022	28/11/2022

Name of student with date

Preface

This report on “**Rumour Detection in Social Networks**" is prepared under the guidance of **Dr. Abhishek Srivastava, Professor,Department of Computer Science and Engineering**.

SHIVESH DAVE

B.Tech. IV Year

Discipline of Computer Science and Engineering.

IIT Indore

NIMISH BANSAL

B.Tech. IV Year

Discipline of Computer Science and Engineering.

IIT Indore

Acknowledgement

We wish to thank **Dr. Abhishek Srivastava, Professor, Department of Computer Science and Engineering** for their kind support and valuable guidance and for giving me this opportunity to work in this interesting domain. Without their support this report would not have been possible.

We are really grateful to our respected **HOD Dr. Somnath Dey**, **DUGC Convener Dr. Bodhisatwa Mazumdar**, and all the respected faculty members of the **Department of Computer Science and Engineering , IIT Indore** for their kind approval and guidance, without which we wouldn’t have been able to pursue this opportunity.

We would also like to thank my friends and family who provided constant encouragement and enabled me to perform at the best of my abilities.

Shivesh Dave

B.Tech. IV Year

Discipline of Computer Science and Engineering.

IIT Indore

Nimish Bansal

B.Tech. IV Year

Discipline of Computer Science and Engineering.

IIT Indore

Abstract

Social media's influence on how people form opinions has broad ramifications for all facets of society. Despite offering opportunities for sharing news and opinions, social media platforms like Twitter and Facebook have such high post quantities that it is difficult to regulate the quality or credibility of messages. Misinformation and rumours have a long-lasting impact on society because they tend to shape people's opinions and often also lead to disastrous actions in the heat of the moment. Therefore, it is crucial to identify and delete rumours from these networks. Given the numbers, only automatic identification and classification of social media posts based on veracity can stop the spread of rumours. We have chosen to concentrate on Twitter’s data in this research given the simplicity of obtaining it. Our methods use feature extraction to extract context and content features from tweets' text in order to distinguish between rumours and non-rumors. According to some experts, tweets contain a linguistic pattern that can be utilized to identify rumours on Twitter. Manually extracting features takes time. By employing language embedding with BERT to identify rumours on Twitter rather than the more traditional feature extraction techniques, we suggest a novel solution to this problem. BERT is a unique transformer that can interpret words in their context. To represent each tweet's sentences into a vector in accordance with the context of the tweet, we employ sentence embedding using BERT. We use a variety of supervised learning approaches to divide those vectors into rumour- and non-rumor- containing categories.

Table of Contents

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

Absence of sufficient control over the contents of posts on social media like Twitter and Facebook gives rise to an unprecedented spread of false information. Users post, share and engage with posts on breaking news without verifying the authenticity of their content or sources. These unverified rumours have been shared thousands of times, causing readers to experience stress and irrational conduct. People all across the world share news, ideas, and experiences via microblogging platforms like Facebook and Twitter. They are becoming the new go-to for news and information because of their explosive growth in popularity [1]. Short communications, or "tweets," can be posted and exchanged on Twitter. Tweets are circulated to the author's followers and are simple to "re-tweet." Twitter has recently been regarded as the microblogging site most frequently utilized as a news source [2], [3]. The majority of the news on Twitter originates from those who are public. The lack of supervision on Twitter makes it a good place for rumours to spread. Numerous studies have shown that Twitter has a ton of fantastic content, especially during high-profile events [4] – [6]. Additionally, rumours have negative consequences on society. For instance, the 2013 Twitter post "Explosion at White House and Obama Injured" caused the stock markets to plummet and caused investors to lose roughly 136 billion dollars in just two minutes [4]. Approximately 86% of the fabricated tweets during Hurricane Sandy, according to research by Gupta et al. [7], were retweeted. As a result, it's important to automatically identify rumours and determine whether material provided on Twitter is reliable because manually fact-checking information is time-consuming and extremely difficult.

1. Definition of Rumour

The Merriam-Webster Dictionary defines a rumour as “a currently circulating story or report of uncertain or doubtful truth”. We use a similar definition, stating that they are unconfirmed at the time of posting. This uncorroborated information could be true, partially or completely false. It could also go unresolved. Throughout this study, we use the commonly accepted definition of a rumour, which is "an item of circulating information whose veracity status is yet to be verified at the time of posting." (we will refer to our research paper here as we are directly taking their definition). An unverified rumour is one which has no evidence supporting it, or one which has not been officially confirmed by authoritative sources or credible sources owing to the context of the information.

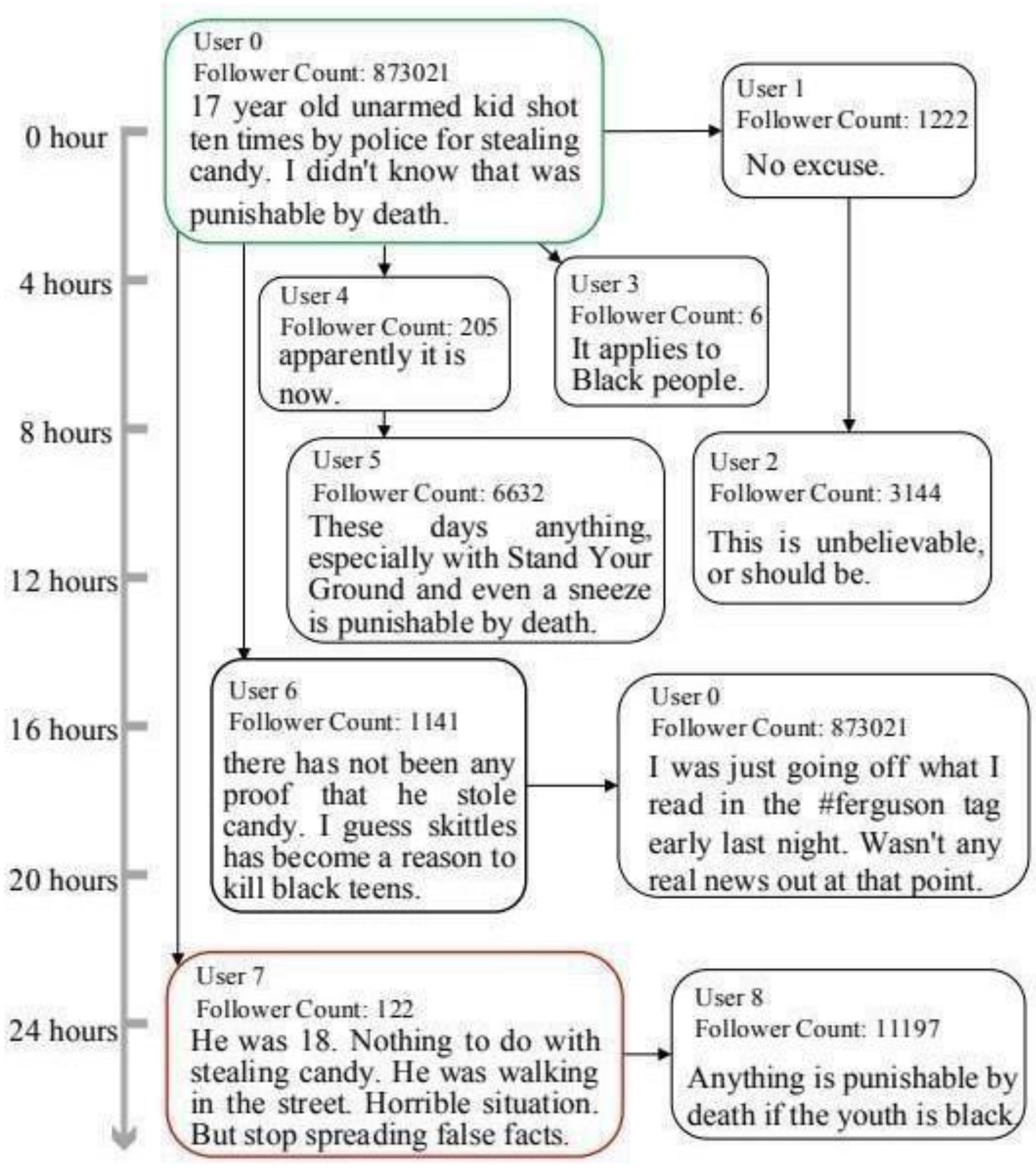
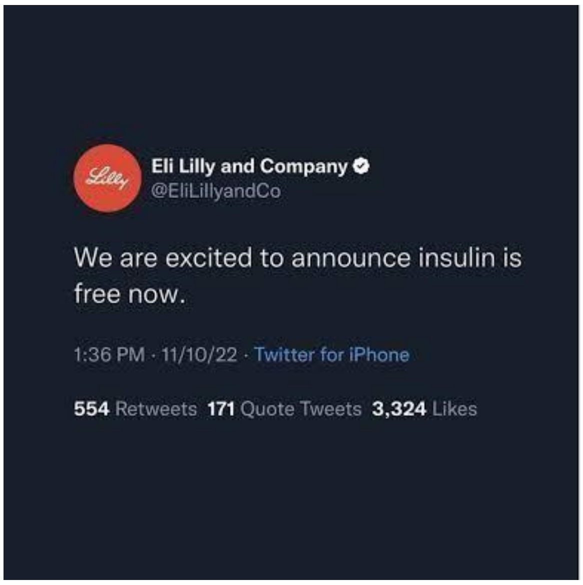

0 hour
4 hours
8 hours
12 hours
16 hours
20 hours
24 hours
User 0
Follower Count: 873021
17 year old unarmed kid shot ten times by police for stealing candy. I didn't know that was punishable by death.
User 1
Follower Count: 1222
No excuse.
User 4
Follower Count: 205
apparently it is now.
User 3
Follower Count: 6
It applies to Black people.
User 5
Follower Count: 6632
These days anything, especially with Stand Your Ground and even a sneeze is punishable by death.
User 2
Follower Count: 3144
This is unbelievable, or should be.
User 6
Follower Count: 1141
there has not been any proof that he stole candy. I guess skittles has become a reason to kill black teens.
User 0
Follower Count: 873021
I was just going off what I read in the #ferguson tag early last night. Wasn't any real news out at that point.
User 7
Follower Count: 122
He was 18. Nothing to do with stealing candy. He was walking in the street. Horrible situation. But stop spreading false facts.
User 8
Follower Count: 11197
Anything is punishable by death if the youth is black

Figure 1 An illustration of a rumour propagating on TWITTER. The green box indicates the source message, and the red box highlights a post that rebuts the rumour.

