Identifying Possible Rumor Spreaders on Twitter: A Weak Supervised Learning Approach

Shakshi Sharma and Rajesh Sharma
Institute of Computer Science
University of Tartu, Estonia
Email: {shakshi.sharma, rajesh.sharma}@ut.ee

***Abstract*—Online Social Media (OSM) platforms such as Twitter, Facebook are extensively exploited by the users of these platforms for spreading the (mis)information to a large audience effortlessly at a rapid pace. It has been observed that the misinformation can cause panic, fear, and financial loss to society. Thus, it is important to detect and control the misinformation in such platforms before it spreads to the masses. In this work, we focus on rumors, which is one type of misinformation (other types are fake news, hoaxes, etc). One way to control the spread of the rumors is by identifying users who are possibly the rumor spreaders, that is, users who are often involved in spreading the rumors. Due to the lack of availability of rumor spreaders labeled dataset (which is an expensive task), we use publicly available PHEME dataset, which contains rumor and non-rumor tweets information, and then apply a weak supervised learning approach to transform the PHEME dataset into rumor spreaders dataset. We utilize three types of features, that is, user, text, and ego-network features, before applying various supervised learning approaches. In particular, to exploit the inherent network property in this dataset (user-user reply graph), we explore Graph Convolutional Network (GCN), a type of Graph Neural Network (GNN) technique. We compare GCN results with the other approaches: SVM, RF, and LSTM. Extensive experiments performed on the rumor spreaders dataset, where we achieve up to 0.864 value for F1-Score and 0.720 value for AUC-ROC, shows the effectiveness of our methodology for identifying possible rumor spreaders using the GCN technique.**

Keywords: Online Social Media, Misinformation, Graph Neural Network, Weak Supervised Learning.

I. Introduction

Online Social Media (OSM) platforms, initially developed for connecting individuals have become a hotbed for users who spread misinformation regularly by exploiting the connectivity and globality of these platforms [1]. On one side, these platforms offer a place for the expression of views. However, on the flip side, these platforms have not been able to regulate the spread of misinformation. According to [2], false information propagates six times faster than the true news on these platforms, which can result in panic, fear, and financial loss to society [3]. Thus, it is important to control misinformation propagation before it spreads to the masses. However, it is not a trivial task to identify such users as they blend very well by creating many connections with users who are not involved in spreading misinformation activities [4].

Misinformation can be categorized into two main types. The first one is the *fake news*, the news which is certainly not true. The second one is the *rumors*, a piece of information whose validity is in doubt at the time of posting. In other words, there is a doubt whether the information being posted is true or false. In this work, we focus on identifying the "possible" *rumor spreaders*. We define *rumor spreaders* as those users who are often engaged in spreading the rumors [3], and the term "possible" in our work points to the fact that it is very likely that the user could be a *rumor spreader*.

In the past, researchers have proposed various techniques for identifying suspicious or malicious users involved in the spread of misinformation on OSM platforms such as Twitter. These works include analyzing the user profiles' information [5, 6], observing user activity patterns across a specific time window [7, 8], tracking the profiles through the usage of a smartphone's battery [9]. Furthermore, few additional approaches utilized graph-based techniques [10] and multi-modal feature exploitation [11] for the detection of malicious profiles.

Identifying possible *rumor spreaders* is important, as they could be the potential source of misinformation propagation. Curbing on such users means controlling the misinformation diffusion as well. However, this problem has not acquired significant attention in contrast to detecting *rumors* or *fake news*. This is primarily due to the lack of an annotated dataset about *rumor spreaders* [12]. Thus, identifying possible *rumor spreaders* is challenging in many aspects, which is the theme of this research work.

In this paper, we use the **PHEME** dataset, which contains *rumor* and *non-rumor* tweets about five incidents, that is, i) *Charlie hebdo*, ii) *German wings crash*, iii) *Ottawa shooting*, iv) *Sydney siege*, and v) *Ferguson* occurred between 2014 and 2015. In order to transform the tweets dataset into a rumor spreaders dataset[1], we explore the sentiments of the tweets and calculate the *rumor spreaders' intensity score* (that is, how often a user posts *rumor* tweets). In addition, we also utilize the weak supervised learning approach, which is a branch of machine learning used to label the un-annotated data using few or noisy labels in order to avoid the expensive task of manual annotation of the data [13]. Please note that the labels that are generated using this approach are the *near ground-truth* labels. Thus, we use the term 'possible' for rumor spreaders dataset.

After the data transformation step, we leverage the following three distinct features for classifying possible *rumor spreaders*:

[1]Code and data is available at - https://github.com/shakshi12/RumourGNN

1) **User Features:** This includes users' features such as *number of followers, number of favorites*.
2) **Text Features:** We exploit users' tweets as a second set of features.
3) **Ego-Network Features:** We also explore the network of users who posted the tweets and the users who respond to those tweets. We illustrate this using Figure 1, wherein the central node I is a node who has posted a tweet, which we refer to as initiator user and the nodes **R1, R2, R3**, and **R4** are the responder users who have replied to the user **I**'s tweet. The weights on the edges correspond to the number of times user **I**, and responder user interacted with each other. Based on our dataset, we are only able to create a one-hop network (ego-network) comprising of initiator user and its responders.

