2023 International Conference on Computing, Networking and Communications (ICNC) | 978-1-6654-5719-4/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICNC57223.2023.10074242

Early Rumor Detection in Social Media Based on Graph Convolutional Networks

Niteesh Reddy Thota, Xiaoyan Sun, Jun Dai
Department of Computer Science
California State University, Sacramento, CA 95819, USA
{niteeshreddythota, xiaoyan.sun, jun.dai}@csus.edu

***Abstract*—As a popular source of news, social media provides a platform to spread and gather information, including false information due to its unrestricted fashion of information sharing. Rumor detection is a trending topic in the social networking field, and remains difficult to solve, especially challenged by massive data and large scale in social networking. Most state-of-art efforts use machine learning approaches: hand-crafted textual features from the dataset, rumor classification, and rumor prediction. However, with the availability of abundant information and unreliability in the use of hand-crafted features, models are expected to capture and leverage data features dynamically. This project demonstrates such a model implemented to work on the dynamic features of rumors. Our primary goal is to capture the influential features of rumors and learn dynamic structures built over its propagation for early rumor detection. Specifically, we use Graph Convolutional Networks (GCN) to represent the rumor propagation tree with source and response posts as graphs, and update node representations based on responses to rumors discovered over time. We use a pattern matching algorithm to detect similar sub-graph patterns generated across the graphs for efficient structure reconstruction. Over the updated graph structures is a recursive learning process through which we can accurately predict rumors. This way we deliver an efficient working model that predicts rumors earlier than existing approaches.**

***Index Terms*—rumor detection; pattern matching; GCN; structure reconstruction**

I. Introduction

The emphasis on detecting rumors in social media has evolved over the past decade. With its immense reachability, users participate in information exchange, interact with others, and express thoughts over social media platforms. This easy accessible nature of online media fosters the growth of fake news and promote rumor propagation. We describe rumor as – "a claim whose veracity is unverified or deliberately false [1]". Spread of rumors over platforms such as Twitter, Weibo etc., if left unrestrained, have extreme consequences, and can intentionally deface the fame of organizations.

Automated rumor detection over social media streams is an arduous task because of humongous, unstructured, and incomplete data over social media. Numerous machine learning models were employed on the data accumulated over social media platforms due to the enhancement of Artificial Intelligence (AI) in Computer Science disciplines. Traditional detection models count on feature engineering to extract hand-crafted features: textual information, rumor propagation, and user attributes to train supervised learning models such as Support Vector Machine [2] and Decision Tree [3]. Current classification models have limitations in real-time environment concerning time-consumption and laborious efforts. Numerous classification models employ feature extraction methods from such claim sets and obey four step revision process: label detection, rumor tracking, polarity classification, and veracity prediction of rumor as depicted in Fig. 1. However, feature engineering techniques avoid high-level structure information.

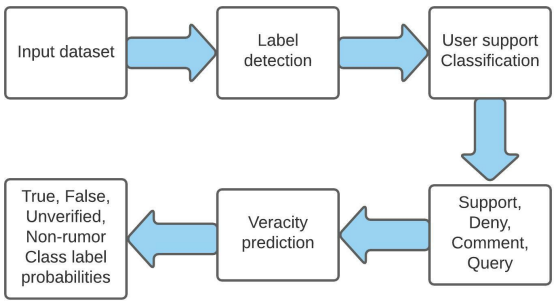


Fig. 1: Rumor Detection and Classification Flow

For approaches of rumor detection in social media that started with manual verification of identifying rumors based on public reports, they demand ample workforce and take plenty of time. Such approaches soon adapted statistical machine learning models that use feature engineering methods and detect rumor as a classification problem.

Most of the supervised learning models over the years employ clustering techniques that focus on recognizing the properties of rumor [4] and obtaining important features of rumor from the content of the posts [3]. Furthermore, Ma et al. [5] and Kwon et al. [6] used propagation patterns and temporal characteristics. These hand-crafted features, content-based, propagation-based, and user-based, were used to train supervised machine learning models of Support Vector Machines (SVM), Clustering Neighbors, Decision Trees (DT), and Random Forests. Because hand-crafted feature extractions are performed manually, they limit these models to capture high-level patterns in rumors.

