Fake News Detection based on User Engagements in Online Social Networks

A Project Report

submitted by

FARSHANA FATHIMA (COE17B001)

GAYATHRI DEVI KONERU (COE17B029)

PRANAVI GATTU (COE17B048)

in partial fulfilment of requirements

for the award of the degree of

BACHELOR OF TECHNOLOGY

Department of Computer Science and Engineering

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, DESIGN AND MANUFACTURING KANCHEEPURAM

MAY 2021

DECLARATION OF ORIGINALITY

We, **Farshana Fathima, Gayathri Devi Koneru, Pranavi Gattu**, with Roll Numbers: **COE17B001, COE17B029, COE17B048** hereby declare that the material presented in the Project Report titled **Fake News Detection based on User Engagements in Online Social Networks** represents original work carried out by me in the **Department of Computer Science and Engineering** at the Indian Institute of Information Technology, Design and Manufacturing, Kancheepuram.

With our signatures, We certify that:

- We have not manipulated any of the data or results.
- We have not committed any plagiarism of intellectual property. We have clearly indicated and referenced the contributions of others.
- We have explicitly acknowledged all collaborative research and discussions.
- We have understood that any false claim will result in severe disciplinary action.
- We have understood that the work may be screened for any form of academic misconduct.

Farshana Fathima
Gayathri Devi Koneru
Pranavi Gattu

Place: Chennai

Date: 04.05.2021

CERTIFICATE

This is to certify that the report titled **Fake News Detection based on User Engagements in Online Social Networks**, submitted by **Farshana Fathima (COE17B001), Gayathri Devi Koneru (COE17B029), Pranavi Gattu (COE17B048)**, to the Indian Institute of Information Technology, Design and Manufacturing Kancheepuram, for the award of the degree of **BACHELOR OF TECHNOLOGY** is a bona fide record of the work done by him/her under my supervision. The contents of this report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr B Sivaselvan

Digitally signed by Dr B Sivaselvan
DN: cn=Dr B Sivaselvan, o=IIITDM Kancheepuram, ou=Dept of CSE, email=sivaselvanb@iiitdm.ac.in, c=IN
Date: 2021.05.04 14:16:47 +05'30'

Dr. SIVASELVAN
Project Guide
Associate Professor
Department of Computer Science and Engineering
IIITDM Kancheepuram, 600 127

Place: Chennai

Date: 04.05.2021

ACKNOWLEDGEMENTS

We would like to extend our sincerest gratitude to our project guide, Dr. B. Sivaselvan who encouraged us to take up this project and providing us with his guidance whenever needed and also for being a constant support. We would also like to express our gratitude towards the Research Scholar, Santhosh Kumar Uppada for encouraging us throughout the project and providing us with his suggestions that helped in the betterment of the project.

We would also like to thank the entire COE department for helping us throughout our journey in IIITDM Kancheepuram.

We are also indebted to IIITDM Kancheepuram for giving us an opportunity to research in the field of our interest and helping us to improve our skills in all ways possible.

ABSTRACT

News is the main source of information to know everything happening around the world. It is highly essential that this news is true and real. But unfortunately, we see high dissemination of fake news in our day-to-day lives. Fake news is intentionally and verifiably false information which is created for various reasons such as political and financial gain, etc. Due to the rapid shift in the consumption of news from traditional news media to online social media because its often more timely, less expensive and easy to share and comment on, the reach of fake news also rapidly increased due to the lack of proper authenticity of the users. This fake news has proven to have many negative impacts on people, directly or indirectly.

Humans can be irrational and vulnerable differentiating between truth and falsehood when they are bombarded with deceptive information. Research done in the field of social psychology and communications have demonstrated that the human ability to detect deception is just slightly better than chance - with a mean accuracy of 54% over 1,000 participants in over 100 experiments[1]. And hence, it's highly essential that fake news is detected as early as possible.

The Existing methods of traditional Expert-oriented fact checking for the detection of fake news is intellectually demanding and time-consuming process for the fact-checcher, this results in low efficieny and limits the potential for scalability. Hence, fake news detection needs to be automated. In this paper, we are going to present the literature review of some of the existing automatic fake news detection methods and also related areas like rumour detection methods. We also present our problem statement and our methodology of detecting fake news.

We have started with the data mining approaches for fake news detection looking into the characteristics and foundations of the people spreading and consuming the fake news on traditional news media and online social media. Then the various feature extraction techniques for linguistic and visual based news content and also on social con-

text (user, post and network based) has been discussed. A brief on the existing datasets and evaluation metrics has also been included in this report.

A **heterogenous tweet-word-user graph attention networks is discussed for the early detection of rumours** which uses subgraph masked attention mechanism considering both global semantic relations between the text contents and the global structural information of the propagation of the tweets to detect fake news using Twitter15 and Twitter16 datasets along with the user information of the tweets collected using Twitter API and it is seen that it gives an **overall accuracy of 0.911 for Twitter15 and 0.924 for Twitter16.**

Next, we looked through a paper which discusses about detecting **Fake news spread on OSNs during HongKong Protests using only Text-based features** like PoS, sentiment, tweet length etc. They have achieved An **F1 score of 92.4%**.Here we reached a conclusion that **sole dependence in text features might lead to discrepancies when tweets of different languages are looked at.**

The paper on Weakly Supervised model for Fake news detection discusses about using a dataset with low quality labels and still achieving an F1 score of 0.94. This report goes into detail about the procedures taken to deal with noisy inaccurate dataset. **Random Forest and XG Boost is applied resulting in a F1 score of 0.78 without inclusion of user level features** and **F1 score of 0.94 after including user level features**.

The **CSI (Capture, Score and Integrate)** paper proposes a novel method **a Hybrid Deep model that not only accurately classifies fake news but also identifies groups of suspicious users**. The Temporal representation of engagements that are spreading the fake articles is *Captured* using LSTM and the user behaviour in spreading these articles is *Scored* and finally the output is *Integrated* to classify the News as fake or real. **An F1 score of 0.894 and 0.954 is achieved by this model on the Twitter dataset and the Weibo dataset.**

To improve detection performance with limited information, a multi-depth Graph Convolution Network Model is proposed. This model constructs a network using the users profile data by taking into account the similarities among the news

based on the information that is obtained from the neighbours of every node in the network using multi-depth GCN blocks, and combines them using attention mechanisms. This model is tested on LIAR dataset, which **detected fake news with an accuracy of 49.2% which is higher than some of the latest methods like Hybrid-CNN, LSTM-Attention, Memory-Network, MMFD and GCN**.

The aim of **network-based pattern-driven fake news detection** approach is to study the patterns of fake news in social networks, which refers to the spreaders of the news, the news being spread and relationships among the spreaders. These **patterns are represented at various network levels** over which feature-engineering is performed to model the features that are later used to detect fake news. After this, several supervised machine learning algorithms are employed to detect fake news over real world datasets provided by PolitiFact and BuzzFeed. **Random Forest(RF) classifier gives the best accuracy and F_1 score of around 0.93 and 0.84 on Politifact and BuzzFeed.** Additionally, it is said that the model can give even more better results by introducing more such patterns.

To develop our model of fake news detection, we collected a labelled dataset called **FakeNewsNet** which contains articles and tweets related to them and also information about the users who sent out the tweets. The articles and the related user and tweet information are obtained using this crawler with the help of Twitter Developer Account.

For feature engineering, we analysed **text of the tweets, temporal aspects and user profile information** and arrived at some inferences. Based on these inferences, all the features are collated. Multiple subsets of the available features set are chosen and fed to various classifiers like SVC, Random Forest Classifier, Gradient Boosting Classifier and Adaboost Classifier. Based on the performances of these classifiers over various subsets of the features set, a final features set is agreed upon after eliminating the least contributing features.

Feature selection phase shows that the performances of the classifiers varies with the features chosen. Hence our fake news detection model should incorporate all the strategies followed by different machine learning models to have the advantages of all the different models. Hence we choose an ensemble learning model, **Stacked Generalization** which combines different machine learning models and trains the dataset in

two levels. We performed the training using different combinations of machine learning models and the model having **KNN, LR, SVC, RFC, GBC, ABC and XBC** as Base models and RFC as Meta model performed well. This Stacking model is based upon the advantages of the multiple classifiers it combines and classifies the tweets into fake and real with an **accuracy of 99.86% and 93.48% on the train and test sets**.
Based on the predictions made by this Stacking model, the articles (whose tweets are classified) are classified into Fake and Real.

KEYWORDS: Fake News; Politifact; Online Social Networks (OSNs); FakeNewsNet; Stacked Generalization.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

NR	Non Rumor
TR	True Rumor
FR	False Rumor
UR	Unverified Rumor
RNN	Recurrent Neural Network
TF-IDF	Term Frequency Inverse Document Frequency
PMI	Pointwise Mutual Information
DTC	Decision Tree Classifier
RFC	Random Forest Classifier
SVM-TS	Support Vector Machine - Time Series
GLAN	Global Local Attention Network
CSI	Capture Score Integrate Model
SVD	Singular Value Decomposition
OSN	Online Social Network
NLP	Natural Language Processing
GCN	Graph Convolutional Network
M-GCN	Multi depth Graph Convolutional Network
CNN	Convolutional Neural Network
LSTM	Long Short Term Memory
MMFD	Multi-source Multi-class Fake news Detection
FNN	Fake News Network
TNN	True News Network
NER	Name Entity Recognition
NB	Naive Bayes
KNN	K Nearest Neighbours
SVC	Support Vector Classifier
LR	Logistic Regression

GBC	Gradient Boosting Classifier
XBC	XgBoost Classifier
ABC	AdaBoost Classifier
GPE	Geo-Political Entity
ORG	Organization
NORP	Nationalities or religious or political groups

CHAPTER 1

INTRODUCTION

The usage and impact of online social media in our life is increasing day by day. Population of people seeking news from online social media is rapidly increasing when compared to traditional news media. According to journalism.org, in the year 2016, 62% adults in the U.S received news from online social media and in 2012, only 49% was reported to receive news from online social media. On social networking sites, the reach and effects of information spread are significantly escalated and occur at a very fast pace. Therefore distorted, inaccurate or false information acquires a tremendous potential to cause real impact on people within minutes, for millions of users.

1.1 Background

Fake News can be defined as a news article that is intentionally and verifiably false[2] designed to manipulate people's perception of real facts, events and statements. Fake news can be categorised into two types: Traditional Fake news and Fake news on Social Media.

Traditional Fake News is spread when a publisher gives more importance to maximizing profit based on the number of customers than to the authenticity of the news and when the customer gives more importance to their existing views and perceptions than trying to obtain the true and unbiased news. Due to the existing ***Psychological Foundations*** amongst people, they always believe that their perceptions and assumptions of reality are the accurate views, while others who contradict with their views are considered as uninformed, irrational or biased and also they prefer to receive information that validates their existing views. At times when actual fake news is spread out as real news and later on its found out to be fake, it may create more misunderstandings among the people. And also due to the ***Social Foundations*** that exist in societies, usually agree

with the socially safe news when consuming and disseminating news i.e, they usually accept news which is accepted by the people in their society.

Fake News on Social Media is spread by ***Malicious accounts*** viz cyborg users, social bots and trolls that are created for the purpose of spreading misinformation and creating a sense of uncertainty among people. Cyborg users are humans who use automation to amplify their messages. The easy switch of activities from user to machine helps them to spread the fake news. Social bots are computer programs which act as an agent for a user to communicate with other bots or users. Trolls are real human beings who provoke users and also spread the fake news easily. Sometimes, they are paid for spreading the fake news. Since social media is personalized for every user, users get the news from the sources they like and follow. They form groups with like-minded people and therefore tend to believe the news which is accepted by most of the people and also believe the news which they frequently hear. This is called ***Echo Chamber Effect***. In Echo Chambers, people tend to consume and share the same news repeatedly, so that is enough to create a positive opinion on a news regardless of it being fake or not.

The massive spread of fake news can have a serious negative impact on individuals and society. It can disturb the authenticity and balance of the news ecosystem. People will no longer be able to differentiate between what is real and what is fake. The more people are exposed to fake news the high probability that they tend to believe that its true.

1.2 Motivation

In order to distinguish between Fake news and Real News, some websites like *Politifact* manually fact check if a news is fake or not. Each article is checked by three expert journalists before being labelled as fake or true. With the increase in the volume of the newly created information, the time taken for fact checking the news manually increases, especially on social media, hence giving the motivation for the automation of fake news detection.

Automatic detection of fake news is really a challenging problem as the real and

fake news is spread at the same time when we are not even aware of the news and when we don't even have proper resources to verify the authenticity of the news in the initial stages. And in the later stages, the spread becomes vigorous and the data is large. Hence it is important that we don't just focus on detecting fake news but also focus on detecting them **as early as possible**.

1.3 Automated Fake News Detection in Social Media

The features that are selected for the detection of fake news have a major impact on the quality of insights we gain and also on the performance of the detection. Therefore in this section, we will look at features used in more detail. The features that are considered during automated fake news detection can be split into two categories namely, **News Content** features and **Social Context** features.

News Content Features are split into Visual based and Linguistic based features.

- ***Visual Based Features*** are videos or images present in the article.
- ***Linguistic Based Features*** are publisher/author, headline, body of the text etc present in the news article. Since our project focuses on analysing textual data for fake news detection in OSNs, we will look at linguistic features in more detail. As fake news is deliberately created for political or financial gain, it contains inflammatory language (clickbaits) to grab the attention of the users such that they click the links and also to create confusion. In such cases linguistic based features are to be exploited. Two types of linguistic features are common linguistic features and domain based linguistic features. Common linguistic features include lexical features like characters per word, total words, unique words, etc and syntactic features like frequency of words and phrases - bag-of-words and n-grams approach, sentence-level features, etc. Domain based linguistic features include the average length of graphs, number of graphs, external links, quoted words, etc.

Social Context Features consider user-driven social engagements for the extraction of Network based, Post based and User based features.

- ***Network Based Features*** include Stance Network which has tweets related to the news article as nodes and the weight indicates the similarity between them, Co-occurence Network which has users as the nodes and checks whether the posts published by those users are related to the same news articles, Friendship Network which include the followers/following of users who post the news related to same news articles and Diffusion Network which tells that a diffusion path exist between user u1 and u2, if u2 follows u1 and u2 posts a post only after u1 posts.

- ***Post Based Features*** include Post level features which deals with stance features (user's opinion on the post), credibility features, etc, Group level features which include the credibility of all the posts associated with the particular news article and Temporal level features which include the change of post level features over time.
- ***User Based Features*** include both Individual level features which consider user registration age, the tweets posted by user, the number of followers/ followings, etc to check the credibility and reliability of each user associated with a particular news article and Group level features which usually takes the average of the credibility of all the users associated with the particular news article.

Lastly we will have a brief overview of the existing datasets and evaluation metrics.

A few publicly available **Datasets** for the detection of fake news are **BuzzFeed-News** (News published in Facebook near to US presidential election for a week), **LIAR** (Dataset collected from the website PolitiFact), **BS Detector** (Data is collected from a browser extension called BS detector built for finding the authenticity of the news) and **CREDBANK** (Large scale crowdsourced dataset of approximately 60 million tweets).