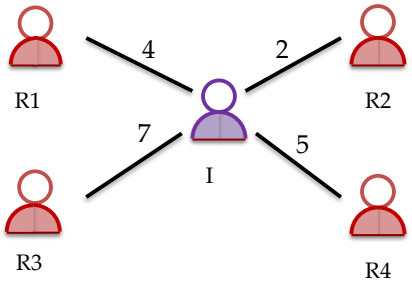


Fig. 1: Undirected Weighted Ego-Network of a Twitter User

It is important to note that the network formed between the initiator of the tweets and their corresponding responders naturally calls for a technique that can exploit network features as well. Thus, in order to capture the network properties, we employ Graph Neural Network (GNN) based approach for identifying possible *rumor spreaders*. To be specific, we explore Graph Convolutional Network (GCN), a particular type of GNN for our analysis. In contrast to the baseline approaches: SVM, RF, and LSTM, the GCN approach exhibit better performance and is able to achieve a value of 0.864 for F1-Score and 0.720 for AUC-ROC Score. To the best of our knowledge, this is the first work that has explored the **PHEME** dataset to identify possible *rumor spreaders* using a weak supervised approach.

The rest of the paper is organized as follows. Section II covers related work. Section III covers dataset description and methodology used for identifying possible *rumor spreaders*. Section IV discusses experiments and their results, and Section V concludes with some future directions.

II. Related work

Researchers have proposed numerous techniques for identifying suspicious or fake user profiles across various OSM platforms such as Facebook [7], Twitter [5, 14], Tuenti [10], to name a few. In this section, we discuss past studies across two dimensions. First, we discuss work related to the identification of fake or suspicious users. Next, we present works about the detection of rumors. Our work lies at the intersection of these two types of works.

A. Suspicious Profile Detection

Identification of suspicious (or fake) profiles on Twitter, a representative of OSM was initially analyzed using apparent features such as the *number of followers* and the *number of followings* [5], *user-profile-name, screen-name, and email parameters* [6] Our work is different from these global approaches as they assumed the whole bird-eye view of the network. In contrast, we are only aware of the ego-networks. Some other works have focused on identifying suspicious followers by using multi-modal information [11], which is out of the scope of this work.

It is a well-observed fact that mobile phones have been playing a vital role in the popularity of OSM platforms. Nonetheless, it has facilitated the rise of fake profiles as well [9]. Therefore, mobile phones too have been used as a medium in tracking fake profiles. For instance, in [15], the authors observed the daily behavior of users using mobile phone activity for detecting fake profiles. Besides researchers have also used camera-based sensors in detecting fake profiles within or across multi social networks [16].

B. Rumor Detection

We assume that users who are often involved in disseminating rumors are more likely to be *rumor spreaders* in comparison to users whose involvement in spreading rumor is less. Our approach for identifying the possible *rumor spreaders* exploits (rumor and non-rumor) tweets along with other features present in the dataset. Therefore, we further present related literature with respect to rumor detection.

Initially starting with the theoretical framework for rumor spreading [17], and later identification of rumors' temporal, structural, and linguistic features [18], the topic related to rumor detection has attracted considerable attention in recent years, especially because of advancement in the field of artificial intelligence techniques. For example, in [19], various flavors of LSTM architecture are explored and in [20], a multi-task deep learning model is presented for rumor detection. In addition, several techniques such as the use of particle swarm optimization [21], multi-modal approach by exploiting textual as well as visual features from the data [22, 23], have also been examined in the literature.

Few approaches in rumor detection have also explored graph-based methodologies. Specifically, in [24], the authors constructed a Twitter follower graph and exploited diffusion patterns of (mis) information for rumor detection. Some of the papers [25]–[27] have employed Graph Convolutional Network (GCN) based approaches for detecting rumors. Unlike these works, we study the rumors to identify possible *rumor spreaders*.

User profiles have been analysed as an important aspect in detecting rumor propagation. Analysis of the user profile [28], identification of source of the rumor [29], genuineness score of the users in the social network which are spreading

the rumors [30] have been utilized in the past works. Our work lies at the boundary of these works as our aim is to identify possible *rumor spreaders* by using not only the textual data that is being spread by these *rumor spreaders* but also exploring their ego-networks. To accomplish this objective, we used GCN approach which has been mainly used in the past for identifying rumors and not for identifying possible *rumor spreaders*.

III. Dataset Description and Methodology

In this section, we first discuss the **PHEME** dataset (Section III-A). Next, in Section III-B, we describe how we transform the **PHEME** tweets dataset into the rumor spreaders dataset. Finally, we explain three different types of features extracted from the dataset, which are provided as input to the machine learning algorithms (Section III-C).