2. Why is Rumour Detection required?

Social media's influence on how people form opinions has broad ramifications for all facets of society. Despite offering opportunities for sharing news and opinions, social media platforms like Twitter and Facebook have such high post quantities that it is difficult to regulate the quality or credibility of messages. Misinformation and rumours have a long-lasting impact on society because they tend to shape people's opinions and often also lead to disastrous actions in the heat of the moment. Therefore, it is crucial to identify and delete rumours from these networks.

We can understand this with the help of an example.

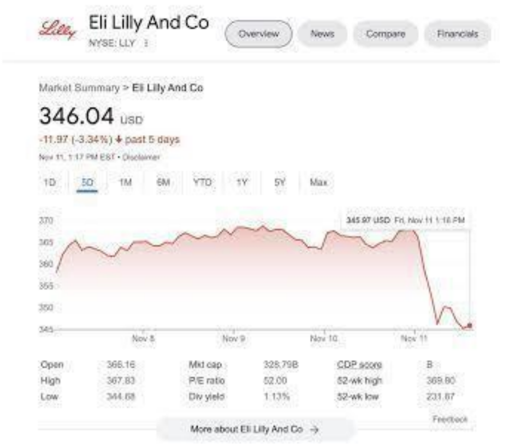


Eli Lilly and Company
@EliLillyandCo

We are excited to announce insulin is free now.

1:36 PM · 11/10/22 · Twitter for iPhone

554 Retweets 171 Quote Tweets 3,324 Likes



Eli Lilly And Co
NYSE: LLY

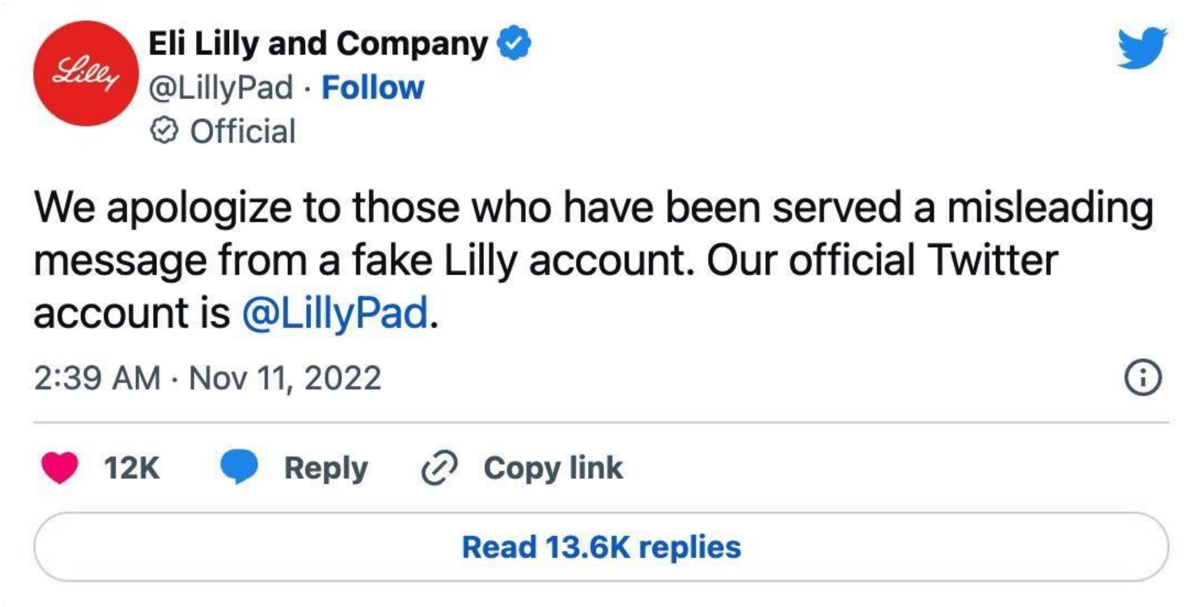
Market Summary > Eli Lilly And Co

346.04 USD

-11.97 (-3.34%) past 5 days

Nov 11, 1:17 PM EST · Disclaimer

Open	366.16	Mkt cap	328.79B	CDP score	B
High	367.83	P/E ratio	52.00	52-wk high	369.80
Low	344.68	Div yield	1.13%	52-wk low	231.87



Eli Lilly and Company
@LillyPad · Follow
Official

We apologize to those who have been served a misleading message from a fake Lilly account. Our official Twitter account is @LillyPad.

2:39 AM · Nov 11, 2022

12K Reply Copy link

Read 13.6K replies

Figure 2 Impersonification of Eli Lilly on Twitter

Eli Lilly (LLY) fell 4.37 percent Friday to US $352.30, wiping out more than $15 billion in market capitalization, after a Twitter Blue verified account impersonating the brand promised free insulin.

3. Studies of Rumours

Reliable feature extraction is used in the majority of current studies on rumour detection on Twitter. These researches [8], [9], [10]– [12] manually extract features from the context and content of tweets in order to classify the authenticity of tweets using supervised learning algorithms. The accompanying information from tweets, such as user and network information, is included in the context-based feature components [8], [10] – [13]. As opposed to the content-based feature components, which entail extracting features from tweets' text, notably from linguistic elements including lexical, syntactic, and semantic information [1], [10] – [13]. However, feature extraction frequently requires additional data that isn't always accessible on Twitter [14].

Deep learning techniques have recently become more popular for spotting rumours in tweets. These techniques are advantageous for effective rumour identification because they can use the deep data representation from the text and automatically uncover features. Recurrent neural networks (RNN) [16, 17], convolutional neural networks (CNN) [16, 17], and long short-term memory (LSTM) networks [18, 19] are a few of these methods. Reinforcement learning has been suggested in some studies [20] as a way to enhance rumour detection. However, for automatic rumour detection to be a useful technique, deep learning model performance must be increased.

4. Brief idea of our approach

We demonstrate an innovative method for determining the veracity of a tweet by using the sentence embedding feature of the Bidirectional Transformer Encoder Representation (BERT). This revolutionary transformer technique BERT can capture contextual meanings of a word by considering phrases from both the left and right sides of a word [21]. In order to identify rumours on Twitter, we use a number of classifiers and BERT to represent each tweet as a vector based on the context of the sentence. We found that, in terms of results, our suggested strategy exceeds the accuracy of most recent state-of-the-art techniques.

CHAPTER 2

RELATED WORK

So far most of the existing studies have used supervised learning models to classify tweets into rumorous and non-rumorous. Such feature-based approaches fall into two broad categories: context and content-based approaches. Figure 3 shows the most common strategy that has been employed in state-of-the-art approaches.

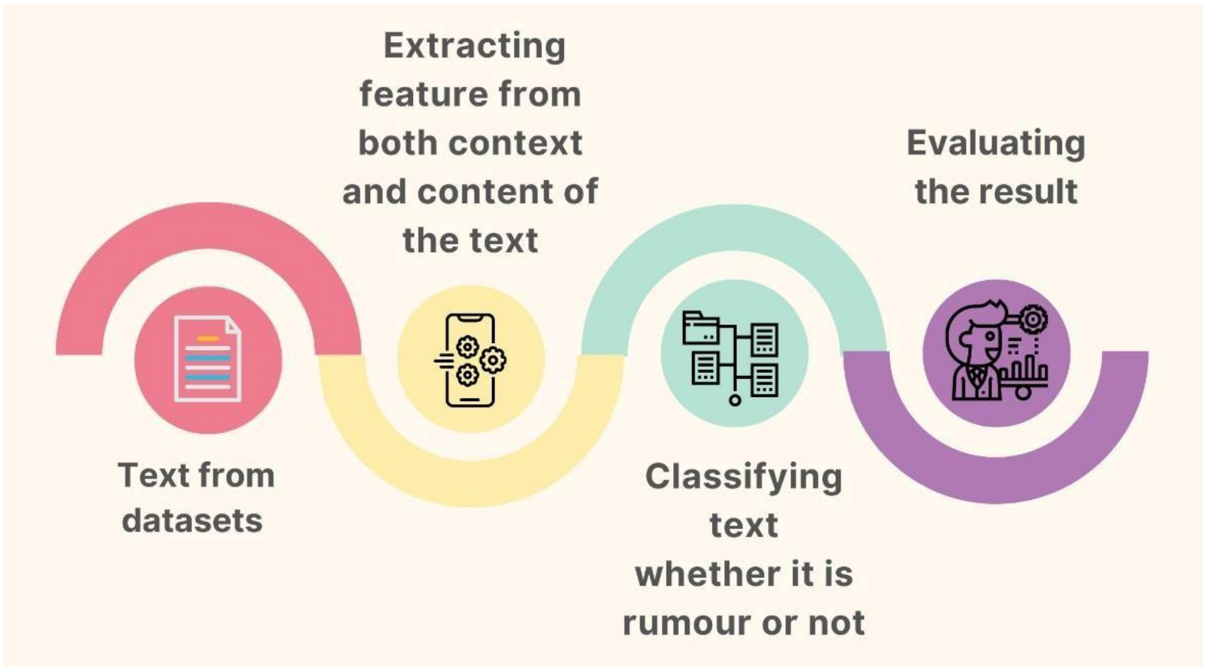

Extracting feature from both context and content of the text
Evaluating the result
Text from datasets
Classifying text whether it is rumour or not

Figure 3 General architecture of rumour classification models

According to Qazvinian et al., [27], traditional rumour detection research has mostly concentrated on creating custom characteristics for machine learning systems. Using cue phrases and tweet statistics, Takahashi and Igata [29] suggested a method for rumour identification on Twitter. Yu et al., [27] employed some features based on location and customer, two novel categories of features, to detect rumours on Sina Weibo. However, these studies have only found rumours that were already known. What happens is that a classifier takes a bundle of pre-established rumours (for example, “Obama is Muslim”), next it categorizes upcoming tweets on having links to one of the pre-established rumours or not (for example, “I believe Obama is not Muslim is related to the rumour, but Obama was speaking to a group of Muslims is not"). Such an approach may be a success for long-lasting rumours; this is called rumour tracking. However, this is a failure in fast scenarios like live or breaking news, where newly discovered rumours emerge. The relevant words associated with these newly discovered rumours are not known in this case. First the classifier needs to be trained on this newly detected fake news dataset to learn patterns that will help in further fake news detection.