For early rumor detection and to avoid problems in feature engineering, several works adopted Deep Neural Network models. Application of neural network models captures high-level representation using Convolutional or Recurrent Neural

Networks. Ma et al. [7] presents a Recursive Neural Network (RvNN) model that captures structural and textual properties at each node and updates representations over multiple branches of the tree to detect rumors. These existing approaches ignore the "user relationship network graph" formed based on the dynamic behavior of users over rumor propagation. Besides, limitations over conventional models can be addressed by employing Graph Convolutional Networks (GCN).

Our proposed solution is to create an advanced neural network model that uses Graph Convolutional Networks to learn user representations and update node relationships of the graphs derived from user behavior information. GCN has made great progress in diverse fields, e.g., image classification, relation extraction, text classification, social media, and physical systems. GCN accumulates the information on user relations along with the graphs and can update the node representation with high reliability and better accuracy. By pattern matching to detect sub-graph similarity, GCN can capture propagation information and content-based features from rumor streams.

The main contributions of this work are as follows:

- To the best of our knowledge, this is the first study to integrate a pattern-matching algorithm with GCN feature extraction and structure reconstruction to detect vital rumor patterns.
- GCN updates nodes' representations through its neighbors from the information learned by capturing structure-based information on graph properties such as user node classifications and dynamic node links of responses over streaming media.
- We identify a group of patterns by considering both behavioral and structural features for stance classification over sub-graphs. This series of graph snapshots over time are combined for structure reconstruction recursively to distinguish rumors from non-rumors.
- Evaluation over Twitter and Weibo datasets proves that this approach shows better performance over modern rumor detection methods.

II. Related Work

This section explains different strategies and rationales used for detecting rumors in social media platforms.

A. Supervised Machine Learning Approach

Traditional approaches do manual rumor detection through investigative journalism. Few online firms such as snopes.com [9] and factcheck.org [10] depend on public surveys and manually verify statements. These approaches are laborious, time-consuming, and require extensive resources.

Models based on supervised machine learning techniques depend on content, propagation, and user features. The content-based feature set has textual data and sentiment of text in claims. User-based feature sets contain data on users' activity, personal information, and history of posts. The propagation-based feature set consists of a tree-based structure of source and response posts with multiple branches.

Supervised learning methods collect a group of false rumor posts over online platforms and extract feature sets to train various machine learning models that predict the veracity of a claim. Subsequent approaches for automatic detection conduct feature engineering to form clusters of rumor characteristics used for classification problems. A decision-tree classifier model to engineer various statistical features manually was proposed by Castillo et al. [3]. A SVM time-series model to learn textual properties based on chronological order was proposed in works of Ma et al. [5]. Kwon et al. [6] used random forest classifier model to capture temporal properties and feature sets of users, textual, and structural properties.

Approaches on the tree-based structure to model information propagation over kernels [12] showed profound results over other existing supervised machine learning models. Though these models demonstrate accurate classification of rumors, they limit capturing high-level patterns and structural information, and heavily rely on hand-crafted features.

B. Deep Neural Network Approach

A Convolutional Neural Network (CNN) used to detect the stance of the claim was proposed by Chen et al. [13] to determine rumor veracity. From the tree structure formed with the source-response relationship, the CNN model extracts the feature vector at each node from textual data and presents this vector to neural networks. Another type of neural network primarily used in this field is Recurrent Neural Networks (RNN). A RNN model was proposed by Ma et al. [14] that learns the representations of individual posts in the tree structure and captures the contrast in content-based information among relevant posts over rumor propagation time. LSTM or Gated Recurrent Units (GRU) are used as hidden layers in RNN model to improve the performance. Liu et al. [15] proposed a combination of RNN and CNN with hidden units to extract a user-based feature set over time. Ma et al. [7] worked on rumor detection using the Recursive Neural Network (RvNN) model that demonstrated the best performance over other existing models and captured both content semantics and propagation tree structure. The model updates node features selectively based on the connections and propagation path of the branch.