A. Original Dataset

This paper utilizes the **PHEME**[1] dataset, which is a collection of *rumor* and *non-rumor* tweets that have been extensively used in previous works [3, 31]. The dataset comprises five events (or incidents) - *Charlie hebdo*, *German wings crash*, *Ottawa shooting*, *Sydney siege* and *Ferguson*. For the rest of this work, we refer to them as *Charlie*, *German*, *Ottawa*, *Sydney*, and *Ferguson*, respectively. The dataset contains information about the tweets pertaining to these incidents that have been posted as breaking news during the year 2014 – 2015. To be specific, data is provided in the form of five files, where each file is related to five particular incidents. The data in each file is stored in JSON format, having information about the source (or initiator's) tweet and its corresponding information. Table I provides detailed information about various fields.

TABLE I: Dataset Description

No	Fields	Description
1	user id	unique id of the initiator user
2	tweet	tweet posted by initiator
3	# of followers	of the initiator
4	# of favorites	of the initiator
5	verified user	source user has verified account or not
6	reply user id	unique id of the reply user
7	reply tweet	tweet posted by reply user
8	# of reply followers	of the reply user
9	# of reply favorites	of the reply user
10	verified reply user	reply user has verified account or not
11	label	initiator's tweet is rumor or not

Furthermore, each source tweet has the ground-truth regarding whether the tweet is a *rumor* or *non-rumor*. Table II, column '# of Tweets (%)' provides information about the number of *rumor* and *non-rumor* tweets for each of the incidents for our analysis.

B. Transformation of Tweets Dataset into Rumor Spreaders Dataset

Due to the lack of an annotated dataset of users who are spreading the *rumors* on OSM platforms, we first transform the **PHEME** dataset (which carries information pertaining to the initiator's tweets, such as users who replied to the initiator's tweets, the *followers count* of the initiator user, etc.) into the rumor spreaders dataset. Table III, column 'Tweets', shows the original dimensions of the incidents, wherein each cell value represents the total number of initiator's tweets (36189 rows in case of *Charlie*) and the total number of features, including ground-truth labels (11 columns for all the incidents).

[1]https://figshare.com/articles/PHEME_dataset_of_rumours_and_non-rumours/4010619

TABLE II: Distribution of Tweets and Spreaders for each of the incidents in the dataset

	# of Tweets (%)		# of Spreaders (%)	
Incidents	**Rumor**	**Non-rumor**	**Rumor**	**Non-rumor**
Charlie	458 (22%)	1621 (78%)	13879 (74.2%)	4821 (25.8%)
German	238 (50.7%)	231 (49.3%)	1464 (50.3%)	1442 (49.7%)
Ottawa	470 (52.8%)	420 (47.2%)	3978 (51.1%)	3794 (48.9%)
Sydney	522 (42.8%)	699 (57.2%)	7545 (61.8%)	4658 (38.2%)
Ferguson	284 (24.8%)	859 (75.2%)	3792 (35%)	7001 (65%)

TABLE III: Dimensions of the **PHEME** dataset at various levels

Incidents	Tweets	Rumor Spreaders	Adjacency Matrix
Charlie	36189, 11	18700, 304	18700, 18700
German	4020, 11	2906, 304	2906, 2906
Ottawa	11394, 11	7772, 304	7772, 7772
Sydney	22775, 11	12203, 304	12203, 12203
Ferguson	46064, 11	10793, 304	10793, 10793

In order to identify possible *rumor spreaders*, we start by placing each user with its corresponding tweets followed by its responders and their respective reply tweets. As part of the data cleaning process, we remove non-alphanumeric characters, URLs, stopwords, punctuations, lowercase all the words and perform additional Natural Language Processing operations as well, for instance, Porter Stemming of the words. We cover the steps taken for the conversion:

Step 1: Sentiment Analysis of the Reply tweets: It is highly likely that a tweet may have attracted multiple responses (or replies). To exemplify, Table IV shows an example of the two tweets from *Ferguson* incident. The first row corresponds to the *non-rumor* tweet along with its replies (in this Table, we have shown only two replies, but a tweet can have any number of replies), whereas the second row corresponds to the *rumor* tweet. It can be seen that the reply tweets possess sentiments with respect to the posted tweet, which can help in identifying the *rumors*, and thus, *rumor spreaders*. For instance, *non-rumor* tweets are in support of the initiator's tweet, hence, represents a positive sentiment. Whereas in the case of *rumor* tweets, reply tweets do not show support of the initiator's tweet, indicating negative sentiment. This could be a key indication that sentiments of the reply tweets play a key role in identifying whether the tweets posted by the user is *rumor* or not.

In this regard, we first try to analyze the sentiments of

TABLE IV: An Example of the Tweets on Ferguson incident

No	Initiator Tweet	Reply Tweet 1	Reply Tweet 2
1	The mother of the boy killed in #Ferguson speaking to media about the loss of her son. http://t.co/YlxEDKoebB	@AntonioFrench @b9AcE guess the cops were protecting and serving the community again.	@AntonioFrench my heart aches for her! This was so wrong!
2	Police in #Ferguson once charged a man w/ destruction of property for bleeding on their uniforms after they beat him http://t.co/MRVP76sdUP	@AnonyOps that's not true Dudeee!! please go and read good newspapers.	@RianAlden not at all, but they need to change some things at #ferguson PD. @AnonyOps

the reply tweets using *TextBlob API*[2]. Figure 2 displays the sentiments of the reply tweets with respect to *rumor* and *non-rumor* tweets for all five incidents. Specifically, the x-axis represents the positive and negative sentiments with respect to *rumor* and *non-rumor* reply tweets for all the incidents and y-axis corresponds to its percentage. To capture the same, we consider the reply tweets under the *rumor* category if the initiator's tweet is a *rumor* otherwise *non-rumor*. It is clear from Figure 2 that reply tweets under the *rumor* category have mostly negative sentiments and vice-versa for all the incidents (we have excluded the # of neutral sentiments, which are very few in number to avoid confusion). Thus, it can be validated that the sentiments of the reply tweets can be utilized to identify *rumor* tweets, and hence, possible *rumor spreaders*.