So far, just one study has tried to address the issue of discovering new rumours (Zhao et al., [22]). Their strategy is based on the idea that some doubtful users will reply or tweet back to the rumoured tweets in an attempt to check their truthfulness; the presence of numerous questioning tweets in relation to a piece of information would thus heavily imply that the source material is rumoured. They developed a carefully organized set of “five regular expressions” that are used to recognize tweets that are being queried, such as "is (that | this | it) true." These questioning tweets are then grouped together based on similarities, with each group being ultimately classified as a candidate rumour. The authors' best method produced 52% and 28% precision for two datasets, however they were unable to evaluate by recall. We anticipate three potential limitations despite the fact that this study relies on a reasonable premise and proposes a sophisticated strategy for solving the rumour detection task: (1) The approach may not generalize well because it is based on manually organized data, (2) the afore-mentioned hypothesis may not always be accurate and lead to lower recall since some rumours may not be judged or questioned (Zubiaga et al., [8]), and (3) It disregards the pre-rumour context which is/can be helpful in finding whether the tweet is a rumour or not.

For automatic detection of fake news from a tweet without knowing the topic or having any context, Ajao et al., [18] suggested using both LSTM and CNN. They attained 0.82 and 0.437 respectively for accuracy and precision. Similar to this, Kotteti et al., [13]'s goal was to speed up rumour detection using supervised learning algorithms. They proposed a conceptual framework for a series of time related data analysis to lessen the complexity of computation by focusing solely on the contextual features of tweets rather than their contents, which necessitate careful feature selection and text mining. Their suggested approach utilized the Gaussian Naive Bayes classifier to achieve 0.94 precision score while drastically reducing computing cost. However, their results as shown in Table 1 are not that impressive.

Topic-Driven Novel Detection (TDRD) was suggested by Xu et al., [24] to assess a tweet's reliability only based on its microblog source. According to communication theory, the topic of a tweet can assist decide whether or not it is a rumour, according to the study [25], [26]. Pheme dataset as well as Weibo dataset was used to train the model classifier. They used Convolutional Neural Network jointly with FastText for the Twitter and Weibo datasets, respectively. The research showed a 0.816 accuracy rate.

Other researchers have worked on analogous tasks; however, they are not specifically connected to rumour detection. There was some research by Qazvinian et al, [27] and Lu et al., [28] to determine how to classify the stance of tweets that discuss rumours by identifying whether they are confirming, denying, or raising doubts about them. A labelled collection of tweets was used to train a classifier to identify and classify the mindset seen in new tweets discussing rumours. However non-rumours have been completely ignored assuming that the dataset consists of only rumour tweets. Stance and veracity classification tasks are considered supplementary to the task on hand. For example, a classifier that determines the stance of rumour could be fed the set of rumours that act as input for a rumour detection system. To differentiate between rumours and other information, the first phase, is relatively unexplored, and most research focuses on the next.

The investigation by Zubiaga et al., [8] was another study that sought to evaluate the veracity of certain tweets. They tried to detect and alarm users to unverified posts. The dataset consists of 5802 tweets that were gathered during five breaking news items using the Twitter streaming API. CRF (Conditional Random Fields) was compared with some other sequential classifiers such as NB (Naive Bayes) and SVM (Support Vector Machines). According to some earlier studies [8], [10], and [32], semantic and linguistic traits are the most important indicators of rumour.

In our study, we use BERT to uncover the unique linguistic patterns of a tweet and extract the contextual meaning of its sentences. We suggest a model utilizing BERT to detect rumours in tweets and improve upon the existing work in this field.

CHAPTER 3

RESEARCH GAPS

Many efforts have been made in recent years to increase the dependability and credibility of online content, but some of the most important issues are still unresolved. The existing research deficit and potential future research directions are highlighted in the section that follows. To stop the unprecedented spread of fake news and lessen its detrimental effects on society, quick and real-time source detection is helpful. Real-time dataset collection, automatic rumour detection, and locating the source of rumours are difficult problems. Some of them have been discussed below:

1. Automated Detection

It is a certainty that the classifier will make mistakes down the line, simply yielding a definite conclusion on the veracity of a rumour may not be adequate. Many have strived to automatically analyse the veracity of a rumour. We contend that a veracity classifier's output needs to be more detailed and explain the rationale behind the conclusion in order to be more trustworthy (Procter et al., [28]). A veracity classifier will be more trustworthy if it yields both the automatically calculated veracity rating along and the links to rumour classification sites which gives confirmation. Selecting some supportive and opposing viewpoints and feeding them to the stance classifier, will enhance the result of a veracity classifier. It is impossible to get a perfectly accurate veracity classifier. Thus more research should be done to obtain information sources which enables the end client to assess the rumors' truthfulness by themselves.

2. Cross-platform detection

Source identification in these situations can be challenging because people frequently disseminate rumours across their various social networks since they have accounts on numerous social networking platforms. Cross-platform dissemination and detection, along with the transfer of misleading information from one web community to another, has grown to be a serious difficulty for academics to track.

3. Real-time learning

Implementation of an net-based fact-checking interface that can understand and detect fraud in real-time from verified information sources.

4. Lack of unsupervised models

The majority of the work being done today employs supervised learning techniques. Due to the infinite amounts of unlabelled data from social media, unsupervised models must be constructed. The ability to learn from unlabelled data, which mitigates the challenge of locating and labelling pertinent data, is the fundamental benefit of utilising such techniques. Furthermore, these models might also help with a better comprehension of the issue and its main elements.

5. Datasets

Development of compelling verified datasets is crucial. The dearth of publicly accessible large- scale datasets prevents the benchmark comparison of various techniques.

6. Multilingual platform

Existing studies have focused majorly on linguistic features in English-language texts. Many popular and widely-used regional languages are not yet being implented into the multilingual platform for detecting fake news.

7. Complex network structure that is dynamic

If we perform the veracity classification task before it is resolved, it turns into a prediction problem that necessitates a substantial amount of supporting data. Because of the intricate and ever-changing network architecture of social networks, the problem becomes even more complicated.

8. Early detection

It is extremely difficult to identify fake news at an early level before it spreads widely so that prompt action may be made to mitigate it and intervene. Once bogus news has gone widely and earned users' confidence, it is nearly impossible to alter public opinion.

9. Cross-domain analysis

So far focus has mainly been on singular method of fake news detection, based on subject matter, dissemination, style, and so on. Cross-domain analysis, which considers factors such as languages, website, topic, images, and URL, aids in the identification of unique quasi-attributes and enables the quicky and precise rumour detection.

10. Multimedia false information detection

Data analytics, computer vision, and signal processing approaches must be developed to deal with created and altered sounds, images, and videos. Machine learning and deep learning algorithms are essential for finding the distinguishing features of modified and manufactured multimedia.

CHAPTER 4

OBJECTIVES

The main goal is to detect rumours automatically using BERT, a revolutionary transformer technique that can capture context - specific meanings of a word by considering terms from both the left and right sides of a word. To detect rumours on Twitter, we use a handful of classifiers and BERT to portray each tweet as a vector based on the context of the sentence. In regards to accuracy, we found that our suggested scheme surpasses the most recent state-of-the-art techniques. We use the source tweets, their reactions and also the user features as an input to our model.

CHAPTER 5

DATASETS

Developing a method to get a sizable set of news that would not be known before and hence untrained data consisting of both rumours and non-rumours was one of our primary goals while preparing to compile a dataset of rumours and non-rumours. We learnt from the reactions to the situation which was set up wherein a user is following stories related to breaking news. A user would mark each tweet as a rumour or a non-rumour after viewing a timeline of tweets concerning breaking news. Gathering already known rumours will help us search for specific keywords containing tweets such as “London Eye Fire” that leads us to extract tweets spreading the rumour that the London Eye has been set on fire. This method was used during the 2011 England Riots. However, this assumes that the rumour in question is known beforehand and will not find rumours connected to occasions for which particular keywords have not yet been determined. We use the Twitter streaming API to extract tweets about noteworthy events that can lead to the start and spread of rumours as part of our data collection strategy.

We used the procedure outlined above for five distinct significant occurrences, all of which received a lot of media attention and were riddled with rumours:

Ferguson unrest: On August 9, 2014, a white police officer fatally shot an 18-year-old African American man named Michael Brown, following which residents of Ferguson, Michigan, protested.

Ottawa shooting: On October 22, 2014, gunfire erupted on Parliament Hill in Ottawa, Canada, claiming the life of a Canadian soldier.

Sydney siege: On December 15, 2014, a shooter held eight employees and 10 customers at a Lindt chocolate cafe in Sydney, Australia, hostage.

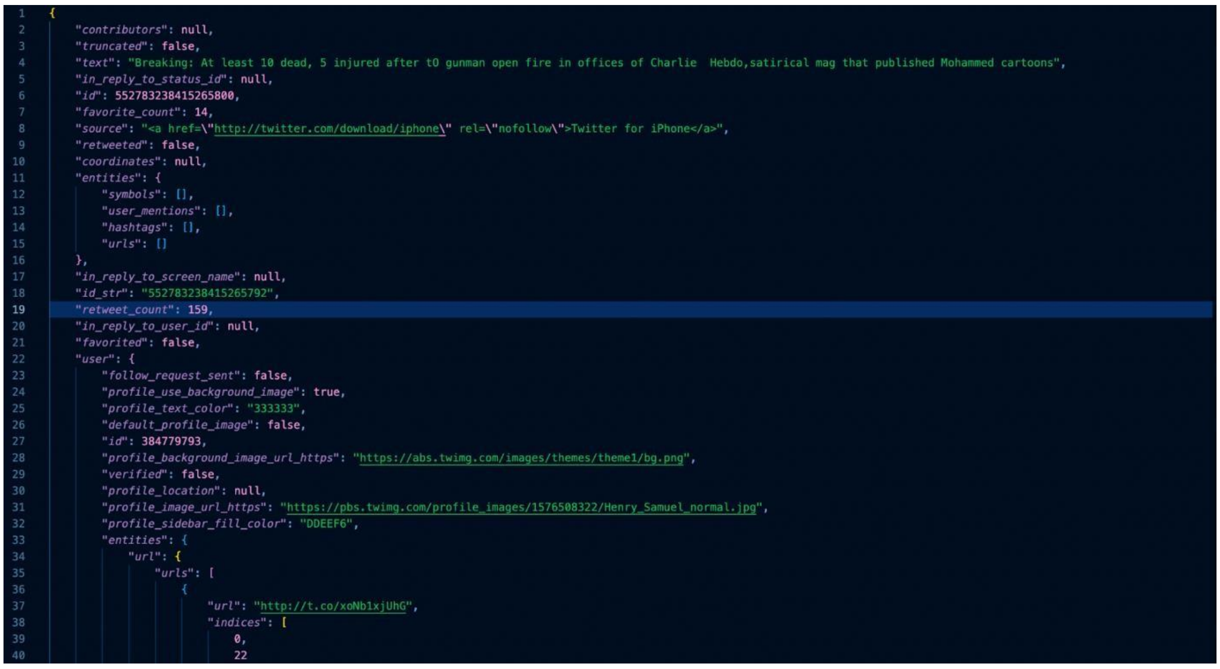
A total of 5,802 tweets were annotated for the five events, with 1,972 of those tweets being classified as rumours and 3,830 as non-rumours. According to Table 1, these annotations are allocated differently throughout the five events. Less than 25% of tweets about Charlie Hebdo and Ferguson were rumours, compared to slightly more than 50% of tweets about the Germanwings Crash and the Ottawa Shooting. The percentage of rumours during the Sydney Siege was in the middle (42.8%).