III. Problem Statement

We define rumor detection problem as a set of posts P = $\{p_1, p_2, ..., p_n\}$ with corresponding label set {True, False, Non-rumor, Unverified} composed of users with source post that launch the chain of messages along with the responsive posts forming a tree-structure with multiple branches. Each post consists of a set of tweets T = $\{t_0, t_1, ..., t_n\}$, where t_0 is root tweet of the corresponding post. A model forms a graph from the user response relationships over propagation tree structure where edges represent the relation between nodes, e.g. e^i_{ab} is an edge that represents response relationship between $user_a$ and $user_b$. An adjacency matrix 'adj' is defined to keep track of neighbors that influence node characteristics in the graph. The model identifies common patterns and reconstructs graph structures recursively over

time. We construct an "NxF" array with 'N' users and 'F' node characteristics and an 'NxN" adjacency matrix from the feature sets formed at each node. We set fields of a row in the adjacency matrix adj_{ab} to '1' if the current $user_a$ has a well-established edge with other nodes such as $user_b$ in the graph and '0' if there is no connection. GCN extracts textual feature set F = $\{f_0^i, f_1^i, ..., f_n^i\}$ and user stance set information {support, deny, question} at each node. Structure reconstruction increases the probability of differentiating true and false rumors. Fig. 2 represents user relationship graph structure. We calculate probability distribution of each tweet ID among labels, i.e. {true, non, unverified, and false} rumor for Twitter; {true and false} rumor for Weibo.

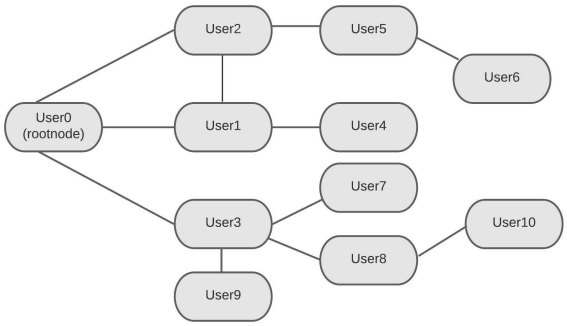


Fig. 2: User Relationship Graph Structure

IV. Proposed Model

Our proposed solution (depicted in Fig. 3) effectively applies Graph Convolutional Networks (GCN) that operate on user relationship graphs to learn high-level node characteristics. The three core components considered are graph structure reconstruction over dynamic pattern changes of propagation over time, application of GCN to accurately capture influential factors: content, user, and propagation information, and early rumor detection over online media.

A. GCN Feature Extraction

GCN performs graph classification to predict the associated class label, node classification to understand the features and stance of a node, link prediction to assemble all relations between nodes, clustering to group similar nodes based on their behavior, embedding to depict characteristics into a vector for better perception, and graph reconstruction to generate new graphs with information learned by models.

Computing over neighbors' features updates the latest node and models the flow of information from source to current node. Knowledge on neighbors of a user in a relationship graph is denoted by the corresponding row of an adjacency matrix. GCN extracts content semantics and user stance: support, deny, question information at each node. We replace Gated Recurrent Unit (GRU) with GCN in RvNN to update all users simultaneously over multiple branches of a tree. We concatenate the root node feature vector at every node to ensure the influence of source post across rumor propagation.

$$G_1 = GCN(input, adj) \tag{1}$$

$$G_2 = GCN(G_1, adj) \tag{2}$$

G_1 and G_2 are hidden GCN layers to improve the model's learning capability and update nodes' representations recursively in a graph. The input for the GCN model is a user relationship graph structure over rumor propagation with textual and stance information of relative posts at each node. An adjacency matrix 'adj_{dxd}' is formed with a degree 'd', where 'd' represents number of users in the relationship graph. GCN learns neighbor features based on the number of convolutions and updates nodes properties in the co-relation graph.

B. Pattern Matching

Algorithm of GraphPatternMatch(RS, G_p)

Input: graph G
for $edge =< nodestart, nodeend, timestamp > \in RS$ do
  for $\forall newnode \in \{nodestart, nodeend\}$ do
    addNode(newnode) in G
    for $\forall pattern \in G_p$ do
      if patternCheck($nodestart, edge, pattern$) then
        updateGraph(G)
      end if
    end for
  end for
end for

After creating a graph structure using GCN composed of content, user information at each node, and propagation relations represented as edges of the node, we present this series of edges as input to our pattern matching algorithm. This section of the model processes data over the stream recursively and yields a list of matched patterns and their occurrence times. The index data structure used represents the label information of all nodes in a sub-graph. GCN updates node relations depicted by the edges of a graph on iterations and this information assists model in discovering sub-graph patterns at each interval.