Some of the commonly used **Evaluation Metrics** are:

$$\textbf{Precision} = \frac{|TP|}{|TP| + |FP|} \tag{1.1}$$

$$\textbf{Recall} = \frac{|TP|}{|TP| + |FN|} \tag{1.2}$$

$$\textbf{F1} = 2 \times \frac{Precision.Recall}{Precision + Recall} \tag{1.3}$$

$$\textbf{Accuracy} = \frac{|TP| + |TN|}{|TP| + |TN| + |FP| + |FN|} \tag{1.4}$$

- True Positive (TP): Predicted fake news are actually fake news
- True Negative (TN): Predicted true news are actually true news
- False Negative (FN):Predicted true news are actually fake news
- False Positive (FP): Predicted fake news are actually true news

The accelerating value of these evaluation metrics indicates that the performance of the model is increasing. Along with these measures, other measures such as **AUC** (Area Under Curve) which is the value from ROC (Receiver Operating Characteristic) curve which has the x-axis as FPR (False Positive Rate) and the y-axis as TPR (True Positive Rate), can be used as an evaluation metric for fake news detection. AUC is a good measure as it is more consistent and discriminating than accuracy.

CHAPTER 2

LITERATURE SURVEY

In order to get a clear and deeper insight on the existing models for the detection of fake news, some papers have been read by us that detect fake news and also related areas like rumors. In this chapter, we will describe some of those papers which we believe will help in building our own fake news detection model in the future.

2.1 Heterogeneous Graph Attention Networks for Early Detection of Rumors on Twitter

Most of the early rumor detection methods extracted features from user profiles, text contents and propagation patterns. But looking at the increasing growth and efficiency of deep neural networks in feature extractions, some of the models like recurrent neural network, tree-based recursive neural network, global-local attention networks are built to identify the semantic variations in the source tweet and the retweets but the limitation in all of these is that the global semantic relations of text contents are not considered which are however proved to be useful.
This paper uses masked subgraph attention mechanism and constructs a heterogenous tweet-word-user graph considering both the global semantic relations of the text contents and also the global structural information of the source tweet propagation and classifies the data as **Unverified Rumor (UR)** or **True Rumor (TR)** or **False Rumor (FR)** or **Non Rumor (NR)**, using the tweet information from **Twitter15** and **Twitter16** datasets along with information of the user related to the source tweets collected from the twitter API.

Here, we see the use of concept called **Attention**. This attention mechanism came as an improvement over the encoder decoder-based RNNs/ LSTMs. Encoder-decoder based RNNs/ LSTMs have two parts namely encoder LSTM/ RNN and decoder LSTM/

	Twitter15	**Twitter16**
Number of tweets	331,612	204,820
Number of users	276,663	173,487
Number of source tweets	1,490	818
Number of unverified-rumors	374	201
Number of true-rumors	372	207
Number of false-rumors	370	205
Number of non-rumors	374	205

Table 2.1: Statistics of Twitter15 and Twitter16 datasets

RNN. Encoder LSTM/ RNN reads the input sentence and encodes it into a context vector (summary) which is the last hidden state of LSTM/ RNN (rest all intermediate states are ignored) and passes it to the decoder vector which translates the input sentence just by seeing it. Here, we see that if the summary of the encoder is bad, the translation is also bad and this is usually seen when encoder is trying to understand longer sentences. It is the long-range dependency problem of RNN/LSTM which occurs due to the vanishing gradient problem. And also in LSTM/ RNN, there is no mechanism to give weights to the words based on their importance. All words are given equal importance. So, when we are trying to find the next word in a sentence, the prediction may not be proper. So, these issues are sorted out using the Attention mechanism which gives weights to the words or tweets based on their importance.

2.1.1 Graph Construction

As we are constructing heterogenous tweet-word-user graph, we have *G=(V,E)* where V is the set of source tweets $\mathcal{T}$, set of words W and set of users U and E is the set of edges (here, considering 3 types of edges namely, tweet-word E_{tw}, word-word E_{ww} and tweet-user E_{tu}). The edge weight between tweet and word is given using TF-IDF (Term Frequency Inverse Document Frequency) which gives the significance of a word to a tweet in a collection of tweets, the edge weight between two words is given using PMI (Pointwise Mutual Information) after finding the co-occurrence of words using a fixed size sliding window which is slided over all the source tweets in the datasets and the edge weight between tweet and user is given by taking the reciprocal of the time after

which the user replied or retweeted to a source tweet.

$$[H]A_{ij} = \begin{cases} \text{PMI}(i,j) & i,j \text{ are words, PMI}(i,j) > 0, \\ \text{TF-IDF}_{ij} & i \text{ is tweet, } j \text{ is word,} \\ \frac{1}{t+1} & i \text{ is tweet, } j \text{ is user,} \\ 1 & i = j, \\ 0 & \text{otherwise.} \end{cases} \tag{2.1}$$

where A_{ij} is the edge weight between node *i* and node *j* and *t* is the elapsed time when the user *j* replied or retweeted to a tweet *i*.

$$\text{PMI}(i,j) = log\frac{p(i,j)}{p(i)p(j)} \tag{2.2}$$

$$p(i,j) = \frac{\#W(i,j)}{\#W} \tag{2.3}$$

$$p(i) = \frac{\#W(i)}{\#W} \tag{2.4}$$

Figure 2.1: Point-wise Mutual Information Calculation

where #*W*(i,j) is the count of sliding windows containing both word *i* and word *j*, #*W*(i) is the count of sliding windows containing the word *i* and #*W* is the total count of sliding windows.

$$[H]\text{TF-IDF}_{ij} = \text{TF}_{ij} \times \text{IDF}_j \tag{2.5}$$

$$\text{TF}_{ij} = \frac{n_{ij}}{\sum_k n_{ik}} \tag{2.6}$$

$$\text{IDF}_j = log\frac{|\mathcal{T}|}{|\{k : w_\text{j} \in t_\text{k}\}|} \tag{2.7}$$

where n_{ij} is the count of a word *j* occurring in a tweet *i*, $|\mathcal{T}|$ is the total count of

tweets and $|\{k : w_{\mathrm{j}} \in t_{\mathrm{k}}\}|$ is the count of tweets that contain the word *j*.

In order to obtain the global semantic relation of text contents and the structural information involved in source tweet propagation, the heterogenous graph is divided into two sub-graphs (tweet-user and tweet-word). Edge weights in these sub-graphs is the same as the original graph.

The problem statement is to determine the probability of fake and real labels of the source tweet given the heterogenous tweet-word-user graph.

2.1.2 Subgraph Attention Network

Here, in the sub-graphs, the importance of each node's neighbours is found and the representation of these neighbours and the importance is merged to form each node's representation. In the constructed graph, X_{W} is a set of word-embeddings of the available words, X_{T} is the set of tweet representations where each representation is the average of the word-embeddings present in that particular tweet and X_{U} is the set of user representations where each representation considers certain user features extracted from user behaviours. It is noticed that the nodes in the tweet-user graph have different feature space while the nodes in the tweet-word sub graph are of same feature space. So, a transformation matrix is used in tweet-user graph to represent the tweet and user nodes into same feature space.

Now that we have tweet and user nodes in tweet-user subgraph and tweet and word nodes in tweet-word subgraphs respectively in the same feature space, weights between the nodes in respective sub-graphs is learnt using self-attention [3]. Using the self-attention mechanism, attention coefficients $\mathrm{e}_{i,j}$ are found which tells how important node *j* representation is to node *i*. Here, masked attention is used where attention coefficients $\mathrm{e}_{i,j}$ are calculated for a node i and node *j* where node *j* is the neighbour of node *i* including itself. For this purpose, a single-layer feed forward neural network is used and the LeakyReLU acts as the activation function. Then softmax function normalizes the coefficients. Now, the embeddings of nodes are updated by aggregating the neighbor representations of the nodes and their corresponding coefficients. This process is repeated for almost *k* times to learn a more stable embedding and finally, their learned

representation is concatenated to get the final output representation.

$$x_i^{'} = \|_{k=1}^{K} \sigma(\sum_{j \in N_i} \alpha_{i,j}^{k} W^k x_j) \tag{2.8}$$

where $x_i^{'}$ is the updated embedding of a node *i*, $\|$ is the concatenation operation, N_i is the set of node *i* and its neighbors, $\alpha_{i,j}$ indicates the normalized attention coefficients obtained by the k^{th} attention mechanism (f^k), and W^k is the corresponding weight matrix of input linear transformation.

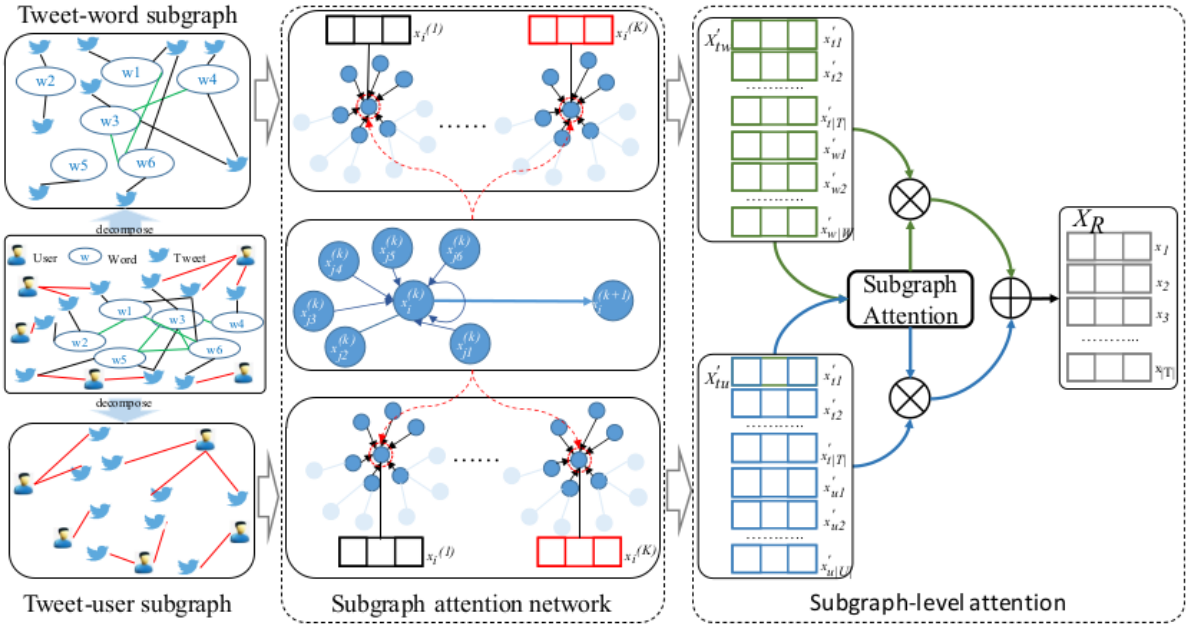


Figure 2.2: Rumor Detection Architecture of Heterogenous Graph Attention Networks

2.1.3 Subgraph-level Attention

So now, we have the updated node embeddings $X_{tw}^{'}$ of tweet-word subgraph and $X_{tu}^{'}$ of tweet-user subgraph with global information. Now, the information present in these two subgraphs is to be fused to accurately identify rumours. For this, subgraph weights are to be identified. This is done using subgraph-level attention mechanism. Initially, the importance of all the nodes in a subgraph is found using a feedforward neural network that performs subgraph-level attention and the average importance of all the nodes in a subgraph is taken as the importance of the whole subgraph. Then the normalized weights of the subgraphs are found using the softmax function. And finally, the two

subgraphs are fused by updating the source tweet embeddings using the learned weight coefficients.

$$x_i = \sum_{\Phi \in tw,tu} \beta_\Phi \cdot x_{t_i}, x_{t_i} \in X_\Phi^i \tag{2.9}$$

where x_i is the embedding of a node *i*, β_Φ is the normalized weight of the subgraph Φ, x_{t_i} denotes the representation of a tweet node *i* with global relation information in the Φ subgraph, and $X_\Phi^{'}$ is the representation of nodes with global relation information in the Φ subgraph.

Now that we have the set of source tweet embeddings, they are fed into a one-layer feedforward neural network and then a softmax function is used to normalise and thus class probability distribution of source tweets is calculated. The cross-entropy loss and a regularization term are used as the optimization function to train the model's parameters.

2.1.4 Results

This proposed model is tested on **Twitter15** and **Twitter16** datasets along with the user information of the source tweets which is extracted using Twitter API and it is seen that it gives an overall accuracy of **0.911** for **Twitter15** and **0.924** for **Twitter16** which is higher than some of the existing methods like DTC, RFC, SVM-TS, GLAN which uses local semantic relation and global structural information of the tweet propagation, etc.

Model	Twitter15 (Acc.)	Twitter16(Acc.)
All	0.911	0.924
Only tweet-word	0.813	0.810
Only tweet-user	0.560	0.690

Table 2.2: Analysis of Subgraphs

The proposed model is also compared with only tweet-user subgraph and only tweet-word subgraph as shown in the table above. It is also observed that the proposed model gives highest accuracy compared to individual subgraphs.

2.2 Hong Kong Protests: Using Natural Language Processing for Fake News Detection on Twitter

The objective was to find out:

- How going solely by NLP helps in the problem of Fake news detection in text data
- Is only NLP enough for fake news detection
- To get an idea of what more features other than linguistic features can be added for the detection

This paper was chosen as it considers only linguistic features from the dataset for the modelling. Also most of the datasets used in papers that focused on NLP were looking at only news articles and not tweets. The language used in news articles is different from tweets, even the length of news articles is much longer than tweets so we had to find a dataset that used social media posts and not news articles. Therefore this paper was chosen as it uses a dataset that has social media tweets posted by users.

2.2.1 Dataset used

This paper analysed tweets related to Hong Kong protests that were tweeted when these protests were happening.

For the fake portion of the dataset, they have used a publicly available Twitter dataset which had fake tweets related to HongKong protests.

For the real portion of the dataset, they manually created a list of credible news agenices like BBC News, Reuters etc and also along with that a list of credible journalists, the tweets from these news agenices and journalists were collected using Twitter's user timeline API.

The dataset they used contained English and Chinese tweets, the Chinese tweets were translated to English and kept along with the original English tweets.

2.2.2 Features that were used in the model

The features they have used is purely linguistic in nature. After feature selection they have narrowed down to 10 features from the inital 38 features they had considered. The features used are:

- Tweet length
- Type to token ratio(which is calculated after tokenizing the tweets, indicating the richness in vocabulary used by the tweet)
- PoS features-(Part of Speech features which keep track of adverbs, verbs, entities, pronouns etc in the tweet). These features were obtained using the Natural Language Toolkit module(NLTK)
- Tweet sentiment that is calculated using the AFINN word list
- Tweet entropy (indicator of word importance for a tweet)

2.2.3 Algorithms that the paper used on the dataset

C4.5,SVM,Naive Bayes and Random Forests of C4.5 are used for training and testing the dataset. In all the above algorithms, above 85% F1 score is achieved. Random Forest was the best of all of them with F1 score of 92.4%.

2.2.4 Conclusions derived

Although the algorithms showed really high F1 scores, a major issue was in the translation of Chinese tweets to English tweets. When the tweets are translated sometimes they might not be accurately translated and also the number of words after translation can be higher than original English tweets. A language like Chinese is very intricate that even a single symbol in Chinese can correspond to a word in English. For example a tweet in the Chinese language which has 125 characters will have 585 characters after converting into English. And since we are looking at aspects like tweet length, word count, etc., these translated tweets can lead to results that may not be accurate. So therefore by reading the paper it was concluded that going solely by NLP will not help in the problem of Fake news detection especially when you have different languages

other than English. The paper suggests to include network based features and user level features as well, this may help in improving the accuracy.

2.3 Weakly Supervised Learning for Fake News Detection on Twitter

After reading the paper on HongKong protests, the objectives to find a suitable method for Fake News Detection were tweaked.
The objective was to find out:

- Along with NLP can user account features help in coming up with a model which is more diverse.
- Can we use a large dataset with low quality labels? Will the large size of the dataset compensate for the low quality labels?
- Is it possible to detect fake news for various events and not specifically for only a particular event (For example, the HongKong paper only looked at one event: the HongKong protests)

2.3.1 Dataset used

In this paper they are initially assuming that a trustworthy source is more likely to give a Real News and untrustworthy source is more likely to put out Fake News. Collection of Fake news sources and Real news sources was taken from various web catalogs. Using the Twitter API the tweets from these sources were retrieved. All the tweets that were collected are from the year 2017. The reason behind this is Twitter API only returns recent 3200 tweets from a particular account.