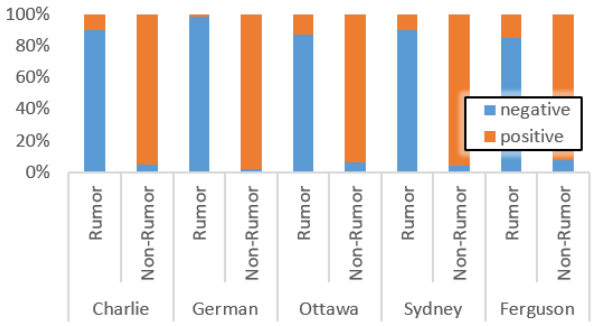


Fig. 2: Sentiments of Reply Tweets with respect to Rumor and Non-Rumor Tweets for all the five incidents

Step 2: Labeling Reply tweets Using Weak Supervised Learning Approach: In order to identify the possible rumor spreaders, we would like to utilize the stance of reply tweets in our approach. However, no such information is present in the dataset. Thus, in order to label each reply tweet, we apply the *MinHash*[3] algorithm in line with [32]. The *MinHash* finds the similarities between each pair of the initiator's tweet and the reply tweet. Specifically, if both these tweets are similar, then we assign the same label to the reply tweet as the initiator's tweet, indicating that the reply tweet is in support of the initiator's tweet. Otherwise, we assign the opposite label to the reply tweet. We considered two tweets to be similar if their similarity score is greater than or equal to 85% (this threshold is validated manually). This approach of labeling the reply tweets is what we call as weak supervised learning approach due to the fact that we do not have the manual annotation of these tweets.

Step 3: Calculating *rumor spreaders' intensity score:* Tweets' labels only indicate whether a particular tweet is a *rumor* or *non-rumor*. Therefore, in order to identify possible *rumor spreaders*, we calculate a score that indicates the intensity with which users spread *rumors*, which we term as *rumor spreaders' intensity score*. We compute this score for each user by using the following formula:

$$score = \frac{\#\ of\ times\ user\ tweets\ rumor}{Total\ \#\ of\ times\ user\ tweets} \quad (1)$$

where the denominator is calculated by counting the total number of tweets posted by a user, whereas the numerator is calculated by counting the total number of *rumor* tweets posted by a user. The score range lies between [0, 1], where 0 means not a *rumor spreader*, and 1, indicating possibly a *rumor spreader*.

In order to validate the effectiveness of this score, we calculate the degree (number of connections) from the user-user reply graph, as shown in Figure 1. We observe that the nodes (or users) who are connected to many other nodes (or users), that is, high degree, are more involved in posting *rumor* tweets as compared to nodes (or users) who have a low degree. We then manually check the users with their *rumor spreaders' intensity score* calculated using Equation 1. The score is in line with the degree, which verifies our approach of identifying possible *rumor spreaders*.

To model the problem as a binary classification problem, we put a threshold of 0.5 to create two classes of users. That is, ***if the rumor spreaders' intensity score is < 0.5, then we assign 0 label (indicating non-rumor spreader class); otherwise, we assign 1 representing the possible rumor spreader class.*** The reason for choosing this threshold is based on the observation that the sentiments of the tweets as discussed in Step 1 are positive when the *rumor spreaders' intensity score* is less than 0.5, which is indicative of *non-rumor* spreaders class and vice-versa. Based on this threshold conversion, Table II, column '# of Spreaders (%)' shows the number of possible *rumor spreaders* and *non-rumor spreaders* for each of the five incidents in the dataset.

C. Extraction of three sets of features

In this section, we describe three distinct sets of features that we extract, to be utilized by our machine learning models for predicting possible *rumor spreaders*.