Each edge consists of two nodes and their time of generation. The two most common patterns over the stream are 'star' and 'path', where star structure parts into multiple branches and path structure have a single path of propagation over sub-graph. We provide the data as a stream of edges in a graph and a query set of the most basic patterns as input to the algorithm. For every edge encountered, all the new nodes of the edge are added to the graph. This updated graph is compared with every pattern declared in the query set to point out matches. Then the stance label of each node is identified and compared with a root node of the given pattern to check whether they share the same label. If they share the same label, then all such nodes are represented as a root node. Root node's label information

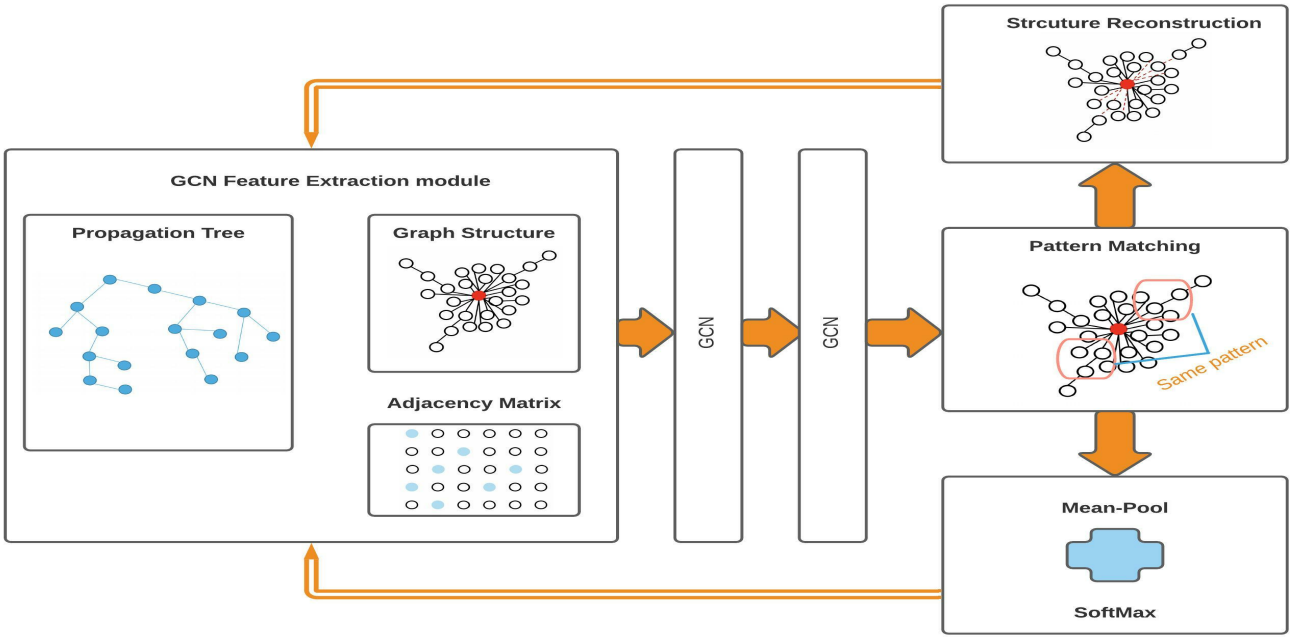


Fig. 3: Overall Architecture of GCN Model

is preserved in the index data structure that stores label-related information of each node; hence, the algorithm identifies other nodes linked to the root node but share a different label and influences the stance of the root node. In the final step, the algorithm updates the main graph with its matches over the new edge. All the new patterns formed along with their time stamps are updated in the index data structure.

Information collected over the graph structure using the pattern matching algorithm differentiates rumor from non-rumor. In general, the frequency of pattern matches inflates for a rumor over a stream of tweets, while patterns are not encountered often for non-rumors. All edges of the propagation tree are represented as a stream of relations RS = $\{r_1, r_2, ..., r_n\}$ and a pattern set for sub-graphs as $G_p = \{gp_1, gp_2, ..., gp_n\}$.