2.3.2 Reducing Noise in the Dataset

It was expected that there will be a possibility of Untrustworthy sources putting out real news as well. On inspecting a sample of data from the fake news class, they found out that the fake news class consisted of only 15% of actual fake news tweets, 40% of them was real news tweets and the rest undecided.

However since both the real class samples and fake class samples were taken from the year 2017 so they have assumed that a news that's actually real but labelled as fake in the Fake news class will most likely be present in the Real News class with label Real news (Not fake news).

Due to this issue they have taken an extra added precaution of collecting more data from real news sources, therefore by having a larger real news class, the classifier is more likely to classify any news that's actually real but labelled as fake in the Fake class to be classified as Real by the algorithm.

On inspecting a sample of the Real news class, a small percentage of actual fake news was labelled as real. In order to prevent this, 116 tweets from 2017 labelled as Fake were collected *Politifact*, a very well known website where tweets are annotated as fake or true by expert journalists. Any tweets in the real dataset that was similar to the 116 tweets were removed. They found out the similarity by using two measures *TF-IDF* and *Cosine similarity*.

Turns out that these assumption works out as the paper shows that despite a not highly accurate but large dataset it is possible to detect fake news that too with a high F1 score of 0.91.

2.3.3 Features used in the model

Five different group of features were considered:

- User level features
- Tweet level features
- Text features
- Topic features
- Sentiment features

Some of the user level features were number of followers, tweets frequency and the retweet ratio etc.

The tweet-level features were word count, ratio of exclamation and question marks. For extracting text level features from the tweets, Bag of words model using TF-IDF and a neural Doc2Vec model is used.

Certain topics are more prone to have fake news spread than others, hence the topic level features are also considered. Two processes have been used for this, the Latent Dirichlet allocation model as well as the Hierarchical Dirichlet Process for topic modelling.

Finally for the sentiment features, to calculate the polarity of the tweets in terms of ratio of positive words, negative words and neutral words present in the tweet using *SentiWordNet* module.

2.3.4 Algorithms used and Evaluation

Random Forest and XG Boost was used for classifying the tweets as Fake or Real. First they have tried to model without including user features and the XG Boost algorithm achieved the highest F1 score of 0.78. After inclusion of user level features, the XG boost algorithm's F1 score is **increased** to 0.94.

2.3.5 Conclusions derived

From this paper we can see that despite using a large dataset that is not of high quality, when we take the right measures to clean the data and reduce the noise and when we choose the appropriate features, the F1 score will be high. The inclusion of user level features giving a higher F1 score is a motivation to look further into the user accounts, the patterns in which the user uses his/her account etc. This will help us more when we have tweets that are of a different language, since user level features do not consider the language of the tweet at all unlike text features. Also now we know that we don't need a dataset that pertains to one event alone but can be comprising of many events that happened at a particular time period.

2.4 CSI:A Hybrid Deep Model for Fake News Detection

There are 3 characteristics of fake news are: the *text* of the article, the *source users* promoting it and the *user* response it receives. This paper proposes a model that combines all these three characteristics for accurate prediction. This paper has incorporated the behaviour of both parties, users and articles and the group behaviour of users who propagate the fake news.

Two datasets have been used namely: Weibo dataset and Twitter dataset. These datasets include all the three characteristics: , *text*,*response* it receives and *user information* of the user interacting with the article. Each dataset has news articles with labels fake or real and each article also has a set of tweets made by the users at a particular instant of time.

2.4.1 Objective of the paper

- Classifying fake news accurately
- Identification of groups of suspicious users who play a role in spreading of fake news

Given a temporal sequence of user engagements(tweets/posts), where e_{ij} is the engagement, u_i is the user whos carrying out this engagement, a_j is the article this engagement is carried out on and t is the time at which this engagement happened.

$$e_{ij} = (u_i, a_j, t)$$

The goal of the paper was to produce a label $\hat{Y}_j$ that can be 0 or 1 for each article a_j and a suspiciousness score s_i for each user.

2.4.2 Model proposed

They have proposed a model called **CSI** which consists of two parts, a module for extracting the temporal representation of the news articles (by looking into the response the article receives and the text of the engagement that is linking this article) and an-

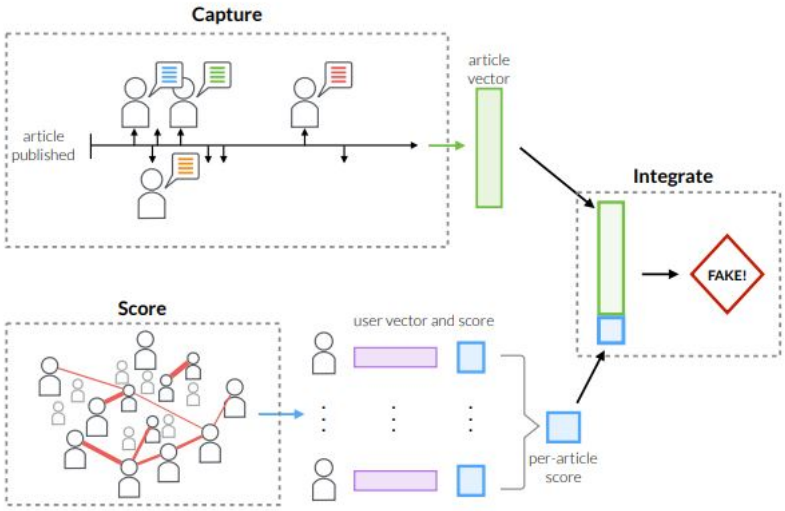


Figure 2.3: Intuition behind CSI

other module for scoring the behaviour of the users this helps in capturing the source characteristic of the article. When we look at it specifically, the CSI model is composed of three modules: **Capture, Score and Integrate**. The final output will be obtained from the Integrate module, this output will classify the article as fake or not.

2.4.3 Capture phase

This module is based on the response the article receives and the text of the article. It seeks to "capture" the temporal engagement of users with respect to an article i.e., the engagement(social media posts/tweets etc) of the user with an article a_j across time t. This is done by capturing how many users are engaging with the particular article and how these engagements were spaced over time.

Initially for each article a feature vector $\mathbf{x_t}$ is constructed.

$$x_t = (\eta, \Delta t, x_u, x_\tau)$$

This feature vector consists of the following:

- η = the number of engagements with this particular article a_j
- Δt = time between the engagements with this article a_j
- a feature vector $\mathbf{x_u}$ which depicts the source users promoting the article. This feature vector is not specific to an article it is global. To obtain this feature vector

a binary incidence matrix is constructed between the users and articles they are engaging with. Then SVD is applied to get a lower dimensional representation for each user u_i. SVD is a matrix decomposition method used for compressing, denoising and data reduction.

- a feature vector $\mathbf{x}_\tau$ that carries the textual characteristic of the posts with a given article a_j, for building this feature vector they have used *doc2vec*(an NLP tool) on the text of each post. doc2vec is used to create a vectorised representation of a group of words taken collectively as a single unit.

Since the temporal and textual features come from different domains, it is incorporated into the Recurrent neural network only after standardizing. x_t is passed to an embedding layer to get $\tilde{x}_t$

Once these features are standardized they are fed into a Recurrent Neural Network using the Long short term-memory model.

The Long short-term memory is a type of Recurrent Neural Network architecture. Unlike feedforward neural networks Long short-term memory has feedback connections i.e., they can not only process single datapoints (Eg. images) but also entire sequences of data (video,speech etc). LSTM was created as a solution for the short term memory problem of RNN's. They solve this problem with the help of internal gates which learn to keep the important data(even if it arrives in the beginning) and pass it down the long chain of sequences to make the prediction. LSTM is composed of a cell,an input gate,an output gate and a forget gate These cells remembers values in arbitrary periods of time and the three gates regulate the flow of information into and out of the cell.

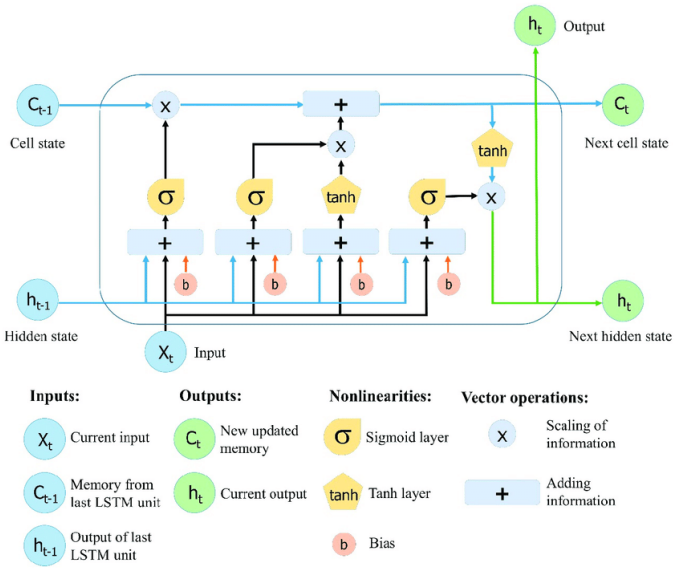


Figure 2.4: Structure of LSTM

The output received from the LSTM is a vector $\mathbf{v_j}$ After this the vector $\mathbf{v_j}$ will be fed to the Integrate module along with output from the Score module.

2.4.4 Score Phase

The objective of the Score module is to come up with a score for every user that will indicate his/her inclination towards participating in promoting these articles.

Initially a weighted graph called the user graph is constructed where the nodes will be the users and the edges will denote the number of articles two users engaged with. SVD(Singular Value Decomposition)is applied onto the adjacency matrix to get a low dimensional feature vector called $\mathbf{y_i}$. Therefore each user u_i has a set of user features associated with it that is represented as y_i. After applying weights and bias on y_i, a feature vector $\mathbf{\Lambda_i}$ is constructed for each user u_i.

To aggregate this information, a weight vector w_s is applied to produce a scalar score s_i for each user as:

$$s_i = \sigma(w_s{}^T . \mathbf{\Lambda_i} + b_s)$$

Eq 1.1 Sigmoid function applied to get score s_i

2.4.5 Integrate Phase

In this module article representations $\mathbf{v_j}$ is combined with user scores s_i to produce a label $\hat{Y}_j$ for each article,if $\hat{Y}_j$=0 then the article is true,if $\hat{Y}_j$=1 then its fake.

The set of scores s_i for all users is represented as **s**. In order to integrate the two modules, from **s** only the scores s_i of users who interacted with particular article a_j is considered. These scores are then averaged to produce p_j which will be the **"suspiciousnesss"** score of the users that engaged with the specific article $\mathbf{a_j}$.

Next the score p_j is concatenated with $\mathbf{v_j}$ and this results in vector $\mathbf{c_j}$. This vector is then fed into a fully connected layer (where sigmoid function is applied after applying weights and bias) to get the final output $\hat{Y}_j$

After the output from the Integrate module is obtained the training and testing of the dataset is done with a loss function that compares output from integrate model and the ground truth labels that already came with the dataset after this back propagation is done to change the weights to minimise loss

$$Loss = \frac{-1}{N}\sum_{j=1}^{N}[L_j log\hat{L}_j + (1 - L_j log(1 - \hat{L}_j))] + \frac{\lambda}{2}\left\|W_u\right\|_2^2$$

Eq 1.2 Loss function

2.4.6 Observations made:

Ability of Suspiciousness score to capture user behaviour

They tried to find out if suspiciousness score of users interacting with an article is able to capture l_i(if user only has interacted with fake articles then l_i will be equal to 1.0 if user interacts with only real articles s l_i will be equal to 0.0) and the results they obtained were that there is a positive correlation between l_i and s_i indicating that s_i does capture the behaviour of the user, whether he/she has a tendency to spread only fake articles or only real or somewhere in between.

Ability of Suspiciousness score to capture group user behaviour

The next step was to evaluate if s_i and $\hat{Y}_j$ are able to capture the group behaviour. A user

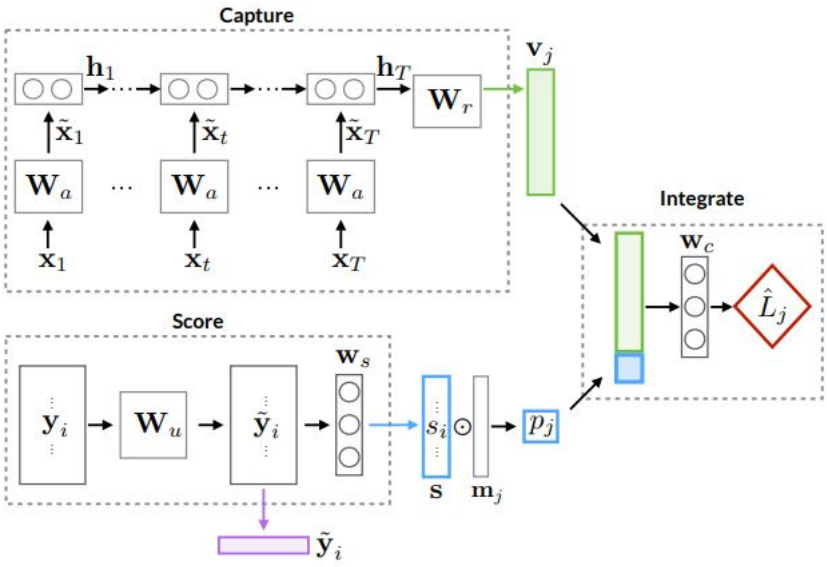


Figure 2.5: CSI Model depiction

graph with edges between users engaging in same article is made, and then they attempt to analyse cluster of these users. After applying BiMax algorithm on the adjacency matrix of the graph, they find out that users who have a large l_i tend to participate in larger clusters than users with a low l_i.

Another very interesting observation made is done by looking at the lag which is the time difference between when the article was put out and when the user tweeted it. Then lag and suspiciousness scores are plotted. It was observed that when the article is first published the most suspicious users are the ones who engage with the article i.e. post about the article or tweet about it first.

The accuracy and F1 score were found to be 0.892 and 0.894 for the Twitter dataset, 0.953 and 0.954 for the Weibo dataset On comparing with the state-of the-art models , DT-RANK,SVM-TS, LSTM-1,DTC, and GRU-2[4], the model that was presented in this paper achieved a higher accuracy and F1 score.

2.5 Multi-Depth Graph Convolutional Networks for Fake News Detection

This paper proposes a multi-depth Graph Convolutional Network model to classify news into one of the 6 categories: pants-fire, false, barely-true, half-true, mostly true and true.

The dataset used is LIAR[5] which contains a large number of speaker profiles, such as speaker name, party affiliations, job title, home state, location of speech, topics and credit history.

2.5.1 Objectives of the paper

- The speaker profiles on the LIAR dataset are to be represented using graph networks and the correlation between two news is to be captured, to improve the performance of fake news detection.
- The GCN is to be expanded to gather multi-scale information of neighbours residing at various depths.

2.5.2 Problem Definition

Let $D = d_1, d_2, ..., d_{|D|}$ be the news set with $|D|$ news, where each news *i* contains text content representation x_i. q_i^t represents the speaker profile t. $Y = \{y_1, y_2, ..., y_c\}$ denotes a set of class labels. Each news article is considered to be a unique node in the network. For a given graph $G = (V, E)$ with $N = |V| = |D|$ nodes and the edge set E, $A_t \in R^{N \times N}$ is the adjacency matrix(binary or weighted) built based on $Q_t = q_1^t, q_2^t, ..., q_{|D|}^t$ and $X \in R^{N \times F}$ represents feature matrix.