1. Data Extraction

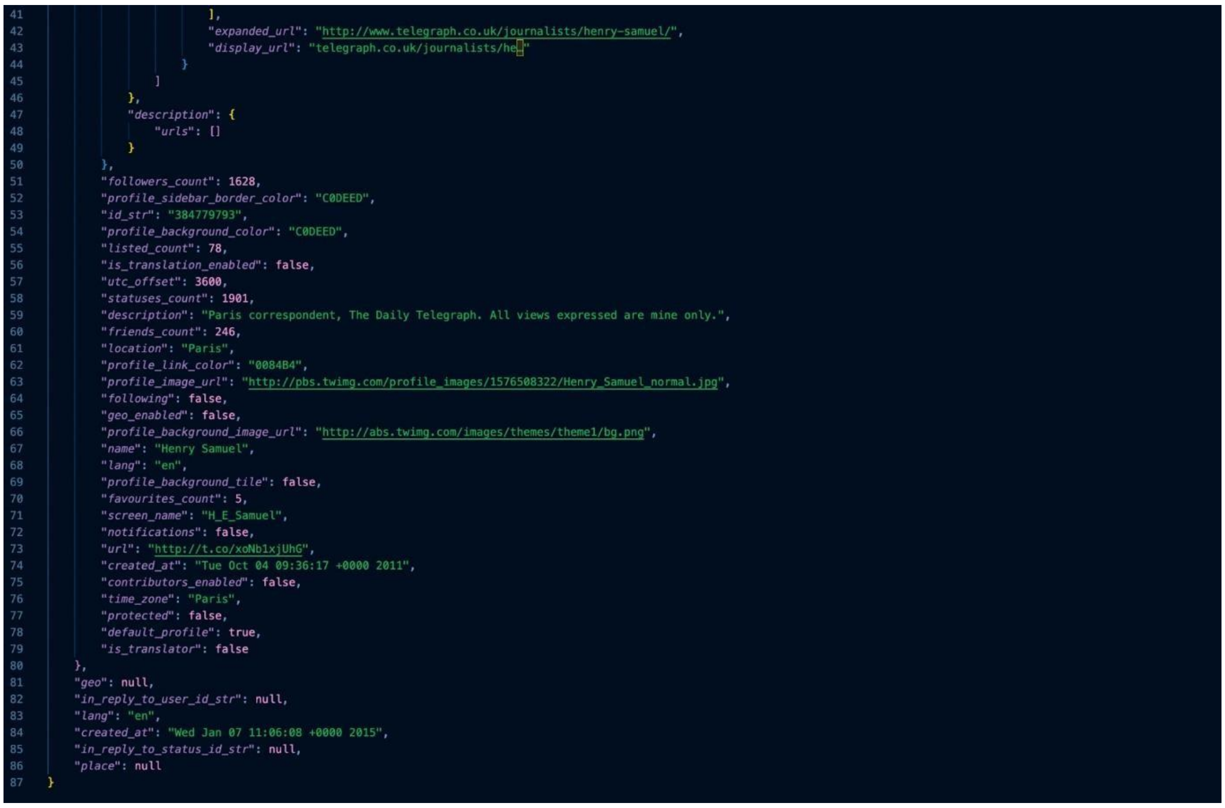
Data extraction allows one to consolidate that information into a centralised system to unify multiple data sets. Extraction gathers data from one or more sources. The process of extracting data includes locating and identifying the relevant data and then preparing it to be transformed and loaded.

Data extraction is essential because it can be used for extracting data from any kind of text. It is especially useful for social media content or any other form of textual data that has been shared on the internet.

In the Pheme dataset, we are given two folders, rumour and non-rumour, for each event. We have two subfolders in these folders: source tweets and reaction tweets. The source tweet folder contains a JSON file containing information about the source tweet, while the reaction tweets folder contains multiple JSON files containing information about the reaction tweets. In the JSON file, we have information about the user name, user mentions and hashtags, contributors, retweets, favourites, follow request, verified, profile locations, follows count, listed count, friends count etc. This is an example of the json file -



```
{
    "contributors": null,
    "truncated": false,
    "text": "Breaking: At least 10 dead, 5 injured after t0 gunman open fire in offices of Charlie  Hebdo,satirical mag that published Mohammed cartoons",
    "in_reply_to_status_id": null,
    "id": 552783238415265800,
    "favorite_count": 14,
    "source": "<a href=\"http://twitter.com/download/iphone\" rel=\"nofollow\">Twitter for iPhone</a>",
    "retweeted": false,
    "coordinates": null,
    "entities": {
        "symbols": [],
        "user_mentions": [],
        "hashtags": [],
        "urls": []
    },
    "in_reply_to_screen_name": null,
    "id_str": "552783238415265792",
    "retweet_count": 159,
    "in_reply_to_user_id": null,
    "favorited": false,
    "user": {
        "follow_request_sent": false,
        "profile_use_background_image": true,
        "profile_text_color": "333333",
        "default_profile_image": false,
        "id": 384779793,
        "profile_background_image_url_https": "https://abs.twimg.com/images/themes/theme1/bg.png",
        "verified": false,
        "profile_location": null,
        "profile_image_url_https": "https://pbs.twimg.com/profile_images/1576508322/Henry_Samuel_normal.jpg",
        "profile_sidebar_fill_color": "DDEEF6",
        "entities": {
            "url": {
                "urls": [
                    {
                        "url": "http://t.co/xoNb1xjUhG",
                        "indices": [
                            0,
                            22
```



```
                        ],
                        "expanded_url": "http://www.telegraph.co.uk/journalists/henry-samuel/",
                        "display_url": "telegraph.co.uk/journalists/he…"
                    }
                ]
            },
            "description": {
                "urls": []
            }
        },
        "followers_count": 1628,
        "profile_sidebar_border_color": "C0DEED",
        "id_str": "384779793",
        "profile_background_color": "C0DEED",
        "listed_count": 78,
        "is_translation_enabled": false,
        "utc_offset": 3600,
        "statuses_count": 1901,
        "description": "Paris correspondent, The Daily Telegraph. All views expressed are mine only.",
        "friends_count": 246,
        "location": "Paris",
        "profile_link_color": "0084B4",
        "profile_image_url": "http://pbs.twimg.com/profile_images/1576508322/Henry_Samuel_normal.jpg",
        "following": false,
        "geo_enabled": false,
        "profile_background_image_url": "http://abs.twimg.com/images/themes/theme1/bg.png",
        "name": "Henry Samuel",
        "lang": "en",
        "profile_background_tile": false,
        "favourites_count": 5,
        "screen_name": "H_E_Samuel",
        "notifications": false,
        "url": "http://t.co/xoNb1xjUhG",
        "created_at": "Tue Oct 04 09:36:17 +0000 2011",
        "contributors_enabled": false,
        "time_zone": "Paris",
        "protected": false,
        "default_profile": true,
        "is_translator": false
    },
    "geo": null,
    "in_reply_to_user_id_str": null,
    "lang": "en",
    "created_at": "Wed Jan 07 11:06:08 +0000 2015",
    "in_reply_to_status_id_str": null,
    "place": null
}
```

Figure 4 Example of JSON File containing the dataset

From this, we only needed the value of a few entries: tweet id, tweet text, no of followers, favourites count, listed count, status and age of the account. For each event, we created one file which contained the following columns - tweet source id, tweets source text, tweet reaction text, verified, followers, listed, status, age, favourite and annotation. Each tweet's reaction is present as an entry in the file, which is then preprocessed and passed into our model.

2. Data pre-processing

We concatenated the data from five events in Pheme to create one dataset file. Punctuation, numbers, single characters, and extra spaces have been removed. We converted the input text from tweets and their reactions to input id and attention masks using the pretrained BERT tokenizer, which were then passed to the BERT model.

Data preprocessing prepares the raw data and makes it suitable for a machine learning model. It is the first and crucial step while creating a machine-learning model.

It transforms the data into a more efficient and effective format for data mining, machine learning, and other data science tasks. The techniques are generally used at the earliest stages of the machine learning and AI development pipelines to ensure accurate results.

The following steps are used in data preprocessing:

2.1. <u>Profiling of data</u> Data profiling is the process of examining, analyzing, and reviewing data in order to collect statistics about its quality. It begins with an examination of existing data and its characteristics. Data scientists identify relevant data sets, inventory their significant attributes, and develop a hypothesis of features that may be relevant for the proposed analytics or machine learning task. They also consider which preprocessing libraries could be used and relate data sources to relevant business concepts.

2.2. **Data purification** The goal is to find the simplest way to address quality issues, such as removing insufficient data, filling in missing data, or ensuring raw data is suitable for feature engineering.

2.3. **Data compression** Raw data sets frequently contain redundant data resulting from different ways of characterizing phenomena or data that is irrelevant to a specific ML, AI, or analytics task. Data reduction techniques such as principal component analysis are used to transform raw data into a more straightforward format suitable for specific use cases.

2.4. **Data conversion** Here, data scientists consider how various aspects of the data should be organised to make the most sense for the goal. This could include organising unstructured data, combining salient variables when appropriate, or identifying significant ranges to concentrate on.

2.5. **Data augmentation** Data scientists apply the various feature engineering libraries to the data in this step to achieve the desired transformations. The end result should be a data set organized to achieve the best possible balance between training time for a new model and compute time.

2.6. **Validation of data** At this point, the data is divided into two sets, the first of which is used to train a machine learning or deep learning model. The second set of data is the testing data, which is used to assess the accuracy and robustness of the final model. This second step aids in identifying any flaws in the hypothesis used in data cleaning and feature engineering. Ifthe data scientists are pleased with the results, they can delegate the preprocessing task to a data engineer, who will determine how to scale it for production. If not, the data scientists can revise how they carried out the data cleansing and feature engineering steps.