C. Dynamic Structure Reconstruction

The proposed model applies reinforcement learning on text, propagation, user, and structure information of a rumor. Node characteristics in a graph network are recursively updated when a new edge has formed over a network of users. GCN updates the node by combining neighbors' attributes with its input on every iteration based on the addition of new users influencing the nodes in the graph. Gaussian distribution is applied using GCN to measure mean and standard deviation to present deviation of values in adjacency matrix.

$$f(x) = \mu + k\sigma \tag{3}$$

$$adj_{ab} = \sqrt{\frac{\sigma * a_i - \mu}{N}} \tag{4}$$

$$G_3 = \sigma(adj.adj^T) \tag{5}$$

$$G_4 = GCN(G_3, adj) \tag{6}$$

Information constructed from latent representations constitutes a new graph 'G_4' with enhanced feature sets generated at each node in user relations as 'G_3'. We can reconstruct the graph with an updated user set using the sigmoid activation function. Gaussian distribution updates adjacency matrix adj_{n*n}. This GCN model trains by improving the total reconstruction loss and calculating cross-entropy loss to measure the changes in the adjacency matrix from the actual values. During structure reconstruction, to enhance root node effect over the propagation tree, we fuse the feature vector of the root node with the feature vector of all nodes in the graph.

D. Rumor Detection

Input to this module is the latent representation of feature sets extracted recursively from nodes of the graph data. We implement mean pooling functions to accumulate complete information around the node's representation over a network. Mean pooling reduces the dimensions of feature maps by calculating the average of related features over every iterative step and is represented as (7), where 'G_4' is the hidden feature vector representation.

$$S = Mean - pooling(G_4) \tag{7}$$

To predict the class label of the claim, we use the SoftMax function and a fully connected layer. The probabilities of a post associated with a class label are calculated, and we regard the label with the highest priority as the actual class label of the post. A SoftMax function is defined as (8), where 'FC' represents a fully-connected layer and 'S' is the vector

obtained after mean-pooling of features. Hence 'y' returns calculated values for each class label.

$$y = SoftMax(FC(S)) \tag{8}$$

$$loss = l_{rec} + l_{kl} + l_{det} \tag{9}$$

We perform Mean pooling and SoftMax at every iterative step from data collected over structure reconstruction. We train all these modules together to reduce the cross-entropy loss of the model. The Sum of structure reconstruction loss (l_{rec}), label detection loss (l_{det}), and kl divergence loss (l_{kl}) adds up to the final loss of the model.

V. Experiment

In the following section we evaluate the accuracy of our proposed solution with respect to other state-of-art models.

A. Datasets

We use Twitter15 and Weibo datasets collected from the works of Ma et al. (2017) [19]. The data collected is presented as a set of claims or statements with corresponding veracity labels. Each statement is composed of a group of tweets with source and responsive claims. These datasets contain propagation trees and relative users. In each tree, nodes represent users and edges represent response relations. Trees are labeled as: True rumor (T), Unverified rumor (U), Non-rumor (N), and False rumor (F) in Twitter15 dataset and one of the two labels: True rumor (T) or False rumor (F) in Weibo dataset. Statistics of data in these datasets are outlined in Table 1.

TABLE I: Statistics of Twitter Dataset [20]

Field	Twitter15	Weibo
No. of posts	331,612	3,805,656
No. of Users	276,663	2,746,818
No. of claims	1490	4664
Avg. time length / event	1,337 hrs.	2,460 hrs.
No. of True rumors	374	2351
No. of False rumors	370	2313
No. of Non rumors	372	0
No. of Unverified rumors	374	0
Avg. No. of posts / event	223	816

B. Experiment Setup

We train the proposed model over a set of distinct parameters throughout the process recursively. The overall process is iterated over 50 epochs with a batch size of 128 training samples per each iteration. The hidden layer dimension at each node is 64 and the learning rate is equal to 0.005 (5e-4). We split data into five folds for cross-validation where each fold constitutes a test set.

The early stopping method is applied for every ten epochs to monitor the validation loss. A training set of data is further divided into two parts where 80% of data is used as a train set and 20% as a test set. For valuation metrics, we use accuracy and F1-score of true and false rumor class in Weibo dataset and accuracy and F1-score of true, false, unverified, and non-rumor class in Twitter data set.