Labels for a subset of nodes $V_L \subset V$ are known. The aim is to train the model M which assigns labels to all unlabeled nodes $V_U = V - V_L$ using the feature matrix X and known labels for nodes in V_L.

2.5.3 Model proposed

The paper suggests a multi-depth GCN model which consists of three parts: node feature, multi-depth input matrix generated by one kind of relationship among nodes and the output components.

Firstly, edges are built among nodes using the information about them belonging to the

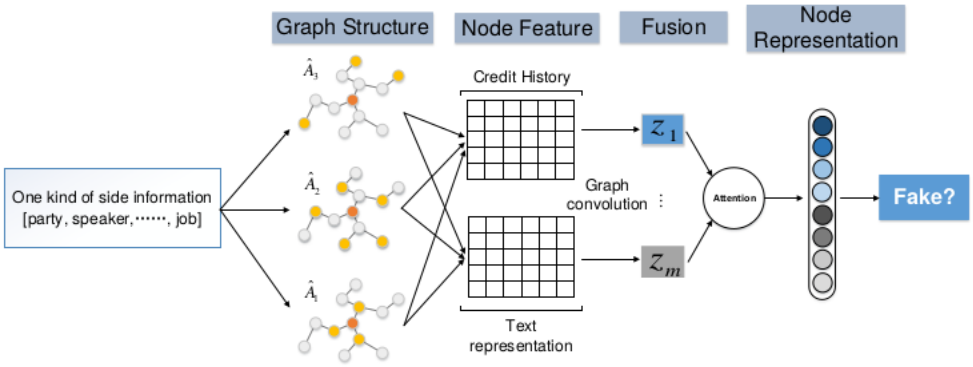


Figure 2.6: M-GCN model; yellow node is a neighbour of red node at different distances.

same group or not. When job-title information is considers, the sparse adjacency matrix A is:

$$A_{ij} = \begin{cases} 1 & \text{if } i, j \text{ have the same job-title} \\ 0 & \text{otherwise} \end{cases} \tag{2.10}$$

where *i*, *j* are the different news entities.

One of the existing graph convolution methods, namely the one used in spectral GCN[6] is used to model the relation matrix and node features.

Different distance proximity matrices are calculated instead of stacking the GCN layers to combine the long distance information. This is done to have a description of the correlation between nodes and preserve the multi-granularity information in explicit way. The k-order proximity matrix $^{k}_{ij}$ is the k-step proximity between node v_i and v_j where $^{k} = \times \cdots$ (i.e, is multiplied with itself *k* times).

The multi-scale information of neighbours is stored in the respective step proximity matrix. Using the proximity matrices, the GCN layers are modelled directly and the training process can be sped up.

Different depth proximity matrices are fed to the GCN to get the following output(z_k) at each step.

$$z_k = {}^{k}ReLU({}^{k}XW_k^{(0)})W_k^{(1)}, k = 1, 2, 3 \ldots \tag{2.11}$$

Here, $W_k^{(0)}$ denotes the weight matrix for one hidden layer and $W_k^{(1)}$ is output weight matrix.

This multi-depth information is aggregated by attention mechanism[7] to form the final representation. All the multi-depth outputs are fed through a non-linear project to acquire corresponding attention scores $u_i(1 \leq i \leq m)$.
Then every attention score is normalized using softmax function.
Each news is finally represented as P_j , which is the weighted average of all z_i (weighted using the normalized attention scores).
Finally, cross-entropy error on labeled samples is evaluated as,

$$L = -\sum_{l \in Y_L} \sum_{f=1}^{C} [Y_{lf} \times ln(P_{lf})] \tag{2.12}$$

where Y_L is the set of labelled nodes and *C* is the dimension of final representation which is nothing but the #class labels.

The above process is repeated with a goal to minimize the error for fake news detection, which satisfies the objective of the experiment.

2.5.4 Experimental results

- First, to evaluate the fake news detection performance on LIAR dataset, the model proposed in this paper is compared with the latest fake news detection methods such as:
  - **Hybrid-CNN**[5]
  - **LSTM-Attention**[8]
  - **Memory-Network**[9]
  - **MMFD**[10]
  - **GCN**[11]

  Below are the results of fake news classification on LIAR dataset according to different methods.

Model	Detection Accuracy(%)
Hybrid-CNN	27.4
MMFD	38.8
LSTM-Attention	41.5
Memory-Network	46.7
GCN	45.3
M-GCN	**49.2**

Table 2.3: Performance of various models on LIAR

- Second, the effect of each speaker profiles information, Speaker, Party, Topic, State and Job, is analysed by performing experiments over the proposed model with one kind of speaker profiles at a time with depth K = 2. The results are shown below. Compared to others, the "Speaker" user profile performs better in test dataset and returns the highest accuracy(49.2%) and F_1 score(49.7%) than other user profiles.
- Lastly, the role of K, depth of neighbour, is analysed for model performance. The below plot presents the changes in accuracy. With the depth K varying from 0 to 3, most of the features reach the max when K is 2. One reason may be because of lack of rich relational information. Most of the relationships in LIAR dataset contain 2 or 3 variables only, and most neighbour nodes can be reached with 1 or 2 hops, so it may introduce noise if K is greater than 2.

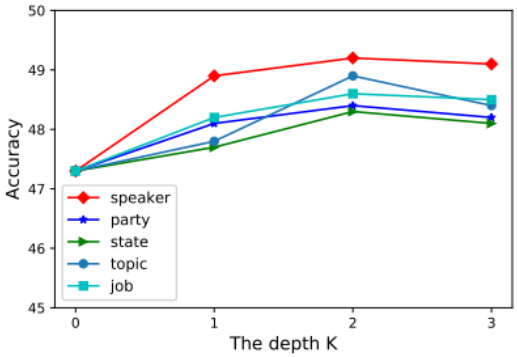


Figure 2.7: Performance with different depth K

2.5.5 Conclusions derived

Instead of encoding the user profiles directly, the multi-depth GCN model proposed in this paper represents each news node with graph structure information derived from the

speaker profiles. The GCN is expanded to capture the multi-scale information of neighbouring nodes inorder to use their features and improve the classification performance. This paper shows that the proposed model is better than the existing models and also says that it can make an even better use of node features and relation graphs alongside the existing inputs.This is because if a particular relationship has large number of variables, it may perform better by adding more information from the dense graph structure.

2.6 Network-based Pattern-driven Approach for Fake News Detection

This paper considers the friendship network where the nodes represent users/spreaders of news and edges represent the relationship among the users.

Two types of subgraphs of the social network are considered namely Fake News Network and True News Network

Fake News Network (FNN): It is a subgraph $G_F = (V_F, E_F)$ of the social network $G = (V, E)$, where $V_F \in V$ are the users(user identities or user attributes(susceptible or normal)) that have engaged with fake news F , and $E_F \in E$ represents the relationships among these users.

On similar lines, True News Network (TNN) is denoted as $G_T = (V_T, E_T)$ for a true news event T.

2.6.1 Objectives of the paper

- The main aim is to investigate fake news patterns in social networks. Their existence is proved based on empirical evidences and social psychological theories.
- The next objective is to represent the proposed fake news patterns and quantify them across multiple network levels, i.e., node, ego, triad, community, and the overall network level, resulting in formulating various features which are further used within supervised machine learning frameworks to detect fake news.
- Lastly, experiments are conducted on real-world data to demonstrate that the developed model which uses various network level features performs stably and

better compared to the state of the art.

2.6.2 Fake News Patterns

The fake news patterns involved in this paper speak about (1) the news being spread, (2) spreaders of the news, and (3) relationships among the news spreaders.

- *More-Spreaders Pattern:* More users spread fake news than true news.
- *Farther-Distance Pattern:* Fake news spreads farther than true news.
- *Stronger-Engagement Pattern:* Spreaders engage more strongly with fake news than with true news.
- *Denser-Network Pattern:* Fake news spreaders form denser networks compared to truth spreaders.

The validations to more-spreaders pattern and farther-distance pattern are provided in [12]. The statistics in [13] support stronger-engagement pattern. The social psychological theories provided in [14], [15], [16] and [17] are in support of the denser-network pattern.

2.6.3 Features extracted from the patterns

- *More-Spreaders Pattern:*

  The More-Spreaders Pattern can be quantified by the number of news spreaders involved in spreading each fake or true news event.
  Here, users are categorized as general (Non-Attributed) Spreaders and Specific (Attributed) Spreaders.

  1. *General (Non-Attributed) Spreaders:* The total #nodes in a network is nothing but the #spreaders of a given news article.
     FEATURES: #news spreaders.

  2. *Specific (Attributed) Spreaders:* "User susceptibility to fake news" and "user influence" are considered as attributes of specific users.

     (a) *User susceptibility:* User susceptibility to fake news is quantified based on (i) the number of involvements in spreading different fake articles and (ii) the frequency of such involvements.
     User susceptibility in terms of involvements is the ratio of number of

fake news to all the news articles that user v_i has spread, which is denoted as $S(v_i)$.

$$\mathbf{S}(v_i) = \frac{\sum_j \mathbf{B}(v_i \in V_{F_j})}{\sum_k \mathbf{B}(v_i \in V_{T_k}) + \sum_j \mathbf{B}(v_i \in V_{F_j})} \tag{2.13}$$

Here, $\mathbf{B}(*) = 1$ if * is true, else $B(*) = 0$.
User susceptibility in terms of frequency of involvements is the proportion of the spreading frequency of fake news articles among all news articles a user has spread.

$$\mathbf{S}(v_i) = \frac{\sum_j \mathbf{B}(v_i \in V_{F_j})\mathbf{T}(v_i, F_j)}{\sum_k \mathbf{B}(v_i \in V_{T_k})\mathbf{T}(v_i, T_k) + \sum_j \mathbf{B}(v_i \in V_{F_j})\mathbf{T}(v_i, F_j)} \tag{2.14}$$

$\mathbf{T}(v, X)$ represents spreading frequency of user *v* for news event *X* where *X* can be fake news event *F* or true news event *T*.
After calculating susceptibility score $\mathbf{S}(v_i)$ of each user, they are then labeled as susceptible if $\mathbf{S}(v_i) > \theta$ and normal if $\mathbf{S}(v_i) < \theta$ based on a fixed threshold value $\theta \in [0, 1]$.
FEATURES: #normal Spreaders, #susceptible spreaders, %normal spreaders, %susceptible spreaders, average spreader susceptibility and median spreader susceptibility, where susceptibility $S(v_i)$ is based on #news or frequency.

(b) *User influence:* User influence is approximated using centrality score of each node based on the criteria (a) [in-, out-] degrees, (b) betweenness, (c) [in-, out-] closeness, (d) hub and authority score, (e) PageRank score. All these criteria identify the positions of the nodes in the network using their connectivity.
FEATURES: mean and median user influence.

- *Farther-Distance Pattern:*

  This pattern is quantified using the distance that a news article can spread in a network. This distance is computed in two ways.

  1. *Geodesic distance:* The shortest distance between the two nodes which are farthest of all in the network.
     FEATURES: Average, median and max Geodesic Distance.

  2. *Effective distance:* The FNNs and TNNs considere at the start are unweighted. They are converted to weighted ones, where the weights are calculated using the amount of information flow from one node to another.
     Effective Distance is defined as follows. Given a network G, let F denote the flow matrix whose entities represent the amount of information flow from one node to another. Based on the flow matrix, the effective distance $d_{Eff}(i, j)$ from node i to node j is defined as $d_{Eff}(i, j) = 1 - \log F_{ij}/\Sigma_l F_{lj}$

where $d_{Eff}(i,j)$ satisfies $d_{Eff}(i,j) \geq 1$.

Information flow in the fake and true news networks is the news flow among news spreaders in the network which is defined as (i) the total count of news articles both users have spread or (ii) the total number of times both users have at least spread the same news articles.
FEATURES: Average, median and max Effective Distance.

- *Stronger-Engagement Pattern:*

  This pattern is quantified based on two categories, group engagements and individual engagements.

  1. *Group Engagements:* At group level, #engagements for a given news article can be similar to counting the overall #times that the news article is spread. Here, user engagements are seen through the two types of users.
     (a) *General (Non-Attributed) Spreaders:*
     FEATURES: #User Engagements.
     (b) *Specific (Attributed) Spreaders:*
     FEATURES: #normal user engagements, #susceptible user engagements, %normal user engagements and %susceptible user engagements.

  2. *Individual Engagements:* This type of engagements of a news article can be evaluated using the average spreading frequencies of each type of users who have participated in spreading the news article.
     FEATURES: avg. normal user engagements, avg. susceptible user engagements and average user(susceptible+normal) engagements.

- *Denser-Network Pattern:*

  To find out the *density* of connections among the users, analysis is performed at various network levels: (I) ego, (II) triad and (III) community levels.

  – *Ego level:* Each edge in the network is considered to be an ego relationship. The density of networks formed by spreaders who participated in propagation of a particular news article is computed using the following features based on the below classification.

    1. *General Ego Relations:* Here, the users are considered to be non-attributed.
    FEATURES: #ego Relationships among Spreaders, average #ego Relationships of Spreaders and ego density.

    2. *Specific Ego Relations:* Each node in the network is labelled as susceptible/normal and edges are considered to be directed relations. Based on this, the edges are divided into 4 subgroups (1) E_{NN}- relations from

a normal to a normal user, (2) E_{NS}- relations from a normal to a susceptible user, (3) E_{SN}- relations from a susceptible to a normal user, (4) E_{SS}- relations from a susceptible to a susceptible user.
FEATURES: #/% N $\rightarrow$ N, #/% N $\rightarrow$ S, #/% S $\rightarrow$ N and #/% S $\rightarrow$ S where S and N denote susceptible and normal users respectively.

Also the edge set can be classified as follows: (1) $E_{\Delta>0}$ if $\Delta = S(v_i) - S(v_j) > 0$, (2) $E_{\Delta=0}$ if $S(v_i) - S(v_j) = 0$, (3) $E_{\Delta<0}$ if $S(v_i) - S(v_j) < 0$.
Based on this classification,
FEATURES: #/% edges where $S(v_i) > S(v_j)$, #/% edges where S$S(v_i) = S(v_j)$ and #/% edges where $S(v_i) < S(v_j)$.

– *Triad Level:* Triad is a set of 3 connected users in a graph/network. Similar to the analysis at ego level, the paper looks into (i) general triads and (ii) specific triads formed between normal and susceptible users within networks. The classification of users and features based on that are as follows.

1. *General Triads:* The features extracted at this level are as below.
FEATURES: #Triads, Average #Triads and Triad Density.

2. *Specific triads:* There can be 12 different triads when the users are classified as susceptible and normal as depicted below.

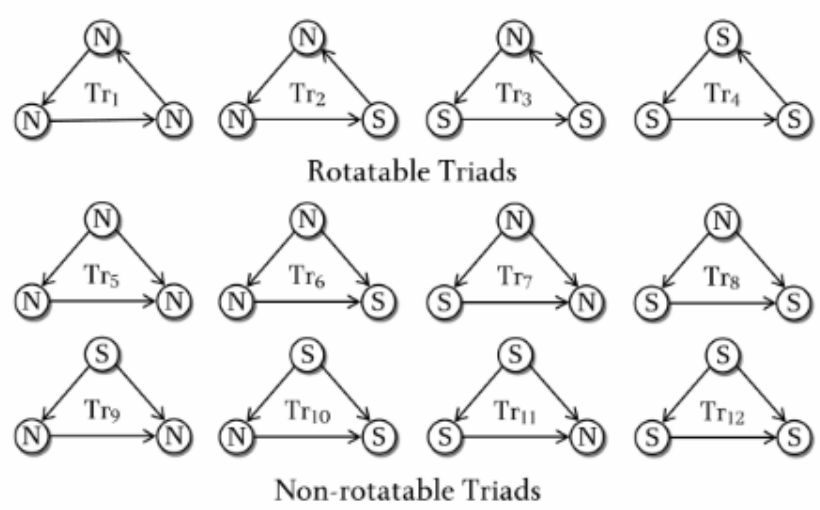


Figure 2.8: Specific Triads. N and S denote normal and susceptible users respectively. A $\rightarrow$ B indicates A follows B.