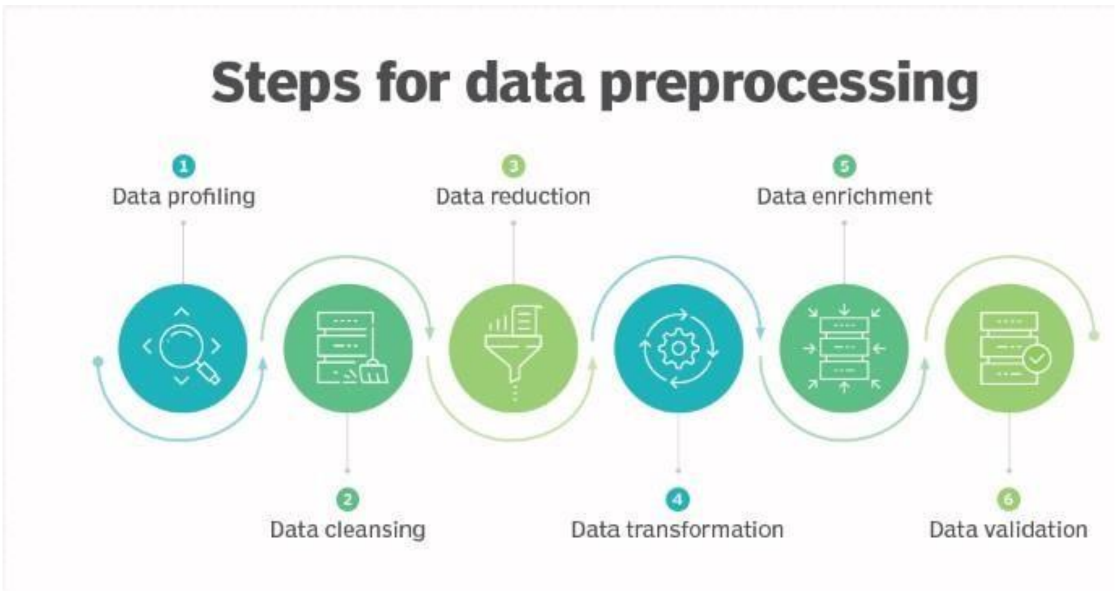


Figure 5 Steps for data preprocessing

3. Why is data preprocessing important?

Data preprocessing is an essential step in data analysis. Preprocessing data has the following advantages:

- It increases precision and dependability. Preprocessing data removes missing or inconsistent data values caused by human or computer error, which can improve a dataset's accuracy and quality, making it more reliable.
- It ensures data consistency. It is possible to have data duplicates when collecting data, and discarding them during preprocessing can ensure the data values for analysis are consistent, which helps produce accurate results.
- It improves the algorithm readability of the data. Preprocessing improves data quality and makes machine learning algorithms read, use, and interpret it more easily.

4. Data preprocessing characteristics.

Preprocessing has many advantages that make it an essential step in data analysis preparation. The two main features are as follows, with a brief explanation:

- **Data validation:** It is the process by which businesses analyze and assess raw data for a project to ensure that it is complete and accurate in order to achieve the best results.

- **Imputation of data:** Data imputation is the process of manually entering missing values and manually correcting data errors during the validation process, similar to business process automation.

5. Techniques for data preprocessing

Preprocessing is divided into two categories: data cleansing and feature engineering.

1. Data cleansing

Some of the goals of data cleansing.

- Identify and classify missing data.
- Reduce noise in your data.
- Identify and eliminate duplicates.

2. Feature development

As previously stated, feature engineering refers to techniques used by data scientists to organise data in ways that make training and inferences more efficient. Among these techniques are the following:

- Normalization or scaling of features
- Data compression
- Discretization
- Encoding of features

6. OUR APPROACH OF PREPROCESSING:

We performed two kinds of preprocessing steps: one for feature comparison and another for our machine-learning model. For feature comparison, we concatenated all of the source tweets from the Pheme dataset's events into a single file. We don't use the replies tweets data for feature comparison because the user features values would be the same for all of the replies data because these are the features of the source tweet. Following this, we obtained data of size 4697 x 8. On this data, we removed any null values as well as any outliers that were present. Then we needed a method to normalize our numerical data while also performing feature scaling.

Feature scaling transforms the values of various numerical features so that they all fall within a similar range. The feature scaling prevents supervised learning models from becoming biased toward a particular range of values. This feature scaling process is carried out so that all features can share the same scale and thus avoid problems such as the following:

- Loss in accuracy
- Increased computational cost as data values vary widely over different orders of magnitude.

For example, suppose the no of followers is in the range of 100000, while the age of the account is just in the range of 1-20. In that case, if we apply our model to this data type, the model will find optimised weights to handle feature values having larger values, resulting in a biased prediction. To prevent this, we do the process of feature scaling. There are two ways to do so: Standardization and Normalization. 1)Standardization is transforming the feature values to fall around the mean as 0 with a standard deviation of 1, while 2) Normalization is transforming the feature values to fall within the bounded intervals.

Since we have already removed outliers from our data, we used Normalisation to perform feature scaling. We used MinMaxScaler to perform Normalisation.

MinMaxScaler shrinks the data within the given range, usually from 0 to 1. It transforms data by scaling features to a given range. It scales the values to a specific value range without changing the shape of the original distribution. The formula used by MinMaxScaler is as follows -

$$x_{norm}^{(i)} = \frac{x^{(i)} - x_{min}}{x_{max} - x_{min}}$$

After that, our dataset is ready for feature comparison.

Preprocessing for Model:

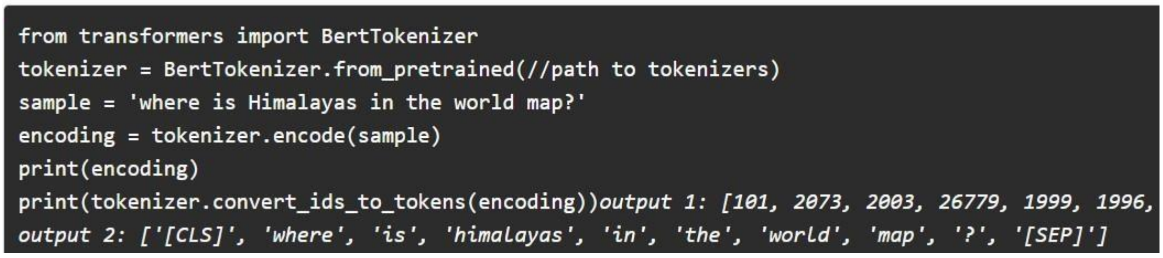
For our model, we concatenated the data of all the events of the Pheme dataset into a single file and used it as our dataset. After this, we had to clean our data. For example, we had to remove punctuation marks, spaces, etc., from the tweet text. For this, we used the RegEx module provided by Python. A RegEx, or Regular Expression, is a sequence of characters that forms a search pattern.RegEx can check if a string contains the specified search pattern.

This built-in package can be called by importing a library called re. This module provides a bunch of functions that help us search for a particular match string. We used the sub-function provided by re, which returns a string where the replace string replaces all matching occurrences of the specified pattern.

Using the sub-function, we removed punctuation marks, numbers, spaces, and single characters from source tweets and their reaction tweets.

We also removed all the tweets whose length was more significant than 140 characters. To pass this data as input to our BERT model, we use the **BertTokenizer**. A tokenizer is in charge of preparing the inputs for a model.

For example –



```
from transformers import BertTokenizer
tokenizer = BertTokenizer.from_pretrained(//path to tokenizers)
sample = 'where is Himalayas in the world map?'
encoding = tokenizer.encode(sample)
print(encoding)
print(tokenizer.convert_ids_to_tokens(encoding))output 1: [101, 2073, 2003, 26779, 1999, 1996,
output 2: ['[CLS]', 'where', 'is', 'himalayas', 'in', 'the', 'world', 'map', '?', '[SEP]']
```

Figure 6 BERT Tokenizer

After that, our dataset is ready for feature comparison. We called the encode function in the preceding code, and what we get in output1 is indices of the input tokens from the vocab file, and output2 is the opposite, a human-readable token of the input ids. In addition to the input tokens, we received two unique tokens, '[CLS]' and '[SEP].' The BERT model requires that the sentence begin with the [CLS] token and end with the [SEP] token.

We generate input ids and attention masks with this tokenizer, which are then used to create a dataset with the TensorFlow dataset utility. The dataset is then divided into 80% training and 20% test sets.

7. Feature Comparison

Next, we aim to add numerical features of tweets like - no of followers, no of listed, age of the account, etc., to our model to increase the accuracy. To analyse how much these features contribute, we do the process of feature comparison.

Feature importance refers to techniques that assign a score to input features based on their usefulness in predicting a target variable. There are many types and sources of feature importance scores, although popular examples include statistical correlation scores, coefficients calculated as part of linear models, decision trees, and permutation importance scores.

Feature importance scores play an important role in a predictive modelling project, including providing insight into the data, insight into the model, and the basis for dimensionality reduction and feature selection that can improve the efficiency and effectiveness of a predictive model on the problem.

For feature comparison, we are using several methods, which are:

- SelectKBest - SelectKBest eliminates all characteristics except for the K-best rating ones.
- Mutual Info - Mutual information between two random variables measures the dependency between the variables. It is equal to zero if two random variables are independent, and higher values mean higher dependence.

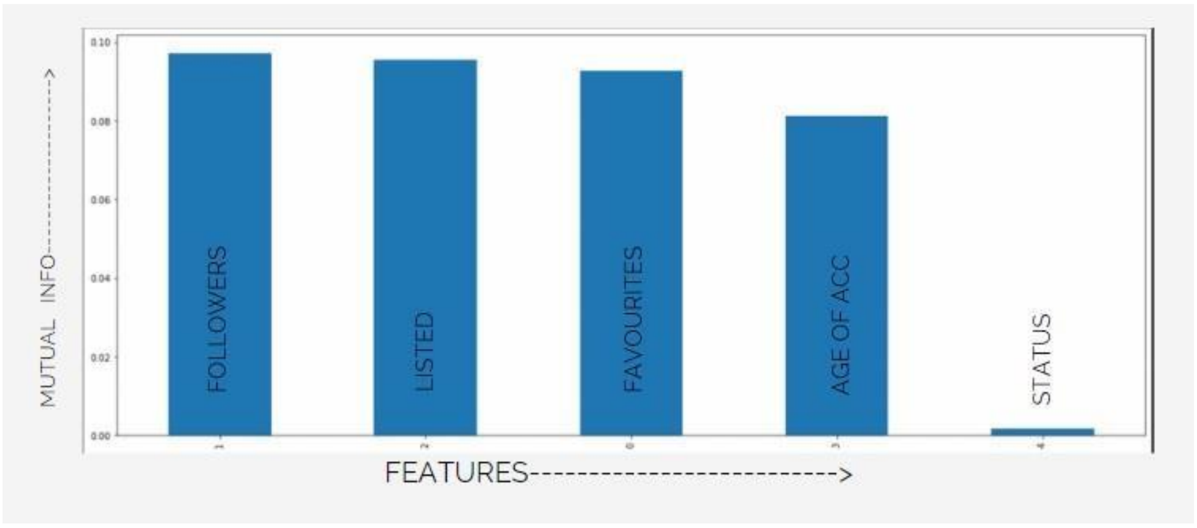


Figure 7 Histogram of contributing features

From Fig. 7, we were able to identify the most contributing features, which we will use in our model as input.