[2]https://textblob.readthedocs.io/en/dev/api_reference.html
[3]snaPy API: https://pypi.org/project/snapy/

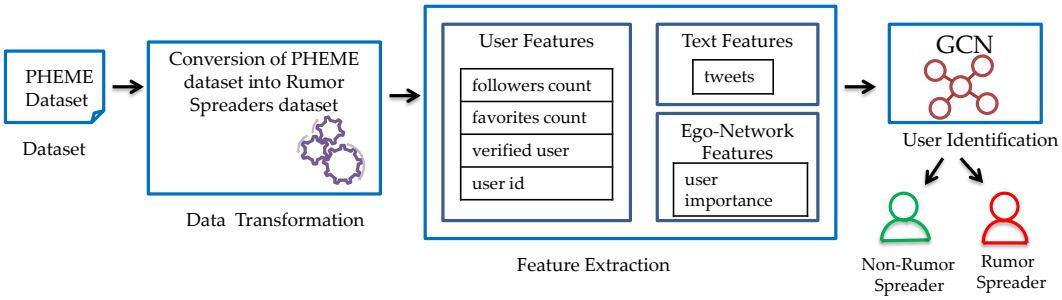


Fig. 3: Framework

1) **User Features:** This set of features represents user's profile-based information that includes *followers count*, *favorites count*, and *verified users*.
2) **Text Features:** This feature represents text-related features, such as the *tweet* column in our dataset. We employ popular *Word2Vec* embedding (a specific type of Vector Space Models) [33] to convert the tweets into a numeric vector. Specifically, for each unique word in the *tweet*, *Word2Vec* generates its corresponding 300-dimension numeric vector, which is then aggregated in such a way that each sentence represents a 300-dimension vector. Besides, we remove few noisy words as well, for instance, aaand, aand, aaaaand using English vocabulary.
3) **Ego-Network Features**: To capture the network properties in the data, we created a feature which we call as *User Importance* (user_imp) feature. This feature helps in adding the network properties in the dataset by calculating the importance of each user with respect to replies it has received. The formula for calculating this feature is as follows:

$$user\_imp = \frac{\#\ of\ replies\ a\ user\ gets}{\#\ of\ replies\ of\ all\ users} \quad (2)$$

Figure 3 summarizes the framework, which we discussed in this Section. Section IV discusses the parameters used by GCN and the results of various approaches.

IV. Experiments and Result Section

In this section, we discuss few of the main hyper-parameter tuning used by our models (Section IV-A). Next, we discuss various optimization techniques (Section IV-B), and lastly, we discuss the results of our approaches (Section IV-C).

As mentioned in Section I, there is a network formation between initiators and responders in terms of tweets. Thus, we use GCN [34], a type of GNN technique, which can exploit both the network structure and features (such as *tweets, followers count*). In particular, we perform a binary classification for identifying if a specific Twitter user is possibly involved in *rumor* spreading or not. In addition, we compare GCN with other baseline approaches (SVM, RF, and LSTM).

A. Hyper-Parameters tuning

Table V shows the values for various hyper-parameter settings that are chosen for fine-tuning the models for improving the performance. Here, we have specified few of the main hyper-parameters only. For instance, SVM has a hyper-parameter, *kernel*, which is a function that finds the similarity score between two data points even from the high dimensional input space in order to find the optimal hyperplane. RF has a hyper-parameter known as, *number of trees in forest* that specifies how many trees should be formed so that they can be used as a parallel estimators in order to make final prediction. The rest of the hyper-parameters are specific to the neural networks (for both LSTM and GCN). The *number of layers in a network* represents the total number of layers used in a model, *number of channels in each layer* represents the total number of output neurons to be used in a layer, *drop out layer* is used to avoid overfitting of the data. The *activation function* decides which neuron to activate for the next layer in the network, *number of epochs* refers to the number of iterations the neural network model is to be trained for. The *loss function* is used for refining the model after every epoch. The *NA* values in the table refer to Not Applicable.

TABLE V: Hyper-Parameters Tuning of models

Parameters\ Models	**SVM**	**RF**	**LSTM**	**GCN**
Kernel	radial basis function	NA	NA	NA
# of trees	NA	100	2	2
# of layers	NA	NA	2	2
# of channels in first layer	NA	NA	32	32
# of channels in second layer	NA	NA	2	2
Drop out layer	NA	NA	0	2
# of Epochs	NA	NA	300	300
Activation function	NA	NA	sigmoid	sigmoid
Loss function	average	hinge	binary cross entropy	binary cross entropy

Apart from the Table V hyper-parameters, GCN model takes two additional inputs - graph adjacency matrix and nodes' features matrix. The graph adjacency matrix stores the nodes' neighbors information in a ZXZ matrix, where Z represents the total number of nodes (users) in the dataset.

The nodes' features matrix ZXF is the final matrix of the preprocessing step, where Z represents the number of nodes and F is the size of total features. As already mentioned, Table III, column 'Rumor Spreaders', shows the dimensions of the nodes' features matrix for each of the incident. Specifically, each cell values represents the nodes, Z and the features, F where F represents the features (attributes) such as *followers count*, word embedding of tweet, *user_imp*. In addition, the dimensions of the adjacency matrix are shown in III, column 'Adjacency Matrix', wherein each cell values represents the total number of users in a dataset. After providing required inputs to GCN, the model is trained to predict possible *rumor* and *non-rumor spreaders*.

In order to predict possible *rumor spreaders*, the nodes have labels as 0 (*non-rumor spreader*) or 1 (possibly a *rumor spreader*). In Section IV-C, we discuss results of all the machine learning approaches.