FEATURES: #triads (from Tr_1 to Tr_{12}) and %triads (from Tr1 to Tr12).

– *Community Level:* In networks, groups of nodes that are more densely connected internally than with the rest of the network are termed as communities.
These communities are seen through two perspectives. From a global view,

communities that nodes belong to within a fake or true news network(which inturn is a subgraph of the social network) are based on the structure of the entire social network. Communities can be found within a fake or true news network in local view.
FEATURES: #Communities and Community Density, both from global and local perspectives.

Communities are found using Louvain algorithm[18].

2.6.4 Experimental Setup

News articles collected from BuzzFeed and PolitiFact are used as dataset. Fact-checking experts provide the ground truth labels for these articles.

Following dataset collection, value of each feature is calculated for both datasets.

Experiments are conducted by using the features set in various supervised learning algorithms including SVC, KNN, NB, DTC and RF.

2.6.5 Performance Evaluation

- *General Performance Evaluation:*
  The results obtained by various supervised learning frameworks are shown. It is observed that classifier that's performing best on both datasets is RF, with 0.93 accuracy on PolitiFact and 0.84 F_1 score on BuzzFeed.

Classifier	**PolitiFact**		**BuzzFeed**	
	Accuracy	F_1 score	Accuracy	F_1 score
Support Vector Machine (SVM)	.871	.886	.830	.833
k-Nearest Neighbours (k-NN)	.875	.888	.752	.784
Naive Bayes (NB)	.788	.767	.748	.767
Decision Tree (DT)	.733	.662	.714	.630
Random Forest (RF)	.929	.932	.835	.842

Table 2.4: Performance of different classifiers

- *Performance of Patterns in Fake News:*
  Each fake news pattern's performance and their combinations are analysed. An observation is that when combinations of different patterns are considered as features, the performance is getting better than when single pattern is used. Hence

using all the patterns yields the best result among all other options. The corresponding accuracies and F_1 scores are shown below.

Pattern(s)	PolitiFact		BuzzFeed	
	Accuracy	F_1 score	Accuracy	F_1 score
DENSER-NETWORKS	.746	.718	.687	.704
STRONGER-ENGAGEMENT	.898	.898	.807	.808
FARTHER-DISTANCE	.639	.587	.678	.698
MORE-SPREADERS	.891	.901	.808	.817
All Patterns - DENSER-NETWORKS	.908	.916	.814	.819
All Patterns - STRONGER-ENGAGEMENT	.929	.928	.819	.815
All Patterns - FARTHER-DISTANCE	.913	.914	.780	.759
All Patterns - MORE-SPREADERS	.879	.871	.802	.803
All Patterns	.929	.928	.828	.823

Table 2.5: Pattern performance in detection.

- *Analysis over Importance of Each Feature:*
  Each feature is ranked according to their contribution in fake news detection. From the analysis, it is observed that the features taken from More-Spreader Pattern, Denser-Network Pattern and Stronger Engagement Pattern play a significant role in classifying fake and real news compared to the other features.

2.6.6 Conclusion

Combining social psychological theories and empirical studies, the work discussed in this paper can help strengthen the understanding of fake news by studying various patterns in the nets. Experiments showed that the proposed approach performs better than many of the existing methods.

The proposed model may/may not detect fake news before it propagates on social media but is robust to the possible variations/changes in styles of writing. Lastly, the proposed approach can be further improved by adding user profile attributes more patterns.

CHAPTER 3

PROBLEM STATEMENT

The aim of this project is the classification of news articles as fake or real based on the tweets related to that particular news article and the profile of the user who sent the tweet. The need for classification arises due to the disinformation that is being spread by fake news, therefore detecting the fake news and terminating the spread of the fake news is extremely important.

Objectives:

1. Collecting news articles and their respective tweets which are labelled as Fake or Real as well as information about the users who participated in the propagation of these tweets.
2. Performing Feature Engineering to extract the features from the tweet related data of news articles.
    - Extracting text level features
    - Extracting temporal features
    - Extracting user profile features
3. After obtaining the dataset with required features, splitting it as train and test and then performing supervised learning on the training data by using multiple classification algorithms.
4. From these multiple algorithms, finding the algorithm that gives the best accuracy on the train and test set.
5. Using the predictions obtained by the algorithm which gives the best accuracy to classify the news articles related to those tweets as Fake or Real.

CHAPTER 4

DATASET

Our objective was to find a dataset with news articles and their respective tweets which are labelled as Fake or Real along with the user information of the user who participated in the propagation of the tweet related to the news article.

4.1 Choosing the dataset

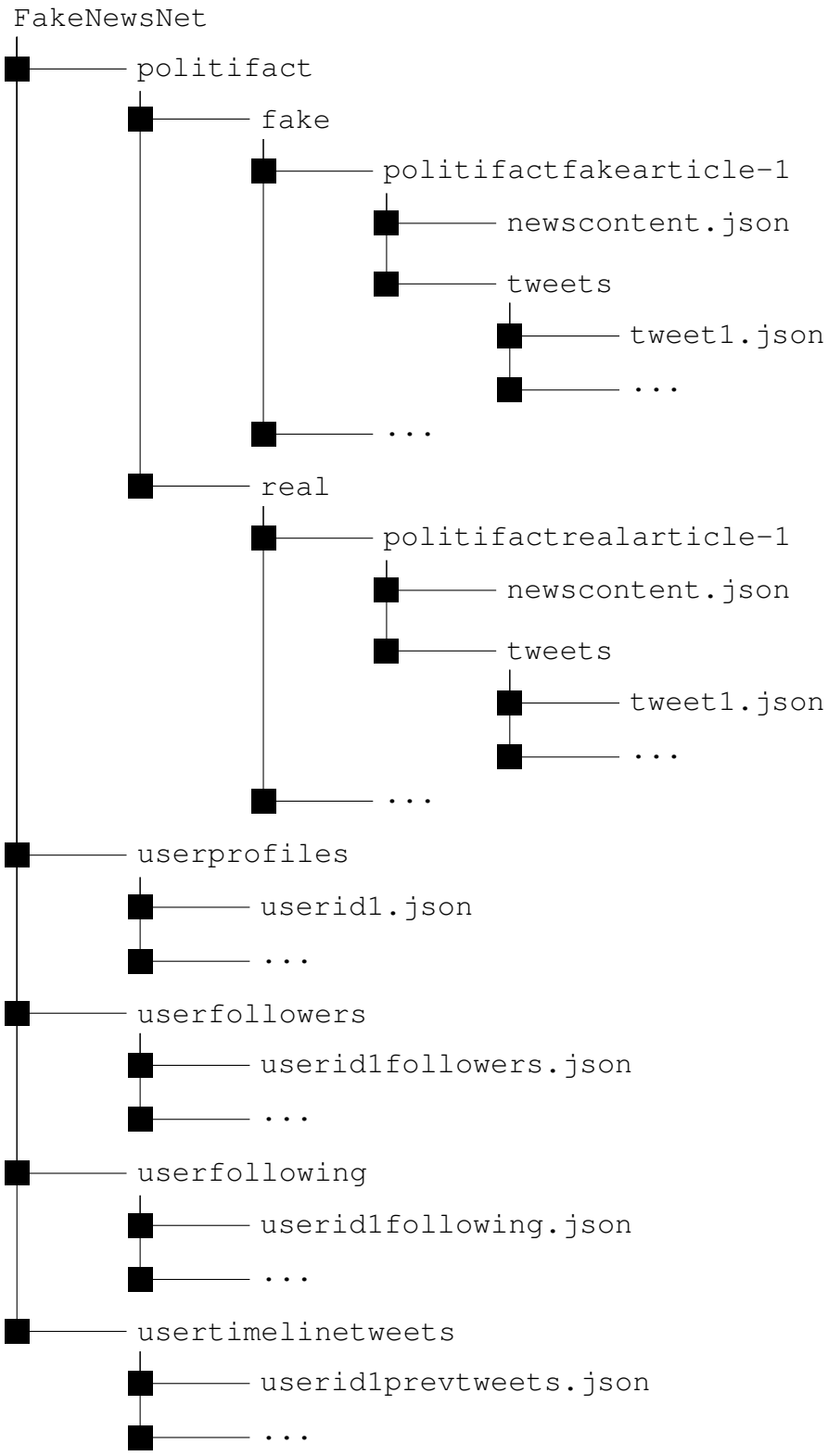
While exploring for this kind of dataset, we came across the following four different data sets.

- **Coronavirus (COVID-19) Tweets Dataset:** This dataset has tweets based on COVID-19, since the tweets were only about a single event, using this dataset we may not be able to build a model that can be generalised to all events.
- **Russian Troll Tweets:** This dataset consisted of tweets from social media troll accounts related to Russia's Internet Research Agency. Since it did not have labels as fake and real this dataset could not be used for classification of fake and real tweets. Moreover they did not have tweets from non troll accounts. Therefore even unsupervised learning can not be pursued with this dataset.
- **Protests Dataset:** This dataset had textual information about tweets related to HongKong protests. Since it was only based on a particular event and also due to lack of user information available, this dataset was not used.
- **FakeNewsNet:** This dataset contains news articles and also the related tweets labelled as fake or real by two popular fact checking websites namely: Politifact and Gossipcop.
  This dataset was ideal for our project since it contained information about the tweet, news articles and information about the user who participated in propagation of the tweet and also these news articles are from different events rather than a single event. Hence, the model can be generalised.

4.2 FakeNewsNet Crawler

FakeNewsNet is a publicly available Github repository which contains the urls to the tweets(alone) that are labelled by Politifact and Gossipcop. So we applied for a Twitter Developer Account to extract data related to the tweets and the user profiles of the users who tweeted them, using a crawler. Hence the data that can be crawled contains different sets of news articles labelled by both Politifact and Gossipcop. Till date, we extracted data from Politifact.

4.3 Structure of FakeNewsNet dataset:



CHAPTER 5

METHODOLOGY

5.1 Feature Engineering:

1. **Text analysis of tweets**
   Text analysis can be done for two kinds of tweets,
   - The news article tweets labelled as Fake and Real (Analysis on this tweet text will give us patterns specific to real and fake tweets).
   - The latest tweets posted by a particular user basically the user timeline tweets available in the dataset (By analysing these tweets, the bias of the user can be obtained. We can then attempt to find out if users who interacted with fake tweets have a certain bias that's different from users who interacted with real tweets).

   The analysis will be carried out in the following steps:
   Preprocessing

   1. Certain special characters like # @ etc needs to be removed using regular expressions.
   2. Lemmatization of tweet text in order to make all the tenses into present tense.

   Sentiment Analysis

   The sentiment evoked by the tweet text can be found using well known python libraries like nltk or textblob. A score can be obtained as to how positive or negative the tweet is and based on a certain threshold we can classify the text as Highly Negative, Negative, Neutral, Positive and Highly Positive.

   For example:
   Tweet text: "This country is very bad!! Going to the dogs and getting worse by the day!"
   Sentiment(polarity=-0.6549999999999999)
   Tweet text: "Today's a bright and sunny day, filled with lots of hope and joy and happiness!"
   Sentiment(polarity=0.65)

   Here the polarity denotes the sentiment of the tweet. Positive number denotes positive sentiment and Negative number denotes negative sentiment.

   The scatter plot below shows that there are more highly positive tweets in Real (label 0) than Fake (label 1) and more high negative tweets in Fake than Real.

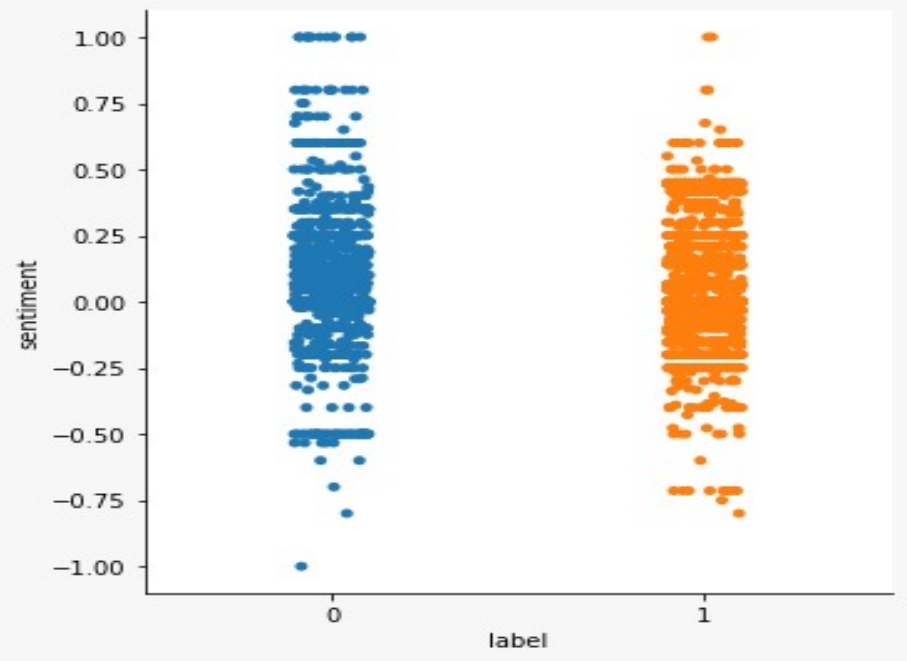


Figure 5.1: Sentiment for fake and real tweets

Every article contains a set of tweets sent by several users. The sentiment for each of the tweets is calculated and average of all the tweets is taken for each article. Fake articles tend to have a lower average sentiment than Real articles.

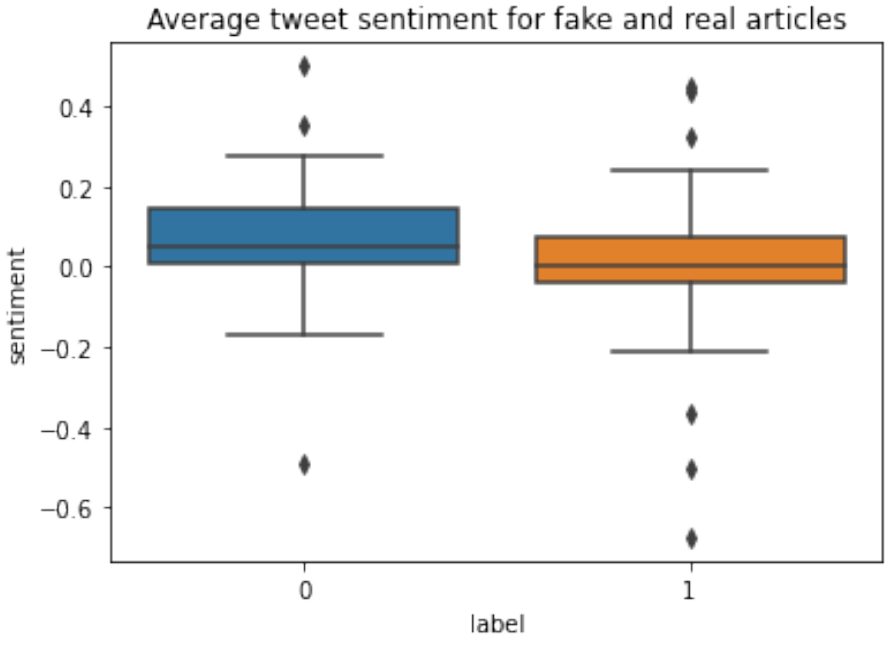


Figure 5.2: Average Sentiment for fake and real articles

Parts of Speech Tagging

Parts of Speech tagging will tag the nouns, verbs, adjectives etc present in a sentence.Count of nouns,Count of adjectives,count of verbs etc can be found from a

tweet text.