We perform the DropEdge method to enhance graph structure by removing trivial connections with a drop rate of 0.2. Results of validation metrics determine the performance of the model. To train the model and monitor the learning rates of feature sets, we use Adam optimizer. Supervised learning models such as DT and SVM are implemented with scikit-learn API, Neural Network models such as RNN and CNN implemented with Keras API, and Recursive Neural Network and the proposed GCN-based model with Pytorch libraries.

C. Baselines

We compare the results with existing learning approaches. Input to these models is a dataset divided randomly. A cross-validation method is performed to achieve accuracy of rumor detection over already defined class labels.

- **Decision Tree [21]**: Classifies the clusters formed using regular expressions defined to detect rumors
- **Random Forest Classification [6]**: Manually extracts feature sets and temporal properties to detect rumors
- **SVM-TS [5]**: Considers structure over time to see variations in content, user, and propagation attributes
- **RNN-GRU [14]**: Predicts rumor over the tree structure by extracting textual contents of posts sequentially
- **RvNN [7]**: Learns information from the top-down and bottom-up tree propagation structure
- **Bi-GCN [20]**: Extracts textual and propagation features using GCN
- **Proposed model**: Our model using GCN extracts feature sets, performs pattern matching, and dynamically reconstructs graph structures

TABLE II: Results on Twitter Dataset

Method	Accuracy	F1			
		T	***F***	***N***	***U***
SVM	0.535	0.404	0.472	0.796	0.483
DT	0.450	0.538	0.423	0.481	0.392
RNN	0.638	0.685	0.630	0.678	0.568
RvNN	0.718	0.82	0.75	0.675	0.66
Bi-GCN	0.776	0.86	0.78	0.745	0.745
Proposed-model	**0.80**	**0.86**	**0.80**	0.75	**0.77**

VI. Performance Analysis

A. Rumor Classification Performance

We can see from the above tables that the proposed model achieves superior performance over other existing approaches.

TABLE III: Results on Weibo Dataset

Method	Accuracy	F1	
		T	***F***
SVM	0.76	0.752	0.786
DT	0.731	0.719	0.731
RNN	0.795	0.78	0.801
RvNN	0.857	0.823	0.882
Bi-GCN	0.88	0.875	0.885
Proposed-model	**0.911**	**0.90**	**0.918**

The model is developed in python programming language using torch libraries to implement neural network models.

Supervised learning models such as DT and SVM based on manual feature extraction techniques perform inadequately. DTR method collects related posts that match a set of regular expressions and ranks the rumor clusters. As only a few posts match such expressions, this method is the least accurate method to perform rumor detection. DNN models learn high-level features of rumors that constitute accurate prediction of rumors. We consider the RNN model with GRU hidden units to remember long-term sequences that classify rumors based on textual content.

RvNN model that learns tree information is used as a baseline to compare results with the proposed model. RvNN method performs the max-pooling operation on leaf nodes over multiple branches, hence results primarily depend on the most recent posts in the tree, which lacks vital information. GRU units used perform propagation sequentially, one branch at a time, hence updating nodes' representation over that branch limits its performance compared to our GCN model that simultaneously updates all users in the graph. The proposed model gives the highest accuracy because it uses GCN to dynamically reconstruct the graph structured data.

B. Structure Performance

To observe the effect of the structure reconstruction module, we compare our model with existing GCN-based models that extract features from graph data and perform classification techniques. Graph structure outlines the user relationship, and GCN updates nodes representations based on the new post encountered over social media streams.

Structure reconstruction performed recursively captures the changes in nodes attributes and updates all adjacent users simultaneously. The pattern matching algorithm identifies similar rumor patterns. It reduces the problem by considering a single representation over such similar rumor patterns; this aids in reconstructing graphs by removing trivial users and hence learning high-level features and better representations. We improved the overall performance of rumor detection by operating on structure information.

C. Case Study

To exhibit the performance of the model, we present case studies of hand-picked tweet IDs as part of test set of data. Graph of each tweet chain is represented as a 'tweetid.npz' file. We determine the class label of the tweet based on the probability distribution among all classes: true, false, unverified, and non-rumor. From the data gathered on performing the experiment, we outline the classification of few test claim IDs in Table 4 and Table 5.