B. Optimization Techniques

As part of the optimization, we perform following steps -

1) **Cross Validation:** To ensure the effectiveness of our model and to avoid overfitting of data, we perform K-Fold cross-validation on our dataset where $K = 5$. To avoid the class imbalance problem in *Charlie* incident, we use Stratified K-Fold.
2) **Standardization:** All the features are standardized before training the machine learning model.
3) **Feature Importance:** In addition to the above two techniques, three feature selection techniques, namely, *Chi-Square, Information Gain, Gain Ratio* are applied to each of the five incidents of rumor spreaders dataset to check whether each feature is correlated with the target variable. Table VI depicts the p-values of three feature selection techniques. For all the features, p-value < 0.05 which shows that these features are important in predicting possible *rumor spreaders*. Thus, we consider all the features in our experiments.

TABLE VI: Feature Selection Techniques (values in the cells indicates their corresponding p-values)

Features	Chi-Square	Information Gain	Gain Ratio
followers count	5.28e-05	4.62e-07	8.56e-09
favorites count	8.66e-17	11.65e-22	5.56e-11
verified users	2.26e-10	1.44e-05	3.67e-07
user importance	7.87e-20	9.93e-23	7.83e-26

C. Results

In this section, we discuss the micro and macro-analysis of our evaluation for all the incidents using five metrics.

1. Macro-Analysis: Table VII provides the macro-averaged results of our machine learning models for all the five metrics for *Charlie*, *German*, *Ottawa*, *Sydney*, and *Ferguson* respectively. In general, GCN outperforms other classifiers in all the five metrics. SVM and RF perform better than LSTM in most of the metrics. Considering per incident evaluations, the results of *German* incident outperforms with a significant margin,

TABLE VII: Metrics performance of different models

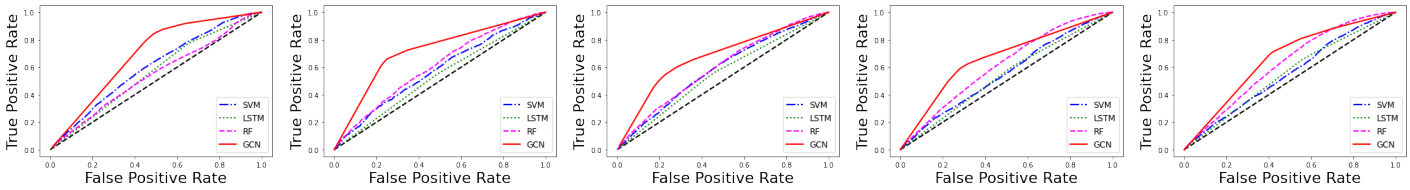
S.No.	Metrics	SVM	RF	LSTM	GCN
	Charlie				
1	**Accuracy**	0.748	0.760	0.671	**0.790**
2	**Precision**	0.748	0.758	0.571	**0.790**
3	**Recall**	0.748	0.628	0.571	**0.790**
4	**F1-Score**	0.853	0.840	0.778	**0.864**
5	**AUC-ROC**	0.600	0.600	0.570	**0.690**
	German				
6	**Accuracy**	0.552	0.567	0.541	**0.715**
7	**Precision**	0.553	0.567	0.541	**0.717**
8	**Recall**	0.552	0.567	0.541	**0.716**
9	**F1-Score**	0.572	0.567	0.546	**0.709**
10	**AUC-ROC**	0.552	0.566	0.540	**0.720**
	Ottawa				
11	**Accuracy**	0.567	0.565	0.552	**0.675**
12	**Precision**	0.567	0.565	0.552	**0.681**
13	**Recall**	0.566	0.565	0.552	**0.677**
14	**F1-Score**	0.578	0.569	0.559	**0.655**
15	**AUC-ROC**	0.566	0.566	0.550	**0.680**
	Sydney				
16	**Accuracy**	0.618	0.639	0.561	**0.655**
17	**Precision**	0.565	0.606	0.541	**0.655**
18	**Recall**	0.520	0.579	0.542	**0.664**
19	**F1-Score**	**0.751**	0.740	0.638	0.690
20	**AUC-ROC**	0.618	0.638	0.540	**0.660**
	Ferguson				
21	**Accuracy**	0.652	0.675	0.585	**0.705**
22	**Precision**	0.598	0.634	0.549	**0.671**
23	**Recall**	0.519	0.589	0.549	**0.658**
24	**F1-Score**	0.782	0.778	0.676	**0.783**
25	**AUC-ROC**	0.652	**0.674**	0.550	0.660

whereas *Charlie* results exceed with a small margin. In spite of using Stratified K-Fold, the class imbalance problem might have affected the *Charlie* results.

2. Micro-Analysis: Figure 4 shows the micro-performance of each of the approaches under five metrics for all the incidents in our dataset. Specifically, the x-axis represents the fold number under five-fold cross-validation, whereas the y-axis represents the metrics used for evaluation. To summarize, each incident consisting of five plots depicting Accuracy, Precision, Recall, F1-Score, and AUC-ROC Score for each fold. It can be seen from the plots that, in general, the GCN approach is performing better than the baseline models. However, GCN approach on *German* and *Ottawa* incidents performed significantly better compared to the other two approaches. This clearly implies that the GCN approach is a natural fit for our problem statement.