For example:
Tweet text: "Playing tennis is awesome"
('Playing', 'VBG'), ('tennis', 'NN'), ('is', 'VBZ'), ('awesome', 'JJ')

Here Tennis is the noun and awesome is the adjective

Named Entity Recognition Named entity recognition is a method that seeks to locate and classify named entity in text into predefined categories such as names of persons, organizations, locations, expressions of times, quantities, monetary values, percentages, etc. We can find the count of named entities occurring for fake and real tweets,this count can be used as a feature for classification of fake and real tweets

For example:
Tweet text: "President Marcus is bringing Nestle to shame!!"
Marcus **PERSON**
Nestle **ORG**

Here the entity 'Marcus' has been flagged as person name and 'Nestle' has been flagged as Organisation

On applying Named entity recognition on all the user timeline tweets. We see that Russia, USA, China are the most common countries.

Figure 5.3: Most commonly occurring Geographical Location names in user timeline tweets

2. **Analysing time variation of account and tweet creation**.

   The creation time of the accounts and the respective tweets are considered for this analysis. These are present in the tweets folder of the respective fake and real articles. Each json file in the tweets folder has certain keys of which "created_at"

indicates the tweet creation time of that particular tweet and the key 'user' which is another dictionary containing certain features of user who posted the tweet of which the key "created_at" indicates the account creation time of the user.

The time is present as "Weekday Month Date Time(Hour:Minute:Second) +0000 Year". From this, year, month, date and the time information is extracted using datetime.strptime module and then the difference is found between tweet and account creation time and the patterns are analysed for all users involved in spreading the fake and real news. From this analysis, it is found that the time gap between the account and tweet creation is bottom skewed for fake articles indicating that the interaction, once started, of fake users after creating their account is high in the beginning and later on decreases and top skewed for real users indicating that the real users interaction is less in the beginning but increases later on.

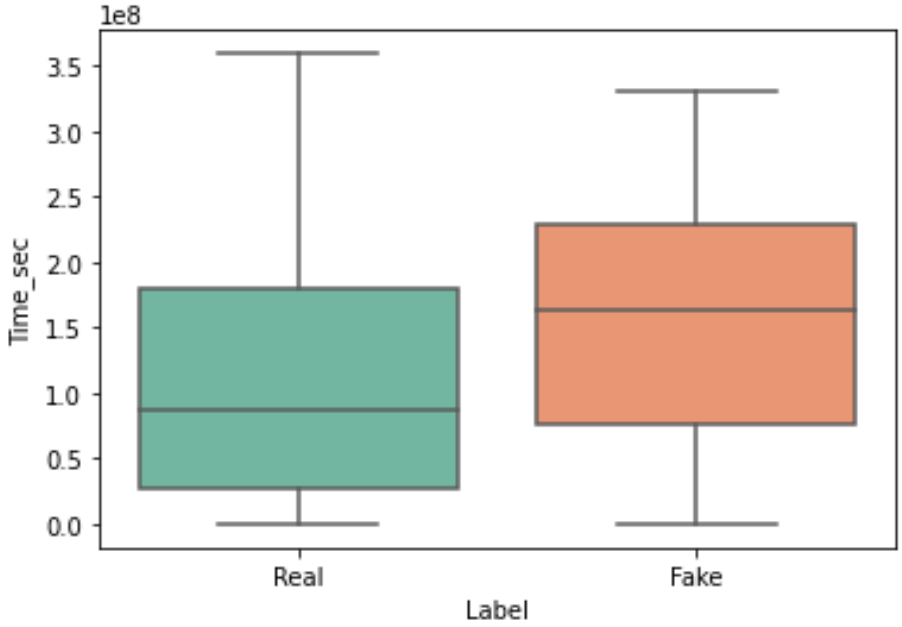


Figure 5.4: Account and Tweet Time variation: Real(Blue) and Fake(Brown)

Event	Minimum Time	Average Time	Maximum Time
Fake	55576	154772517.35772938	330559389
Real	10	109751654.02728285	358999522

Table 5.1: Minimum, Average and Maximum time variation of account and tweet creation for real and fake articles

Also, average time is more for fake users compared to the real ones. It might be because of the denser fake data compared to real ones or may be because of their time range of interaction. As such this can be one of the useful features for the detection of fake information.

3. **Analysing variation in published time of the article and creation time of the tweets and also the account creation time of the user related to the article**

The published time of the article and the creation time of respective tweets are considered for this analysis. Published time of the article is found in the news content.json file present in the respective folders of fake and real articles. Keys of news content.json file $->$ keys of 'meta_data'$->$keys of 'article'$->$ 'published' or 'published_time' indicates the published time of the article.

The article published time is present as "Year-Month-DateτHour:Minute:Second". From this, year, month, date and the time information is extracted using datetime.strptime module and then the difference is found between published time of the article and tweet creation time of the respective tweets and also between published time of the article and account creation time of the respective users. Then, the patterns are analysed for all tweets in the fake and real folders. From this analysis, it is found that there are only 2 real articles and 21 fake articles which have both article published time and the respective tweets' creation time and the respective account creation time.

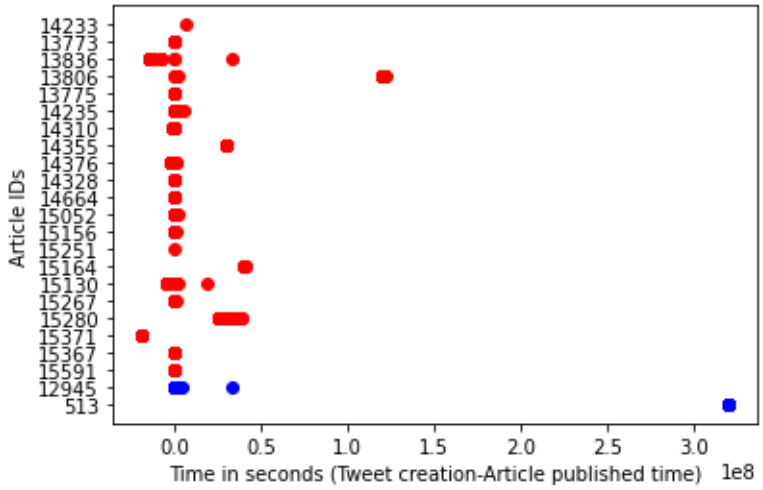


Figure 5.5: Time variation of tweet creation and article publication: Fake (Red) and Real (Blue)

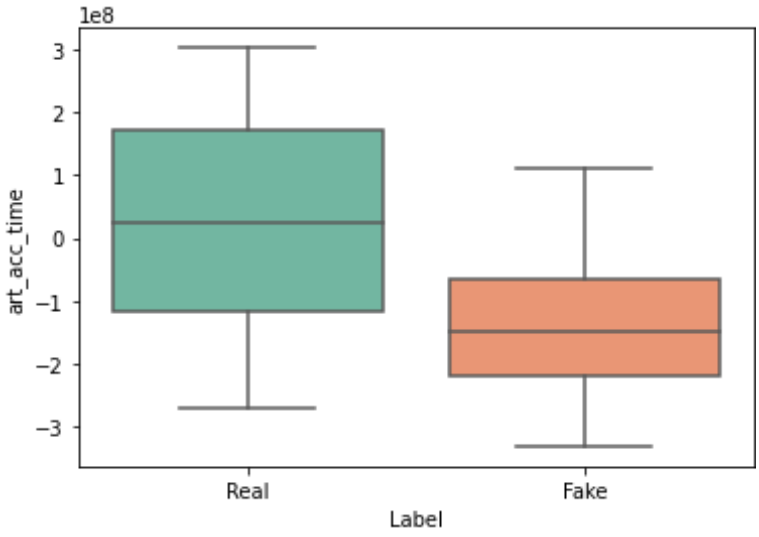


Figure 5.6: Time variation of account creation and article publication: Fake (Brown) and Real (Blue)

In the Figure 5.5, it is observed that most of the fake tweets are posted immediately and around the published time of the article. In the Figure 5.6, it is seen that the interquartile range is small for fake and large for real indicating that most of the fake accounts are created around the same time before the publishing of the articles.

4. **Analysing user tweet frequency in user timeline tweets**

   The time of the account and tweet creation of users in user timeline tweets is considered for this analysis. This analysis is basically to know the interaction patterns of individual users like how frequently he is tweeting, etc. User timeline tweets has json files related to twitter users and their respective timeline tweets. Each json file contains a list of dictionaries where each dictionary is related to one of the timeline tweets of the user. So here 'created_at' of each dictionary indicates the creation time of the account and the 'created_at' of the key 'user' indicates the account creation time which is same for all the tweets in a particular user file.

   This pattern related to the time gap between the account and tweets creation ofcourse varies for different users. But when analysing individual user behaviour, this tells us the timeline activity of user. Some of the users interacted more at the time of their account creation and later on it decreased, some interacted more in certain time gaps, some rarely interacted, etc form some of the user patterns.

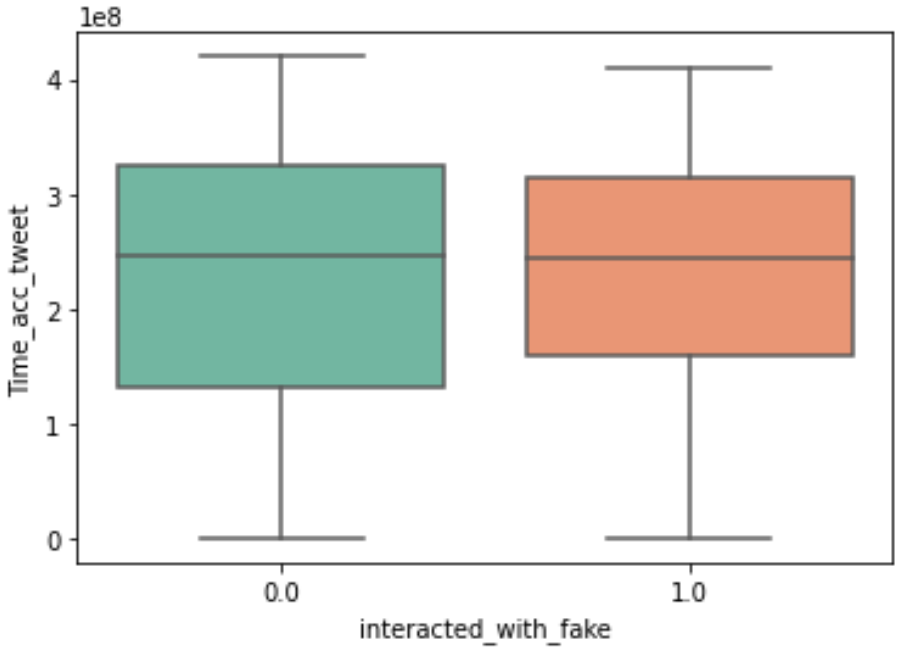


Figure 5.7: Time variation of tweet and account creation based on timeline tweets: Fake (Brown) and Real (Blue)

This time information when merged with the labelled tweet and account related information based on user ids, as shown in the figure 5.7, it is seen that interquartile range is more for real users compared to fake users and also the median value is more for real users indicating that the fake users interact rapidly at a certain point of time and take less time to respond on an overall.

5. **Analysis over user profiles**
   The dataset contains information about the tweets related to the labelled news articles, including the tweet id, user id who tweeted it, published time of the article and the time of creation of the tweet.
   User profiles contain information about #followers, #friends, #statuses uploaded(till the point of extracting the data), account creation time etc of each user.

   This data can be analysed as follows.
   Preprocessing:
   (a) Collect all the user ids of the users who participated in the propagation of fake news and real news, along with the tweet id and timestamps of their tweets while keeping track of the labels of the tweets.
   (b) Extract the #followers, #friends, #statuses and account creation time of the users from the user profiles data, using the user ids collected above.
   (c) Calculate the number of fake and real tweets that each user has tweeted in all along with the frequency of tweets related to each news event.

   Analysis of #followers, #friends and #statuses data:
   The following can be done to observe if there lies a pattern among the users who tweeted fake and real tweets.
   - Average #followers of the users who participated in the propagation of fake and real news respectively can be computed. This metric can be calculated for each news event individually as well as by taking all the news articles available in the dataset together.
   - Average #friends and average #statuses can be computed for each user in a similar fashion as that done for #followers.
   - The ratio of #followers to #friends can be computed for each user.

   The results of average #followers and average #friends for all available fake and real news articles taken together are shown below.

Event	avg #followers	avg #friends	avg #statuses
Fake	4918	3369	65165
Real	65512	3218	86395

Table 5.2: Average number of followers, friends and statuses of the users spreading fake and real news

   Taking two random fake and real events each from the dataset, the results are as follows.

Event	avg #followers	avg #friends	avg #statuses
Fake event-1	3413	2769	64454
Real event-1	10720	4304	51538
Fake event-2	2452	1895	43209
Real event-2	5434	2163	69394

Table 5.3: Average number of followers, friends and statuses of the users participated in spreading two random fake and real news each.

Below are the plots showing #followers for each user who were involved in fake and real news events 1 and 2.