TABLE IV: Probability Distribution of Claim IDs

ID	Claim	Probability distribution			
		T	***F***	***N***	***U***
1	516804849619705859	0.06	0.7952	0.0198	0.125
2	565999191982616577	0.8062	0.0091	0.1080	0.0767
3	692742353736568833	0.1567	0.033	0.7721	0.0380
4	553467311261503488	0.0977	0.096	0.0041	0.8022
5	614054616154550273	0.0641	0.8006	0.080	0.0553

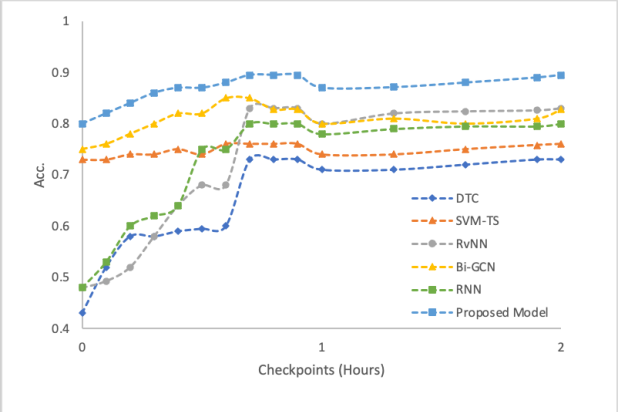
TABLE V: Case Study Results

ID	Actual label	Accuracy	Predicted label	Result
1	false	0.7893	false	pass
2	true	0.8057	true	pass
3	non-rumor	0.7935	non-rumor	pass
4	unverified	0.8015	unverified	pass
5	false	0.8083	false	pass

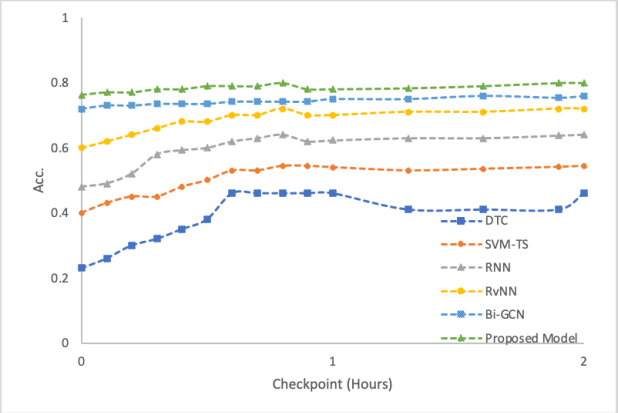
To elaborate more on Table 4, for the Claim ID 614054616154550273 – this tweet is regarding a man named Dylann Roof, who shot all African Americans in a church in Carolina. He was sentenced to death, but a user named NewsWatch33 claimed on Twitter that "Dylann Roof's defense fund received more than $4 Million in donations from supporters" [22]. This tweet created chaos and influenced the perspectives of many over his case. After investigation, it was found that the tweet was fake and NewsWatch33 was a fake website. So, this case is classified as a False case in the Actual class label column in the dataset. The probability distribution from the proposed model also indicates the same, with the highest probability being 0.8006 for False. With the predicted class label matching the actual class label, the test results show passed. The same approach is used to classify all the claim IDs in the test set.

D. Early Rumor Detection

Early detection of a rumor is a critical metric to estimate the quality of a model. By detecting rumors early in their stages, we can intervene and debunk such posts from social media. The early detection method is constructed by setting up deadlines for detection and considering the posts that occur before the deadline based on their timestamps. We add test data incrementally to the model based on time delays to evaluate the performance of early rumor detection. Learning models



a) Weibo



b) Twitter

Fig. 4: Early Rumor Detection Results Across Time

such as RNN+CNN cannot process data of variable length, hence accuracy achieved at each deadline is considerably low.

Fig. 4 outlines the performance of the proposed GCN model with existing models based on DT, SVM, RNN, and RvNN methods at each deadline checkpoint over datasets. The findings indicate that the proposed model achieves improved accuracy early in propagation process from the source post. The model outlines better accuracy at each checkpoint over existing methods.