Furthermore, we plot the AUC-ROC plot for each of the incidents to understand the micro-performance of the algorithm at different thresholds. Figure 5 shows the AUC-ROC Curve for *Charlie*, *German*, *Ottawa*, *Sydney*, and *Ferguson* respectively. It can be noticed that SVM, RF, and LSTM perform little better than the random model, whereas GCN is better with a good margin. However, RF performs better than the other two baselines in the case of the *Ferguson* incident. Besides, the GCN approach performs well on lower thresholds for *German*, *Ottawa*, and *Sydney*. In contrast, the reverse is the case for *Charlie*, and *Ferguson* indicating the reasons, higher values of Accuracy, Precision, Recall, and F1-Score

Fig. 4: Performance of GCN model under five-fold cross validation for all the incidents



(a) Charlie (b) German (c) Ottawa (d) Sydney (e) Ferguson

Fig. 5: Performance of AUC-ROC Curve on all the five incidents

than AUC-ROC values for the *Charlie* and higher values of Accuracy, Precision, and F1-Score than AUC-ROC values for the *Ferguson* incident. In all cases, the results are indicative that GCN is able to exploit the relation among the initiator's tweet and its responders, which helped it to perform better.

V. Conclusion and Future Work

Identifying possible *rumor spreaders* is crucial as it has been shown that they are the potential sources of *rumor* propagation [35]. In this work, we use the **PHEME** dataset to identify possible *rumor spreaders* using a weak supervised learning

approach. We model this problem as binary classification task by applying various machine learning models to the transformed rumor spreaders dataset. Our results show that GCN (compared to baseline models) is able to perform better (to raise red flags for possible *rumor spreaders*) by exploiting relationships of possible *rumor spreaders* who could blend well with *non-rumor spreaders*. The overall performance of the GCN shows the effectiveness of this approach. We would like to improve this work through the following multiple plans:

1) **Multiclass problem:** In our present work, we transformed the dataset to study it as a binary classification problem. However, in our extended work, we would like to model this problem as a multiclass prediction to minimize the loss in transformation.
2) **Additional datasets:** The transformation of *rumor* dataset into rumor spreaders dataset may induces bias. To overcome that, we would like to apply this framework on datasets containing all the tweets posted by a user as opposed to only collecting tweets by a specific topic.
3) **New algorithms:** To extend our research, we plan to apply other graph neural network techniques that works well on unseen graph structures (inductive learning) such as Graph Attention Networks [36], GraphSage [37].

VI. ACKNOWLEDGMENT

This research is funded by H2020 project, SoBigData++, and CHIST-ERA project SAI.

REFERENCES

[1] H. Kwak, C. Lee, H. Park, and S. Moon, "What is twitter, a social network or a news media?" in *Proceedings of the 19th international conference on World wide web*, 2010, pp. 591–600.

[2] P. Dizikes, "Study: On Twitter, false news travels faster than true stories," http://news.mit.edu/2018/study-twitter-false-news-travels-faster-true-stories-0308, 2018.

[3] A. Zubiaga, A. Aker, K. Bontcheva, M. Liakata, and R. Procter, "Detection and resolution of rumours in social media: A survey," *ACM Computing Surveys (CSUR)*, vol. 51, no. 2, pp. 1–36, 2018.

[4] M. Fire, D. Kagan, A. Elyashar, and Y. Elovici, "Friend or foe? fake profile identification in online social networks," *Social Network Analysis and Mining*, vol. 4, no. 1, p. 194, 2014.

[5] F. Benevenuto, G. Magno, T. Rodrigues, and V. Almeida, "Detecting spammers on twitter," in *Collaboration, electronic messaging, anti-abuse and spam conference (CEAS)*, vol. 6, 2010, p. 12.

[6] K. Thomas, D. McCoy, C. Grier, A. Kolcz, and V. Paxson, "Trafficking fraudulent accounts: The role of the underground market in twitter spam and abuse," in *Presented as part of the 22nd {USENIX} Security Symposium ({USENIX} Security 13)*, 2013, pp. 195–210.

[7] A. Gupta and R. Kaushal, "Towards detecting fake user accounts in facebook," in *ISEA Asia Security and Privacy (ISEASP)*. IEEE, 2017, pp. 1–6.

[8] S. Gurajala, J. S. White, B. Hudson, and J. N. Matthews, "Fake twitter accounts: profile characteristics obtained using an activity-based pattern detection approach," in *Proceedings of the 2015 International Conference on Social Media & Society*, 2015, pp. 1–7.

[9] M. Salehan and A. Negahban, "Social networking on smartphones: When mobile phones become addictive," *Computers in human behavior*, vol. 29, no. 6, pp. 2632–2639, 2013.

[10] Q. Cao, M. Sirivianos, X. Yang, and T. Pregueiro, "Aiding the detection of fake accounts in large scale social online services," in *Presented as part of the 9th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 12)*, 2012, pp. 197–210.

[11] M. Jiang, P. Cui, A. Beutel, C. Faloutsos, and S. Yang, "Detecting suspicious following behavior in multimillion-node social networks," in *Proceedings of the 23rd International Conference on World Wide Web*, 2014, pp. 305–306.