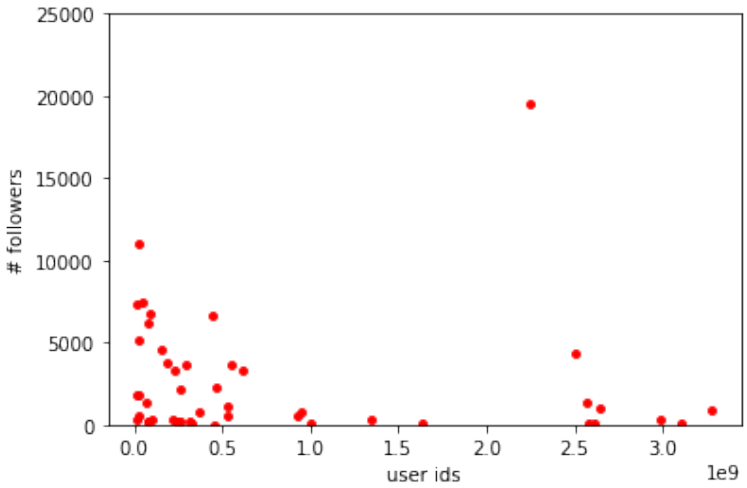


Figure 5.8: User ids vs #followers for fake news event-1

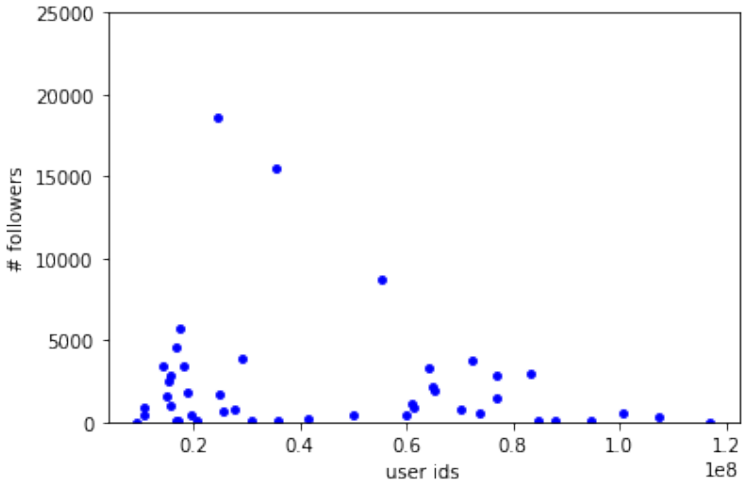


Figure 5.9: User ids vs #followers for real news event-1

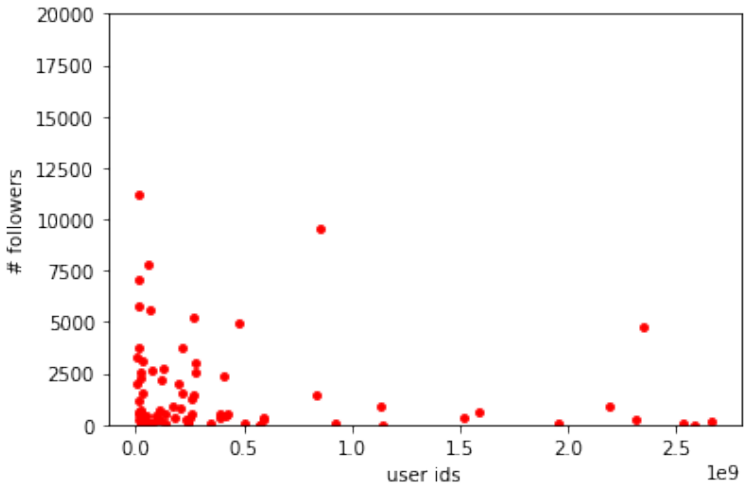


Figure 5.10: User ids vs #followers for fake news event-2

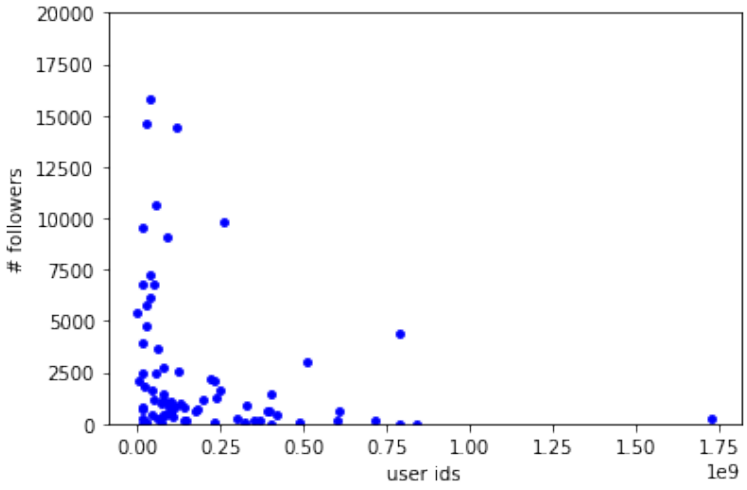


Figure 5.11: User ids vs #followers for real news event-2

Inferences from the above results taken together:

- The users who participate in the propagation of real news tend to have more #followers than that in fake news.
- There is no clear pattern in #friends and #statuses of users tweeting fake and real tweets.

So from these observations, the #followers can be considered as a key feature to classify tweets into fake or real.

6. **Analysis of trend in the frequency of tweets specific to each event**
   The motivation for this kind of analysis is that usually the number of tweets related to a fake news article tends to be high immediately after publishing the article. And later on, the number of tweets decreases with time.

   The time of tweet creation available for all the tweets in the dataset can be utilised for this analysis as follows.

Preprocessing:
For all the fake and real news articles available in the dataset, collect the article id, published time of the article along with the creation time of all the tweets available for each article.

Trend analysis:
Take the fake and real news articles from the dataset and do the following for one news event at a time.

(a) Calculate the number of tweets tweeted in each unit interval of time, starting from the time of publishing the news article. A unit interval of time could be chosen to be any fixed time interval- a minute, 30 minutes or even an hour.

(b) Plot this information as time intervals against the number of tweets(for each news article separately).

Below are the plots for a few fake news events.

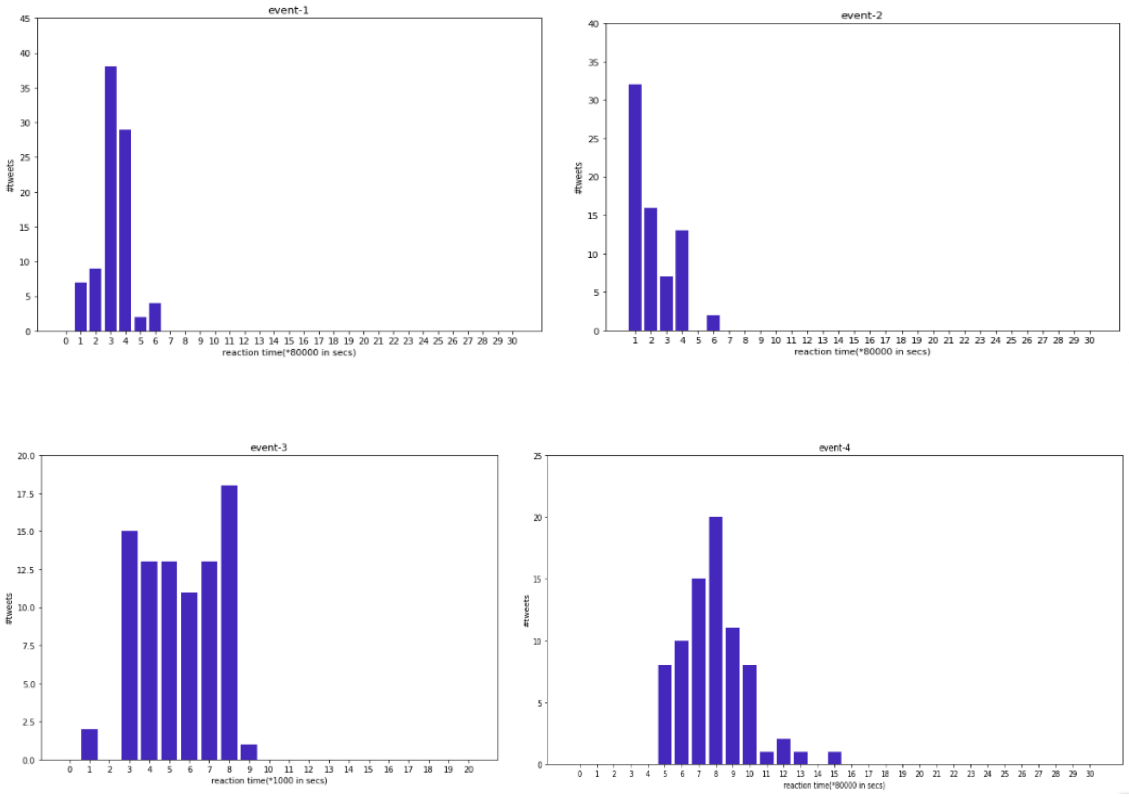


Figure 5.12: Frequency of tweets for a few fake news events

It is observed that the frequency of tweets is very high immediately after publishing a fake news article which complies with the motivation for this analysis.

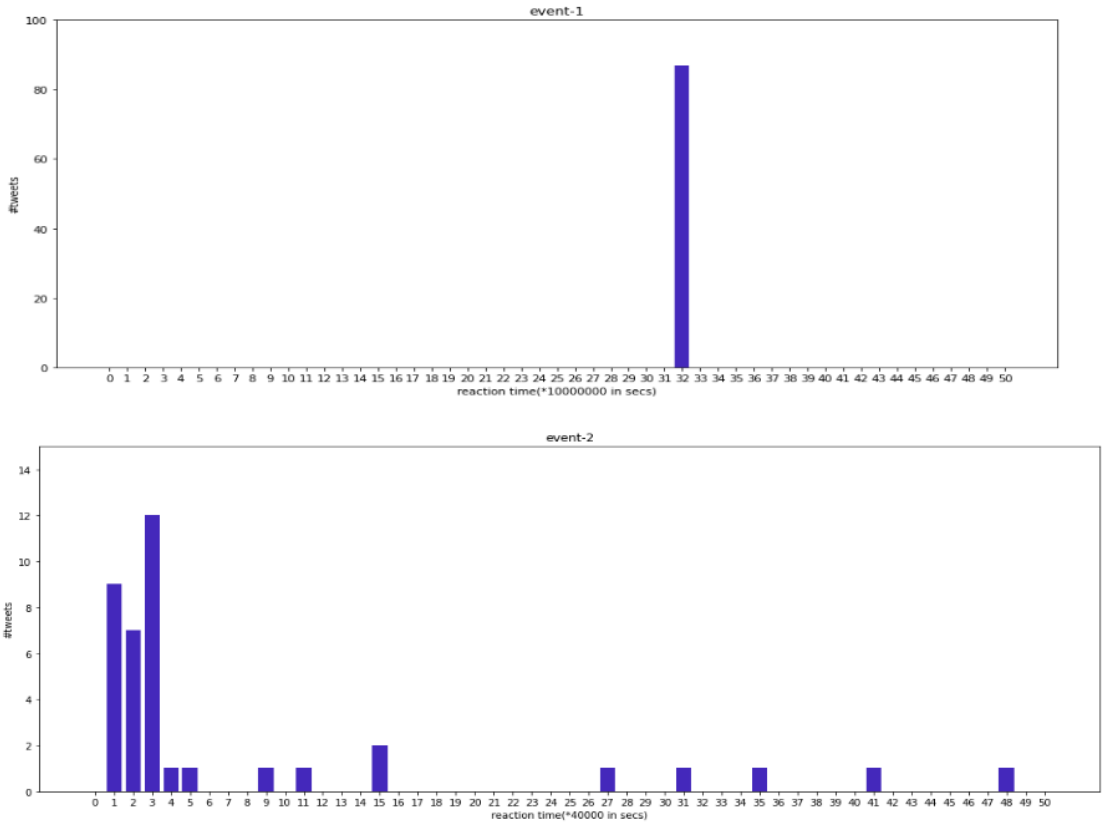


Figure 5.13: Frequency of tweets for a few real news events

However there's no clear trend that can be observed with the available data for real news events.
These trends in the data can be strongly proven with more data in hand by the end of completion of this model for fake news detection.

5.2 Feature Selection & Modelling

Textual features -

Sentiment of the tweet text, Count of verbs, Count of nouns, Adjective count, Named entities count- Count of person names, Count of organization names, Count of geopolitical location names are the textual features that are extracted. We picked these features and tried various combinations to see which features gave the highest accuracy. The classifiers used were Logistic Regression, Linear Regression, Random Forest Classifier, Support Vector Classifier and Gradient Boosting Classifier. Among these five models, Gradient Boosting gave the best results. Given below is a table of the different feature combinations and their train and test accuracies when classified by Gradient Boosting

Classifier.

Features	**Test Accuracy**
Person(NER) Count	0.648
Geopolitical Location(NER) Count	0.653
Nationalities or religious or political groups(NER) Count	0.653
Organization(NER) Count	0.672
Adjective Count	0.680
Verb Count	0.699
Noun Count	0.704
Sentiment	0.706
Noun Count, Sentiment	0.763
Noun Count, Sentiment, Verb count	0.793
Noun Count, Sentiment, Adjective count	0.799
Noun Count, Sentiment, Adjective count, Verb count	0.833
Noun Count, Sentiment, Adjective count, Verb count, Person Count	0.82
Noun Count, Sentiment, Adjective count, Verb count, GPE Count	0.830
Noun Count, Sentiment, Adjective count, Verb count, ORG Count	0.830
Noun Count, Sentiment, Adjective count, Verb count, NORP Count	0.831
Noun Count, Sentiment, Adjective count, Verb count, PERSON count, NORP count, ORG count, GPE count	0.868

Table 5.4: Best Classifier Results for various sets of Textual Features

Sentiment of the tweet text is the feature that is able to classify the dataset with the highest accuracy amongst all text features when each feature is considered individu-

ally. The accuracy obtained on using count of Named entities (PERSON, GPE, ORG, NORP) as the only feature is lower when compared to accuracies obtained while using Sentiment count, Adjective count, Verb count, Noun count etc. We see that combination of tweet sentiment with Noun count increases the accuracy. Again there is an increase observed when we add Adjective count to the set of features. This increase is higher compared to the accuracy obtained when we add Verb count to the feature set. When we take the feature set as Sentiment, Noun count, Verb count, Adjective count, the accuracy increases to 0.833. To this feature set when we add the Named entity counts individually, there is a decrease in the accuracy. On including all the named entity counts together with Sentiment, Adjective count, Verb count and Noun count, the accuracy increases to 0.868.

Temporal Features -

Time gap between account creation and tweet creation, time gap between article publication and tweet creation and time gap between article publication and account creation of the user are the available temporal features. In order to select various features, we took individual features and modelled them using certain classifiers and ensemble learning techniques as Random Forest Classifier, Support Vector Classifier, Gradient Boosting Classifier and Adaboost Classifier. The best results for various features with train & test split ratio 80:20 are as shown in table 5.5.

In the table 5.5, we can see that when we are using single feature itself, the accuracy scores are high for the features Article published time - Tweet creation time and Article published time - Account creation time. So when we analysed, we found that only 29.4% of our dataset has article published time out of which the ratio of fake and real data is around 11:1 . Due to this bias and the decrease in the size of dataset, the time gap between Article publication and Account creation and the time gap between Article publication and Tweet creation are not considered as features for our model. Now, we have only one temporal feature i.e, the time gap between account and tweet creation.

Features	**Classifier**	**Test Accuracy**
Account - Tweet creation time	Random Forest Classifier	0.688
Account - Tweet creation time	Gradient Boosting Classifier	0.686
Account - Tweet creation time	Adaboost Classifier	0.684
Article published time - Tweet creation time	Gradient Boosting Classifier	0.952
Article published time - Tweet creation time	Adaboost Classifier	0.956
Article published time - Account creation time	Random Forest Classifier	0.933
Article published time - Account creation time	Adaboost Classifier	0.933

Table 5.5: Best Classifier Results of Temporal Features taken one at a time

User profile Features -

Friends count, followers count, ratio of #followers to #friends and #statuses uploaded till date by each user are the available features related to user profiles.

Table 5.7 summarizes the results of classifiers which gave best results for the chosen user profile features set.

It is observed that the feature *#followers:#friends* gives the best result when one feature at a time is considered, while *Statuses count* performs the least when compared to the other individual features. It is also noticed that adding features into the features set increases the accuracy of classification models. While the combinations *Followers count, #followers:#friends* and *Friends count, #followers:#friends* perform equally well when two features are taken together whereas the set *Statuses count, #followers:#friends* shows a better performance. Hence the features *Followers count* and *Friends count* are together added to the features set already containing *Statuses count* and *#followers:#friends*, which gives the best accuracy score (*0.742* over test

dataset) among the various User Profile Features sets.

Features	**Classifier**	**Test Accuracy**
Followers count	Gradient Boosting Classifier	0.65
Friends count	Random Forest Classifier	0.65
Statuses count	Gradient Boosting Classifier	0.649
#followers:#friends	Gradient Boosting Classifier	0.726
#followers:#friends	Adaboost Classifier	0.726
Followers count, #followers:#friends	Gradient Boosting Classifier	0.728
Friends count, #followers:#friends	Gradient Boosting Classifier	0.728
Statuses count, #followers:#friends	Adaboost Classifier	0.737
Followers count, Friends count, Statuses count, #followers:#friends	Gradient Boosting Classifier	0.742

Table 5.6: Best classifier results for various sets of User Profile features

Also, the best performing classifier varies over different sets of features .