CHAPTER 6

METHODOLOGY

Our method for classifying rumours on Twitter uses BERT rather than procedures based on feature extraction. BERT uses sentence embedding (explained later) that converts words or phrases into vectors, enabling an AI model to understand and perform mathematical operations while also taking into account the semantic relevance of each word, [33]. Figure 8 shows the architecture of our model.

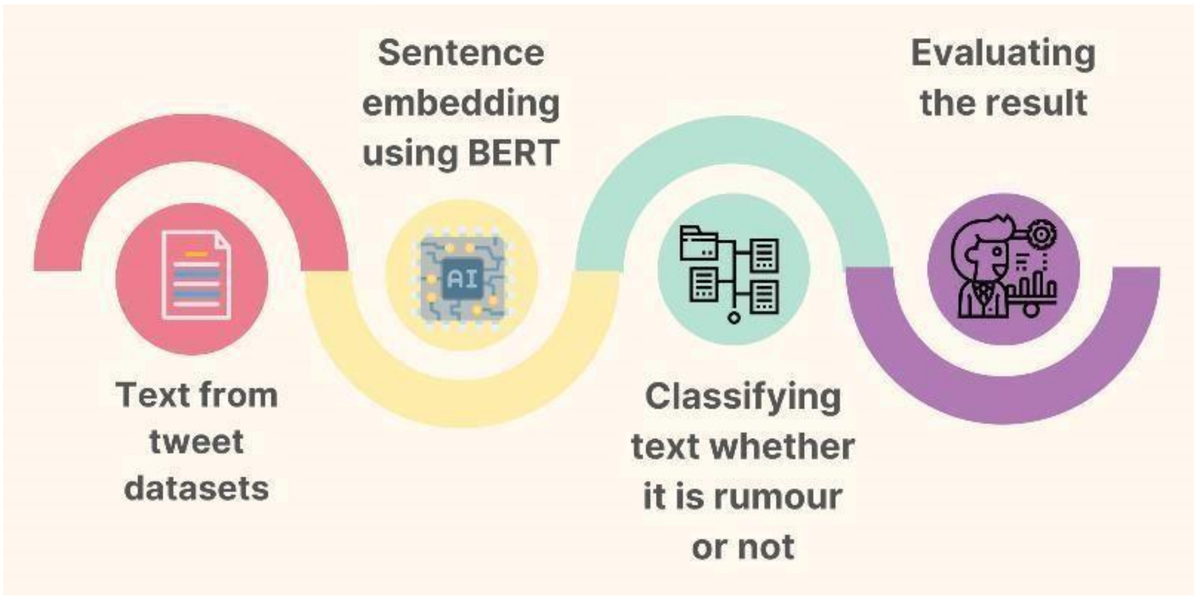

Text from tweet datasets
Sentence embedding using BERT
AI
Classifying text whether it is rumour or not
Evaluating the result

Figure 8 The suggested BERT model for rumour detection

1. Stages in proposed model

1.1. Extracting unstructured tweets from PHEME dataset and their tokenization to get a numerical data structure suitable for machine learning.

1.2. Sentence embedding techniques use vectors to represent entire sentences and their semantic information. This assists the machine in comprehending the context, intent, and othersubtleties in the entire text. BERT gets two sentences as input which are passed to BERT models and a pooling layer to generate their embeddings. Then the embeddings are used to calculate similarity between pairs of sentences.

1.3. In the third stage, these embeddings are employed to train the text classification model for anticipating rumours.

1.4. Evaluation of prediction results takes place in the fourth stage.

2. What is BERT?

Bidirectional Encoder Representation from Transformer (BERT) is a new language contextualising model that is aimed to pretrain deep bidirectional representations from unlabelled text by conditioning on both left and right context concurrently in all layers. As a result, the pretrained BERT model can be fine-tuned with just one additional output layer to create cutting-edge models for a variety of tasks, such as question answering and language interference, without requiring large task-specific architecture changes.

A language model would have studied this text sequence during training from either left to right or from a combination of left to right and right to left in the pre-BERT era. For creating sentences, this one-directional method works well. We may anticipate the subsequent word, add it to the sequence, and then predict the subsequent word until we have a complete sentence.

BERT randomly masks words in the sentences and then tries to predict them. When a word is "masked," the model uses both left and right surroundings, as well as both directions of the sentence, to anticipate the hidden word. It considers the previous and following tokens simultaneously, in contrast to earlier language models. This "same-time portion" was absent from the existing combined left-to-right and right-to-left LSTM based models. Figure 9 shows the difference between the two approaches more vividly

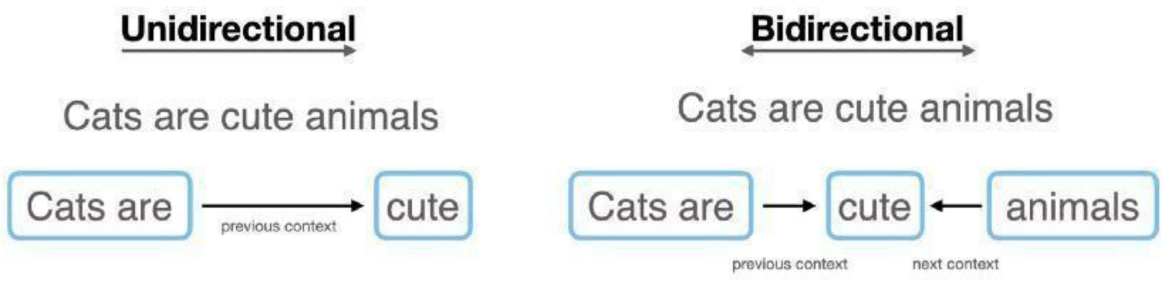


Figure 9 Unidirectional vs Bidirectional Approach

2.1. <u>Unidirectional Models</u>

The aim is to anticipate the subsequent word using the prior context given the statement "Cats are cute animals." By using a unidirectional case, the following word is chosen based on the prior context. The language model specifically tries to anticipate the following word ("cute") based on the words that appeared earlier ("Cats are"). The left-to-right context serves as the foundation for this kind of language modelling. In contrast, the model employs the following context and is therefore based on the right-to-left context when it uses the phrases ("are cute animals") that followed after to predict the word "Cats."

2.2. <u>Bidirectional Models</u>

The left and right contexts serve as the foundation for the bidirectional language models. The activity is comparable to guessing a word from a sentence by reading it backward and forth. In this instance, both the left context "Cats are " and the right context "animals" are used to predict the word "cute" when it needs to be done. The bidirectional models encounter a new restriction because they use both the left and right context of the statement. The model permits words to indirectly see themselves [34] when conditioning on both the prior and subsequent context. BERT mentioned the limitation and introduced the aforementioned masked LM task to train LMs bidirectionally.

3. BERT architecture model

BERT is a multi-layer bidirectional transformer. A primary transformer is made up of a decoder to forecast the task's outcome and an encoder to interpret the text input. Since the purpose of BERT is to produce a language representation model, just the encoder portion is required. A vector is created from a series of tokens as the encoder input for BERT. BERT offers the BEST BASE and BERT LARGE architecture models. Let L denote the number of layers (Transformer Blocks), H the hidden size, and A the number of self-attention heads. Figure 10 shows the two BERT models with different model sizes. BERT-Base consists of 12 stacked transformer encoders, whereas BERT-Large consists of 24. BERT-Base has the same model size as OpenAI GPT for comparison purposes. Both model architectures are similar to each other, but OpenAI GPT uses constraint self-attention, while BERT uses bidirectional self-attention. This study uses BERTBASE to reduce computational complexity.

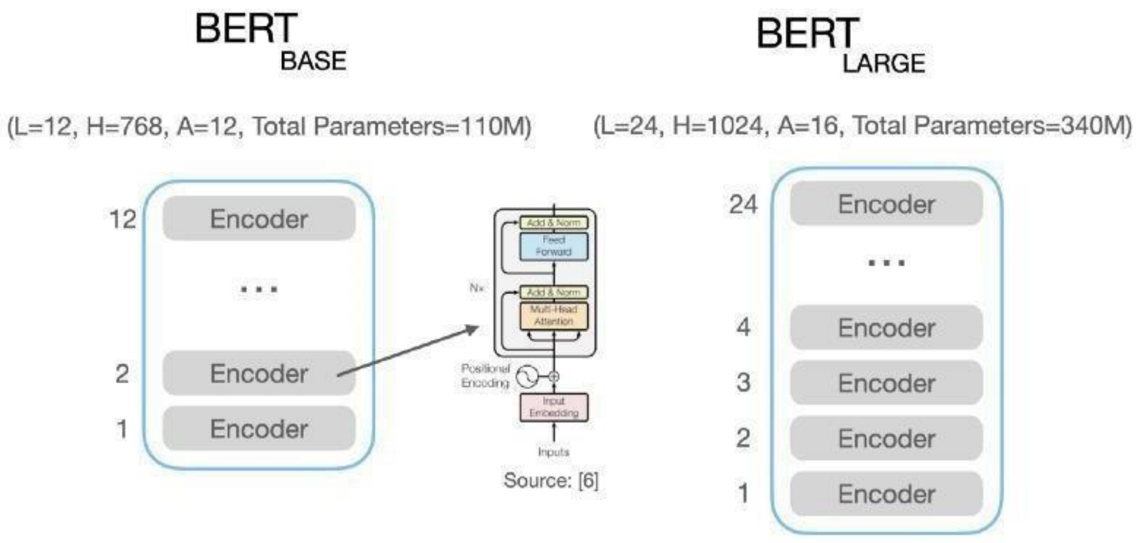


Figure 10 BERT architecture models

4. BERT Input/Output representation

The input of BERT is one token sequence. The input sequence can be either a sentence or a pair of sentences. The word “sentence” does not refer to a linguistic sentence but a contiguous text [1]. The word “sentence” refers to an input token sequence, which can be a single sentence or sentences packed together. BERT uses WordPiece Embeddings [12] in order to create the token embeddings. The WordPiece has 30,000 vocabulary words. If a word is not in the dictionary, the WordPiece algorithm divides the word into subwords until it finds a matching subword that is in the dictionary. In this way, the unknown words can be divided into known words from the dictionary and corresponding embeddings are used to represent the subwords. The WordPiece embeddings of the words correspond then to the token embeddings. Furthermore, BERT has two special tokens to encode the input sequence: the classification token [CLS] and the separator token [SEP]. Every input sequence starts with the classification token. The aggregated final representation of the classification token is used for the classification tasks. Moreover, the separator token is put between sentences and separates the sentences. Figure 11 shows the input representation of BERT.