[12] S. Han, J. Gao, and F. Ciravegna, "Neural language model based training data augmentation for weakly supervised early rumor detection," in *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, 2019, pp. 105–112.

[13] B. H. C. R. Alex Ratner, Paroma Varma, "Weak Supervision: A New Programming Paradigm for Machine Learning," http://ai.stanford.edu/blog/weak-supervision/, 2019, [Online; accessed 10-March-2019].

[14] B. Wang, L. Zhang, and N. Z. Gong, "Sybilblind: Detecting fake users in online social networks without manual labels," in *International Symposium on Research in Attacks, Intrusions, and Defenses*. Springer, 2018, pp. 228–249.

[15] C. Perez, M. Lemercier, and B. Birregah, "A dynamic approach to detecting suspicious profiles on social platforms," in *2013 IEEE International Conference on Communications Workshops (ICC)*. IEEE, 2013, pp. 174–178.

[16] F. Bertini, R. Sharma, A. Iannì, and D. Montesi, "Profile resolution across multilayer networks through smartphone camera fingerprint," in *Proceedings of the 19th International Database Engineering & Applications Symposium*, 2015, pp. 23–32.

[17] F. Chierichetti, S. Lattanzi, and A. Panconesi, "Rumor spreading in social networks," *Theoretical Computer Science*, vol. 412, no. 24, pp. 2602–2610, 2011.

[18] S. Kwon, M. Cha, K. Jung, W. Chen, and Y. Wang, "Prominent features of rumor propagation in online social media," in *2013 IEEE 13th International Conference on Data Mining*. IEEE, 2013, pp. 1103–1108.

[19] M. S. Akhtar, A. Ekbal, S. Narayan, V. Singh, and E. Cambria, "No, that never happened!! investigating rumors on twitter," *IEEE Intelligent Systems*, vol. 33, no. 5, pp. 8–15, 2018.

[20] M. R. Islam, S. Muthiah, and N. Ramakrishnan, "Rumorsleuth: joint detection of rumor veracity and user stance," in *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, 2019, pp. 131–136.

[21] A. Kumar, S. R. Sangwan, and A. Nayyar, "Rumour veracity detection on twitter using particle swarm optimized shallow classifiers," *Multimedia Tools and Applications*, vol. 78, no. 17, pp. 24 083–24 101, 2019.

[22] S. Singhal, R. R. Shah, T. Chakraborty, P. Kumaraguru, and S. Satoh, "Spotfake: A multi-modal framework for fake news detection," in *2019 IEEE Fifth International Conference on Multimedia Big Data (BigMM)*. IEEE, 2019, pp. 39–47.

[23] Y. Wang, F. Ma, Z. Jin, Y. Yuan, G. Xun, K. Jha, L. Su, and J. Gao, "Eann: Event adversarial neural networks for multi-modal fake news detection," in *Proceedings of the 24th acm sigkdd international conference on knowledge discovery & data mining*, 2018, pp. 849–857.

[24] N. Rosenfeld, A. Szanto, and D. C. Parkes, "A kernel of truth: Determining rumor veracity on twitter by diffusion pattern alone," *arXiv*, pp. arXiv–2002, 2020.

[25] T. Bian, X. Xiao, T. Xu, P. Zhao, W. Huang, Y. Rong, and J. Huang, "Rumor detection on social media with bi-directional graph convolutional networks," *arXiv preprint arXiv:2001.06362*, 2020.

[26] P. Wei, N. Xu, and W. Mao, "Modeling conversation structure and temporal dynamics for jointly predicting rumor stance and veracity," *arXiv preprint arXiv:1909.08211*, 2019.

[27] Q. Huang, C. Zhou, J. Wu, M. Wang, and B. Wang, "Deep structure learning for rumor detection on twitter," in *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2019, pp. 1–8.

[28] K. Shu, X. Zhou, S. Wang, R. Zafarani, and H. Liu, "The role of user profiles for fake news detection," in *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, 2019, pp. 436–439.

[29] P. S. Devi, S. Karthika, P. Venugopal, and R. Geetha, "Veracity analysis and prediction in social big data," in *Information and Communication Technology for Sustainable Development*. Springer, 2020, pp. 289–298.

[30] B. Rath, W. Gao, J. Ma, and J. Srivastava, "From retweet to believability: Utilizing trust to identify rumor spreaders on twitter," in *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, 2017, pp. 179–186.

[31] E. Kochkina, M. Liakata, and A. Zubiaga, "All-in-one: Multi-task learning for rumour verification," *arXiv preprint arXiv:1806.03713*, 2018.

[32] S. Nilizadeh, H. Aghakhani, E. Gustafson, C. Kruegel, and G. Vigna, "Think outside the dataset: Finding fraudulent reviews using cross-

dataset analysis," in *The World Wide Web Conference*, 2019, pp. 3108–3115.

[33] Mikolov, K. Chen, and J. Corrado, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[34] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[35] S. Volkova, K. Shaffer, J. Y. Jang, and N. Hodas, "Separating facts from fiction: Linguistic models to classify suspicious and trusted news posts on twitter," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017, pp. 647–653.

[36] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[37] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.