Different Feature Combinations -

On taking a combination of the best performing features of Text, Temporal and User profile namely sentiment, Time and friends followers ratio the accuracy is 0.80. Adding adjective count and noun count to the set increases the accuracy to 0.822. Adding #friends or #followers individually along with Verb count to this set, the accuracy increases to 0.846 and 0.847. When we add Statuses count to all the features mentioned above, the accuracy is decreased. So, this is the reason why the feature Statuses count is not used in our final features set. When we add all the Named Entity Recognition counts to the set of features obtained above excluding Statuses count, the accuracy increases to 0.872. This is the highest accuracy achieved out of all the combinations (Text, Temporal, User profile alone as well as combination of these 3 groups).

Features	**Test Accuracy**
Sentiment, #followers:#friends	0.75
Sentiment, Time, #followers:#friends	0.80
Adjective count, Noun count, Sentiment, Time, #followers:#friends	0.822
Sentiment, Time, Noun count, Adjective count, Verb count, #friends, #followers:#friends	0.846
Sentiment, Time, Noun count, Adjective count, Verb count, #followers, #followers:#friends	0.847
Sentiment, Time, Noun count, Adjective count, Verb count, #followers, #friends, #followers:#friends, Statuses count	0.839
#followers:#friends, Sentiment, Time, Noun count, Adjective count, Verb count, NORP count, ORG count, GPE count, PERSON count, #followers, #friends	0.872

Table 5.7: Best classifier results for combination of Text, Temporal and User profile features.

Classification Model

Analysing the performances of different classifiers with the chosen set of features shows that each classifier gives different accuracy as each one is based on a different classification strategy.

So in order to consider all the varying strategies used by the classifiers, its best to combine multiple classifiers to get a better result.

Ensemble learning is a process of combining multiple models, here classifiers, to solve a particular computational intelligence problem.

The following are the advantages offered by an ensemble learning model over any single learning model:

- Can make better predictions and perform better than any single contributing model.
- Reduces the spread of predictions made.

Bagging, Boosting and Stacking are the main types of ensemble methods. We have used **Stacking for Fake News Detection** where Stacking differs from the other two ensemble methods as follows.

- The models used in Stacking are typically different unlike Bagging where multiple instances of a similar model are used.
- In stacking, a single model learns the way to put together the predictions made by the contributing classifiers in the best possible way unlike in Boosting where a sequence of models are used which correct the predictions made by the previous models.

Stacking consists of several base models and a meta model. The meta model learns how to combine the predictions made by the base models(which perform better on the problem in different ways) to give the best possible result.

The Stacking model is trained as follows.
Given a dataset, it is internally divided into training and test sets(for the purpose of training the model alone). The widely used method to prepare the training set for the meta-model is k-fold cross-validation of the base models. The data which is not used for training the base models is fed to the base models to get their predictions. These predicted values and the expected output are together considered as the input-output pairs of the training set for the meta model. Hence the Stacking model is trained over the given dataset.

Selection of base & meta models for Stacking -
Different combinations of base and meta models are considered to form various Stacking models and their performances are analysed.

Base Models	**Meta Model**	**Train accuracy**	**Test accuracy**
SVC, RFC, GBC, ABC	ABC	99.76	90.87
SVC, RFC, GBC, ABC	RFC	99.59	91.41
KNN, LR, SVC, RFC, GBC, ABC	ABC	99.91	90.54
KNN, LR, SVC, RFC, GBC, ABC	RFC	99.89	91.09
KNN, LR, SVC, GBC, ABC	RFC	91.3	88.91
GNB, KNN, LR, SVC, RFC, GBC, ABC	RFC	99.97	91.41
GNB, KNN, LR, SVC, RFC, GBC, ABC, XBC	RFC	99.86	90.65
KNN, LR, SVC, RFC, GBC, ABC, XBC	ABC	99.89	91.41
KNN, LR, SVC, RFC, GBC, ABC, XBC	XBC	99.84	91.63
KNN, LR, SVC, RFC, GBC, ABC, XBC	**RFC**	**99.86**	**93.48**

Table 5.8: Accuracies(%) of various Stacking models

Based on the above models, the Stacking model with the following base and meta models is used for Fake Tweets Detection, which gives an accuracy of 93.48% on the test dataset.

Base models:

- K Neighbors Classifier
- Logistic Regression
- Support Vector Classifier
- Random Forest Classifier
- Gradient Boosting Classifer
- Adaboost Classifier

- XGboost Classier

Meta model:

- Random Forest Classifier

The performance of the above Stacking model is depicted through the below confusion matrix where the labels *0* and *1* represent *real* and *fake* respectively.

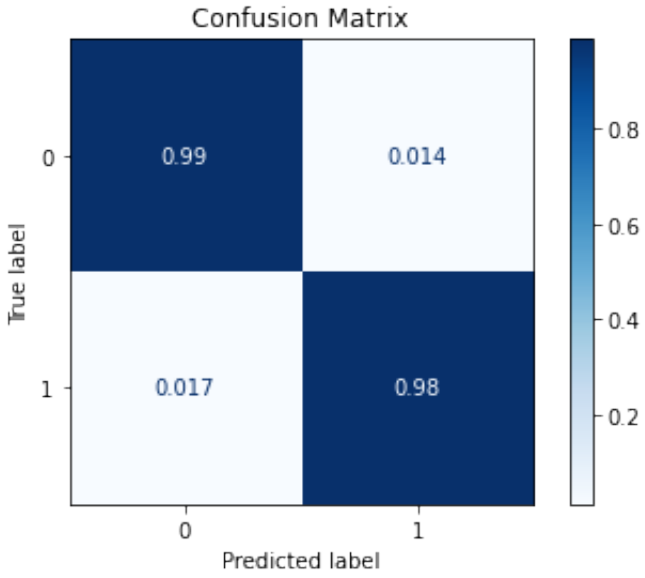


Figure 5.14: Confusion matrix for the chosen Stacking model

So, now that we have classified the fake and real tweets using the Stacking model, considering the predictions from the stacking model, articles are classified as fake or real. For this, we took the mode value of the classified labels for each article. For example, if a article contains 80 tweets and 60 tweets are classified as fake and remaining are classified as real, then the article is classified as fake as the number of fake tweets are more than the real tweets. We have 77 articles in total containing all the required features that we are using for the model. The overall accuracy after the classification of these articles using the method mentioned above is **0.987**.

5.3 Overview of the Proposed Approach

Below is the overview of our proposed approach to classify news articles into fake and real.

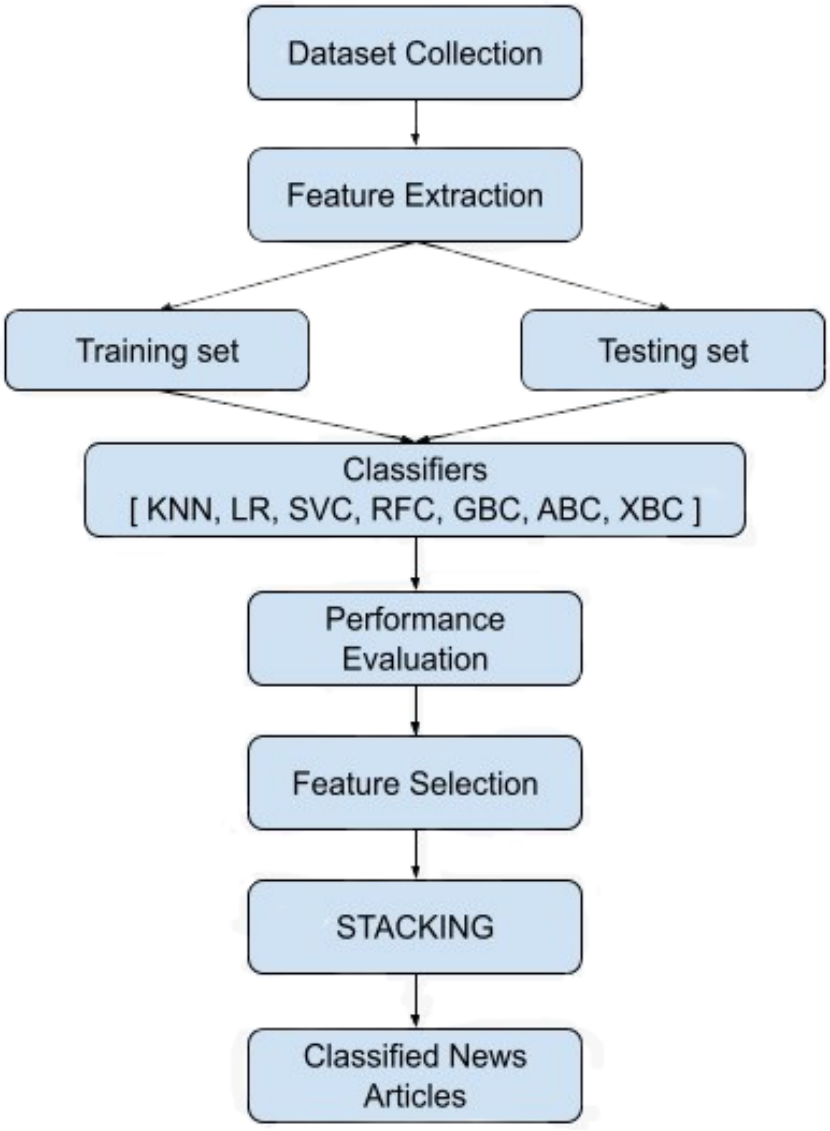


Figure 5.15: Proposed approach for Fake News Detection

CHAPTER 6

CONCLUSION AND FUTURE SCOPE

6.1 Conclusion

From the model proposed by us, we can see that it is possible to classify the news as fake or real based on the user engagements in online social media in the initial stages itself. So, the spread of fake news is decreased gradually. Feature Engineering plays a major role and also the Classification techniques used help in the detection of fake news more accurately.

6.2 Future Scope

1. Obtaining more tweets of different events to increase the dataset size and to obtain more accurate inferences.
2. Finding susceptibility score for each user. Provided we have a large dataset we can find out if a user is more susceptible to fake news or real news by looking at his/her past patterns of interacting with real or fake news. If the user has interacted with more fake news than real news in the past, there would be high probability that the current news article he/she is interacting with is fake. We were unable to take up this approach because in our dataset a particular user had interacted with maximum 3 or 4 articles alone, this number is too low to come up with a susceptibility score.
3. Finding out the patterns in fake and real news networks after obtaining the dataset with more data. Based on social psychological theories and the susceptibility score of each news spreader, certain patterns can be found and analysed at various network levels to extract more features which help understand the propagation of news in a network, and add on to a much better classification of fake and real news events.
4. Combining the textual, temporal and user profile features in different ways using various techniques and calculating the bias for each user. We can combine Named entity recognition with Sentiment analysis and find out the sentiment that occurs when a particular person's name or organization name etc is found in a tweet text.

REFERENCES

[1] V. L. Rubin, "On deception and deception detection: Content analysis of computer-mediated stated beliefs," *Proceedings of the American Society for Information Science and Technology*, vol. 47, no. 1, pp. 1–10, 2010. [Online]. Available: https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/meet.14504701124

[2] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, "Fake news detection on social media: A data mining perspective," *ACM SIGKDD Explorations Newsletter*, vol. 19, no. 1, pp. 22–36, 2017.

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.

[4] N. Ruchansky, S. Seo, and Y. Liu, "Csi: A hybrid deep model for fake news detection," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, ser. CIKM '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 797–806. [Online]. Available: https://doi.org/10.1145/3132847.3132877

[5] W. Y. Wang, ""liar, liar pants on fire": A new benchmark dataset for fake news detection," in *ACL*, 2017.

[6] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," 2014.

[7] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "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*. San Diego, California: Association for Computational Linguistics, Jun. 2016, pp. 1480–1489. [Online]. Available: https://www.aclweb.org/anthology/N16-1174

[8] Y. Long, Q. Lu, R. Xiang, M. Li, and C.-R. Huang, "Fake news detection through multi-perspective speaker profiles," in *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Taipei, Taiwan: Asian Federation of Natural Language Processing, Nov. 2017, pp. 252–256. [Online]. Available: https://www.aclweb.org/anthology/I17-2043

[9] T. T. Pham, "A study on deep learning for fake news detection," 2018.

[10] H. Karimi, P. Roy, S. Saba-Sadiya, and J. Tang, "Multi-source multi-class fake news detection," in *COLING*, 2018.

[11] T. N. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," in *Proceedings of the 5th International Conference*

on Learning Representations, ser. ICLR '17, 2017. [Online]. Available: https://openreview.net/forum?id=SJU4ayYgl

[12] S. Vosoughi, D. Roy, and S. Aral, "The spread of true and false news online," *Science*, vol. 359, no. 6380, pp. 1146–1151, 2018. [Online]. Available: https://science.sciencemag.org/content/359/6380/1146

[13] C. Silverman, 2016.

[14] M. Mcpherson, L. Smith-Lovin, and J. Cook, "Birds of a feather: Homophily in social networks," *Annual Review of Sociology*, vol. 27, pp. 415–, 01 2001.

[15] R. B. Cialdini, *Influence: Science and Practice*, vol. 4, 2009.

[16] A. Mukherjee, A. Kumar, B. Liu, J. WANG, M. Hsu, M. Castellanos, and R. Ghosh, "Spotting opinion spammers using behavioral footprints," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, vol. Part F128815. ACM New York, NY, USA, Aug. 2013, pp. 632–640, 19th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), Chicago 2013 ; Conference date: 11-08-2013 Through 14-08-2013. [Online]. Available: http://www.kdd.org/kdd2013/

[17] S. Xie, G. Wang, S. Lin, and P. S. Yu, "Review spam detection via temporal pattern discovery," in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '12. New York, NY, USA: Association for Computing Machinery, 2012, p. 823–831. [Online]. Available: https://doi.org/10.1145/2339530.2339662

[18] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, p. P10008, oct 2008. [Online]. Available: https://doi.org/10.1088%2F1742-5468%2F2008%2F10%2Fp10008

[19] K. Shu, D. Mahudeswaran, S. Wang, D. Lee, and H. Liu, "Fakenewsnet: A data repository with news content, social context and dynamic information for studying fake news on social media," *arXiv preprint arXiv:1809.01286*, 2018.

[20] K. Shu, S. Wang, and H. Liu, "Exploiting tri-relationship for fake news detection," *arXiv preprint arXiv:1712.07709*, 2017.

[21] Q. Huang, J. Yu, J. Wu, and B. Wang, "Heterogeneous graph attention networks for early detection of rumors on twitter," *arXiv preprint arXiv:2006.05866*, 2020.

[22] A. Zervopoulos, A. G. Alvanou, K. Bezas, A. Papamichail, M. Maragoudakis, and K. Kermanidis, "Hong kong protests: Using natural language processing for fake news detection on twitter," in *Artificial Intelligence Applications and Innovations*, I. Maglogiannis, L. Iliadis, and E. Pimenidis, Eds. Cham: Springer International Publishing, 2020, pp. 408–419.

[23] S. Helmstetter and H. Paulheim, "Weakly supervised learning for fake news detection on twitter," in *Proceedings of the 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, ser. ASONAM '18. IEEE Press, 2018, p. 274–277.

[24] R. Oshikawa, J. Qian, and W. Y. Wang, "A survey on natural language processing for fake news detection," in *Proceedings of the 12th Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, May 2020, pp. 6086–6093. [Online]. Available: https://www.aclweb.org/anthology/2020.lrec-1.747

[25] J. Ma, W. Gao, Z. Wei, Y. Lu, and K.-F. Wong, "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*, ser. CIKM '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 1751–1754. [Online]. Available: https://doi.org/10.1145/2806416.2806607

[26] G. Hu, Y. Ding, S. Qi, X. Wang, and Q. Liao, "Multi-depth graph convolutional networks for fake news detection," pp. 698–710, 09 2019.

[27] X. Zhou and R. Zafarani, "Network-based fake news detection: A pattern-driven approach," *ACM SIGKDD Explorations Newsletter*, vol. 21, pp. 48–60, 11 2019.