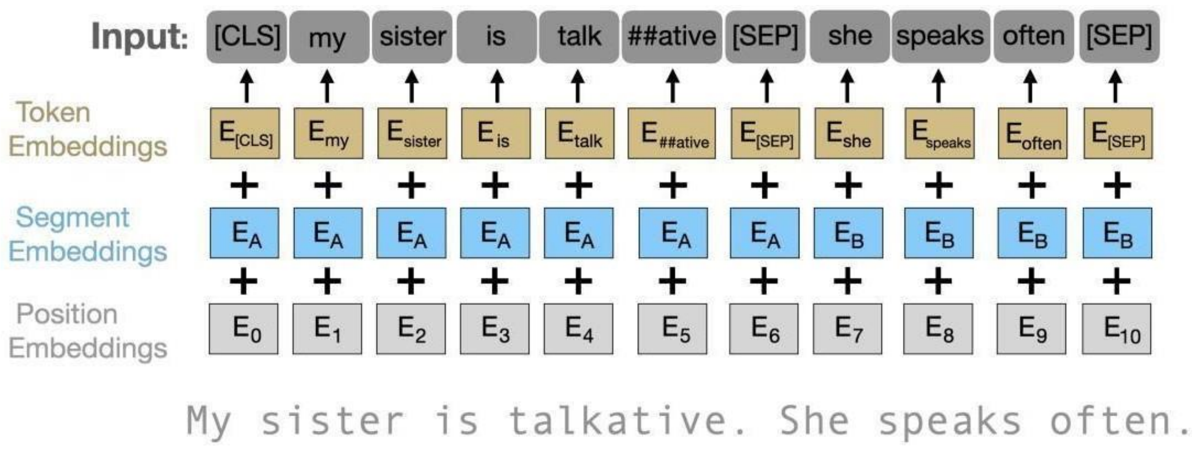

Input:
[CLS]
my
sister
is
talk
##ative
[SEP]
she
speaks
often
[SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} E_{sister} E_{is} E_{talk} $E_{\#\#ative}$ $E_{[SEP]}$ E_{she} E_{speaks} E_{often} $E_{[SEP]}$
Segment Embeddings
E_A E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}
My sister is talkative. She speaks often.

Figure 11 BERT Input Representation

When we consider the example sentence, it can be seen that the word “talkative” is divided into two tokens “talk” and “##ative”. As the word “talkative” is not in WordPiece dictionary, the algorithm divided it into subwords. Since “talk” is in the dictionary, the token embedding of it can be used. Apart from the token embeddings, there are segment and position embeddings. Segment embeddings are vectors that denote if a word belongs to the first sentence or the second sentence. Position embeddings denote the ordering of the tokens. The sum of token embeddings, segment embeddings, and position embeddings corresponds to the input representation.

CHAPTER 7

EXPERIMENTS AND RESULTS

1. Experimental Steps

We are using BERTBASE-uncased, as it provided us with the most accuracy. After preprocessing our data, we have a dataset that contains source tweets, reaction tweets, and user features. We have also generated the input ids and masks from source and reaction tweets. We define two input layers for input ids and input masks, then initialize our BERT model with these two layers. The number of output neurons of these two layers is kept at 280. This layer output is then passed to another dense layer which acts as an intermediate layer and has ReLU as its activation function. This layer helps in neuron activation. The number of output neurons in this layer is 128. Then another intermediate layer is created in which user features are passed as input. The number of output neurons in this layer is 6. These two layers are concatenated with each other, and the output is passed to the output layer. After concatenation, we find that the number of the output neurons will become 134, which is the addition of the number of neurons of the two layers. The output layer is also a type of dense layer with softmax as its activation function, and the number of output neurons in this layer is three, which denotes - positive, negative, and neutral.

We then use the Adam optimizer, setting the learning rate to $1e^{-5}$ and the decay rate to $1e^{-6}$. Adam is an optimization algorithm that can be used to update network weights based on training data iteratively instead of the traditional stochastic gradient descent procedure. The learning rate is a hyper-parameter that governs how much the weights in the network change concerning the loss gradient. The lower the value, the slower the descent. We chose this value for our model after experimenting with learning rate $1e^{-5}$.

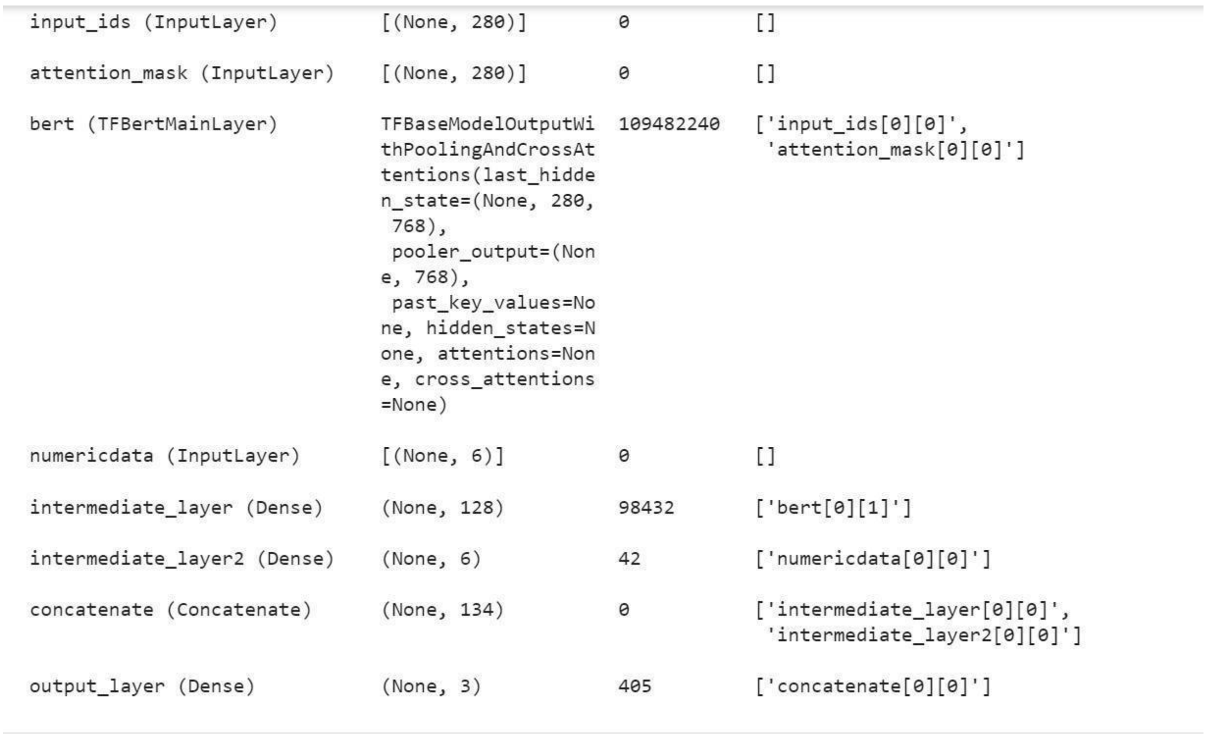
To compute the loss value, we use the categorical cross-entropy function. The loss function categorical cross-entropy is used in multi-class classification tasks.

These are tasks in which an example can only belong to one of several possible categories, and the model must determine which one. The categorical cross-entropy loss function computes an example's loss by adding the following sums:

$$\text{Loss} = -\sum_{i=1}^{\substack{\text{output}\\ \text{size}}} y_i \cdot \log \hat{y}_i$$

where y_i is the corresponding target value, and output size is the number of scalar values in the model output.

The overall architecture of out model can be understood by this diagram:



```
 input_ids (InputLayer)         [(None, 280)]        0           []

 attention_mask (InputLayer)    [(None, 280)]        0           []

 bert (TFBertMainLayer)         TFBaseModelOutputWi  109482240   ['input_ids[0][0]',
                                thPoolingAndCrossAt               'attention_mask[0][0]']
                                tentions(last_hidde
                                n_state=(None, 280,
                                 768),
                                 pooler_output=(Non
                                e, 768),
                                 past_key_values=No
                                ne, hidden_states=N
                                one, attentions=Non
                                e, cross_attentions
                                =None)

 numericdata (InputLayer)       [(None, 6)]          0           []

 intermediate_layer (Dense)     (None, 128)          98432       ['bert[0][1]']

 intermediate_layer2 (Dense)    (None, 6)            42          ['numericdata[0][0]']

 concatenate (Concatenate)      (None, 134)          0           ['intermediate_layer[0][0]',
                                                                  'intermediate_layer2[0][0]']

 output_layer (Dense)           (None, 3)            405         ['concatenate[0][0]']
```

Figure 12 BERT Model Architecture

2. Results

For evaluating our model we have calculated the values of accuracy, precision, recall and F1 score. After training the model for 2 epochs we get the following results –