VII. Conclusion

We present a persistent GCN model to forecast false information on social media in this paper. This model comprises four modules: feature extraction, pattern matching, structure reconstruction, and rumor detection. The fundamental nature of the Graph Convolutional Network (GCN) used gives our model the ability to process graph structures formed from propagation tree information and learn high-level feature representations of nodes over the network. GCN dynamically updates all nodes' representations upon inclusion of the latest post in the network. Pattern matching algorithms discover similar patterns over sub-graphs and cluster them to improve the efficiency of the model. The structure reconstruction module recursively eliminates trivial users and forms a most effective graph structure with updated feature sets on each iteration. All these modules are trained jointly and results on real-world datasets denote that in terms of detecting false information accurately early in propagation, the proposed model outperforms existing approaches.

Future work can be conducted from the following directions.

- More sophisticated information such as user properties and credibility information can be used to enhance the graph-based representation learning to detect rumors.
- Extend the working of this rumor detection method in real-time to evaluate its performance in social media.
- Introduction of unsupervised machine learning techniques to operate on unlabeled datasets and comparative analysis between unsupervised and supervised rumor detection models can be performed.

Acknowledgement

Xiaoyan Sun is supported by NSF DGE-2105801. Jun Dai is supported by NSF DGE-1934285 and NSF DGE-2105801.

References

[1] N. D and B. P, "Rumor psychology: Social and organizational approaches," American Psychological Association, vol. 1, pp. 1-34, 2007.
[2] Y. F, L. Y, Y. X and Y. M, "Automatic detection of rumor on Sina Weibo," in ACM SIGKDD Workshop on Mining Data Semantics, 2012.
[3] C. C, M. M and P. B, "Information credibility on twitter," in WWW 2011.
[4] L. DM, B. MA, B. Y, B. AJ, G. KM and M. F, "The science of fake news," Science 359(6380), p. 1094–1096, 2018.
[5] M. J, G. W, w. Z, L. Y and W. KF, "Detect rumors using time series of social context information on microblogging websites," in CIKM 2015.
[6] K. S, C. M, J. K, C. W and W. Y, "Prominent features of rumor propagation in online social media," in ICDM 2013.
[7] M. J, G. W and W. KF, "Rumor Detection on Twitter with Tree-structured Recursive Neural Networks," in ACL 2018.
[8] H. Lin, Z. Xi and X. Fu, "A Graph Convolutional Encoder and Decoder Model for Rumor Detection," in DSAA 2020.
[9] "Snopes," Snopes Media Group Inc. https://www.snopes.com/.
[10] "FactCheck," A Project of The Annenberg Public Policy Center of the University of Pennsylvania. https://www.factcheck.org/.
[11] A. r. Pathak, A. Mahajan, K. Singh, A. Patil and A. Nair, "Analysis of Techniques for Rumor Detection in Social Media," in ICCIDS 2019.
[12] W. Ke, S. Yang and K. Q Zhu, "False rumors detection on sina weibo by propagation structures," in ICDE 2015.
[13] C. YC, L. ZY and K. HY, "IKM at SemEval-2017 Task 8: Convolutional Neural Networks for stance detection and rumor verification," in SemEval-2017.
[14] M. J, G. W, M. P , K. S and w. KF, "Detecting Rumors from Microblogs with Recurrent Neural Networks," in IJCAI 2016.
[15] L. Y and W. YFB, "Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks," in AAAI 2018.
[16] "Twitter Developer Platform." https://developer.twitter.com/en/docs/authentication/overview.
[17] "Towards Data Science." https://towardsdatascience.com/downloading-data-from-twitter-using-the-rest-api-24becf413875.
[18] K. NT and W. M, "Semisupervised classification with graph convolutional networks," in ICLR 2017.
[19] M. J, G. Wei and W. KF, "Detect rumors in microblog posts using propagation structure via kernel learning," in ACL 2017.
[20] T. Bian, X. XI, X. T, Z. P, H. W, R. Yu and H. J, "Rumor Detection on Social Media with Bi-Directional Graph Convolutional Networks," in AAAI 2020.
[21] Z. Z, R. P and M. Q, "Enquiring Minds: Early Detection of Rumors in Social Media from Enquiry Posts," in WWW 2015.
[22] K. LaCapria, "Snopes." https://www.snopes.com/fact-check/dylann-roof-donations/.