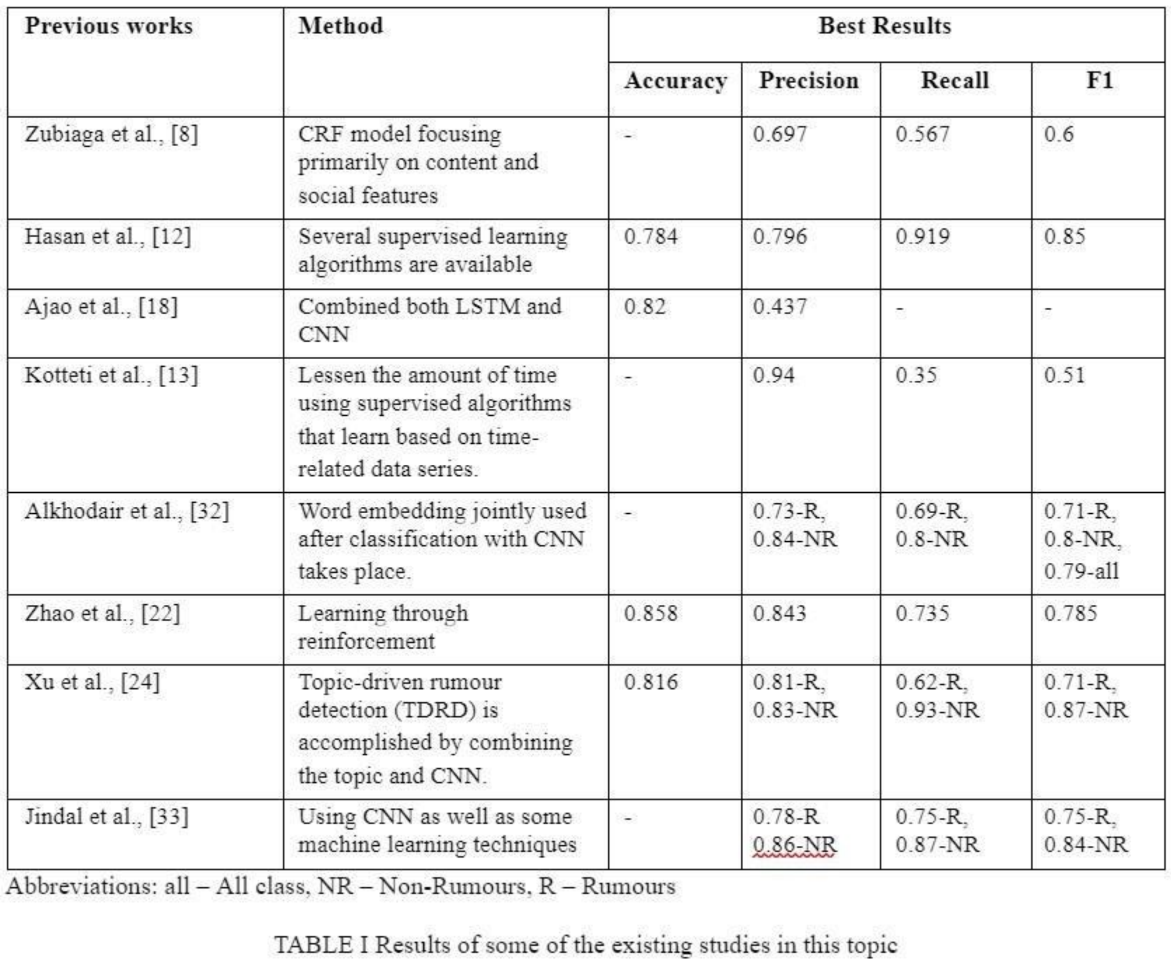
Accuracy: 0.9002631943482476

Precision: 0.7345767575322812

Recall: 0.745269286754003

F1 score: 0.7398843930635839

<u>Comparison With Different Models-</u>



Previous works	Method	Best Results			
		Accuracy	**Precision**	**Recall**	**F1**
Zubiaga et al., [8]	CRF model focusing primarily on content and social features	-	0.697	0.567	0.6
Hasan et al., [12]	Several supervised learning algorithms are available	0.784	0.796	0.919	0.85
Ajao et al., [18]	Combined both LSTM and CNN	0.82	0.437	-	-
Kotteti et al., [13]	Lessen the amount of time using supervised algorithms that learn based on time-related data series.	-	0.94	0.35	0.51
Alkhodair et al., [32]	Word embedding jointly used after classification with CNN takes place.	-	0.73-R, 0.84-NR	0.69-R, 0.8-NR	0.71-R, 0.8-NR, 0.79-all
Zhao et al., [22]	Learning through reinforcement	0.858	0.843	0.735	0.785
Xu et al., [24]	Topic-driven rumour detection (TDRD) is accomplished by combining the topic and CNN.	0.816	0.81-R, 0.83-NR	0.62-R, 0.93-NR	0.71-R, 0.87-NR
Jindal et al., [33]	Using CNN as well as some machine learning techniques	-	0.78-R 0.86-NR	0.75-R, 0.87-NR	0.75-R, 0.84-NR

Abbreviations: all – All class, NR – Non-Rumours, R – Rumours

TABLE I Results of some of the existing studies in this topic

Table 1 Results of some of the existing state-of-the art studies in this area

CHAPTER 8

CONCLUSIONS

We investigated whether the output of BERT could be employed to train a rumour detection model in this experiment. A new technique to recognize rumours that uses BERT and the context of tweets has been suggested. Our experiments revealed that BERT's sentence embedding feature could distinguish between rumour and non-rumour tweets without requiring the extraction of tweet features. Other various feature-based classification models have been outperformed when using BERT's sentence embedding. To detect whether a tweet is a rumour, it is better to use source tweet text, reaction tweets text, and source tweet user features to predict the outcome, as using all these features gives us the best accuracy.

8.1 Future Scope

In this study, we have only used data from Twitter. However, rumours are present on almost all platforms. So in the future scope, we can add detection on various social platforms. The study can also be performed on the real-time classification of tweets. In our study, we have just used text as a classification criterion; however, on many tweets, there are images, documents, and videos that spread rumours, so a study can be conducted to detect those types of rumour.

References

1. Alexander Stocker, Alexander Richter and Kai Riemer, "A Review of Microblogging in the Enterprise", it-Information Technology Methoden und innovative Anwendungen der Informatik und Informationstechnik, vol. 54, no. 5, pp. 205-211, 2012.
2. Haewoon Kwak et al., "What is Twitter a social network or a news media?", Proceedings of the 19th international conference on World wide web, 2010.
3. Álvaro Cuesta, David F. Barrero and María D. R-Moreno, "A Descriptive Analysis of Twitter Activity in Spanish around Boston Terror Attacks", International Conference on Computational Collective Intelligence, 2013.
4. Aditi Gupta, Hemank Lamba and Ponnurangam Kumaraguru, "\$1.00 per rt# bostonmarathon# prayforboston: Analyzing fake content on twitter", eCrime Researchers Summit (eCRS), 2013.
5. Xin Lu and Christa Brelsford, "Network structure and community evolution on twitter: human behavior change in response to the 2011 Japanese earthquake and tsunami", Scientific reports, vol. 4, pp. 6773, 2014.
6. Zahra Ashktorab et al., "Tweedr: Mining twitter to inform disaster response", ISCRAM, 2014.
7. Aditi Gupta et al., "Faking sandy: characterizing and identifying fake images on twitter during hurricane sandy", Proceedings of the 22nd international conference on World Wide Web, 2013.
8. Zubiaga, M. Liakata, R. Procter, G. Wong Sak Hoi, and P. Tolmie, “Analysing how people orient to and spread rumours in social media by looking at conversational threads,” PLoS ONE, vol. 11, no. 3, 2016.
9. Bondielli and F. Marcelloni, “A survey on fake news and rumour detection techniques,” Information Sciences, vol. 497, pp. 38–55, 2019.
10. Castillo, M. Mendoza, and B. Poblete, “Information credibility on Twitter,” Proceedings of the 20th International Conference Companion on World Wide Web, WWW 2011, pp. 675–684, 2011.
11. J. Ito, J. Song, H. Toda, Y. Koike, and S. Oyama, “Assessment of tweet credibility with lda features,” in Proceedings of the 24th International Conference on World Wide Web, 2015,pp. 953–958.

12. N. Y. Hassan and M. H. Haggag, “Supervised Learning Approach for Twitter Credibility Detection,” 2018 13th International Conference on Computer Engineering and Systems (ICCES), pp. 196–201, 2018.

13. M. M. Kotteti, X. Dong, and L. Qian, “Multiple time-series data analysis for rumor detection on social media,” in 2018 IEEE International Conference on Big Data (Big Data). IEEE, 2018, pp. 4413–4419.

14. J. Ma, W. Gao, P. Mitra, S. Kwon, B. J. Jansen, K.-f. Wong, and M. Cha, “Detecting Rumors from Microblogs with Recurrent Neural Networks,” Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI 2016), pp. 3818–3824, 2015.

15. N. Ruchansky, “CSI: A Hybrid Deep Model for Fake News Detection,” Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 797–806, 2017.

16. K. Zhou and B. Li, “Early Rumour Detection,” Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1, pp. 1614–1623, 2019.

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

18. S. Chatterjee, S. Deng, J. Liu, R. Shan, and W. Jiao, “Classifying facts and opinions in twitter messages: a deep learning-based approach,” Journal of Business Analytics, vol. 1, no. 1, pp. 29–39, 2018.

19. Y. Hu, K. Talamadupula, and S. Kambhampati, “Dude, srsly?: The surprisingly formal nature of twitter’s language,” in Proceedings of the International AAAI Conference on Web and Social Media, vol. 7, no. 1, 2013.

20. Zhao, Z., Resnick, P., and Mei, Q. (2015). Enquiring minds: Early detection of rumorsin social media from enquiry posts. In Proceedings of the 24th International Conference on World Wide Web, pages 1395–1405. ACM.

21. F. Xu, V. S. Sheng, and M. Wang, “Near real-time topic-driven rumor detection in source microblogs,” Knowledge-Based Systems, vol. 207, p. 106391, 2020.

22. G. W. Allport and L. Postman, “The psychology of rumor.” ”The psychology of rumor.” (1947)., 1947.

23. R. L. Rosnow, “Rumor as communication: A contextualist approach,” Journal of Communication, vol. 38, no. 1, pp. 12–28, 1988.

24. Qazvinian, V., Rosengren, E., Radev, D. R., and Mei, Q. (2011). Rumor has it: Identifying misinformation in microblogs. In Proceedings of EMNLP, pages 1589–1599.

25. Liu, X., Nourbakhsh, A., Li, Q., Fang, R., and Shah, S. (2015). Real-time rumor debunking on twitter. In Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, pages 1867–1870. ACM.

26. Takahashi, T. and Igata, N. (2012). Rumor detection on twitter. In Soft Computing and Intelligent Systems (SCIS) and 13th International Symposium on Advanced Intelligent Systems (ISIS), 2012 Joint 6th International Conference on, pages 452–457. IEEE.

27. Fan Yang, Yang Liu, Xiaohui Yu, and Min Yang. 2012. Automatic detection of rumor on Sina Weibo. In Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics. ACM, 13.

28. Rob Procter, Farida Vis, and Alex Voss. 2013b. Reading the riots on Twitter: Methodological innovation for the analysis of big data. Int. J. Soc. Res. Methodol. 16, 3 (2013),197–214.

29. Y. Hu, K. Talamadupula, and S. Kambhampati, “Dude, srsly?: The surprisingly formal nature of twitter’s language,” in Proceedings of the International AAAI Conference on Web and Social Media, vol. 7, no. 1, 2013.

30. T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” arXiv preprint arXiv:1310.4546, 2013.

31. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

32. S. A. Alkhodair, S. H. H. Ding, and B. C. M. Fung, “Detecting breaking news rumors of emerging topics in social media,” Information Processing and Management, vol. 57, no. 2, p. 102018, 2020. [Online]. Available: https://doi.org/10.1016/j.ipm.2019.02.016.

33. M. Bharti and H. Jindal, “Automatic Rumour Detection Model on Social Media,” 2020 Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC), pp. 367–371, 2021.