International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
***Article Received:** 05 July 2023 **Revised:** 25 August 2023 **Accepted:** 15 September 2023*

Subgraph Anomaly Detection in Social Networks using Clustering-Based Deep Autoencoders

[1,*]Yallamanda Rajesh Babu, [2]G. Karthick, [3]V.V. Jaya Rama Krishnaiah
[1]Research Scholar, Dept. of Computer Science and Engineering,
Annamalai University, Annamalai Nagar, Tamilnadu – 608 002
E-Mail: yrajeshbabu7@gmail.com
[2]Assistant Professor, Dept. of Computer Science and Engineering,
Annamalai University, Annamalai Nagar, Tamilnadu – 608 002
E-Mail: karthick18588@gmail.com
[3]Associate Professor, Dept. of Computer Science and Engineering,
ASN Women's Engineering College, Tenali, Andhra Pradesh – 522 201
E-Mail: jkvemula@gmail.com

Abstract—Social networks are becoming more prevalent all across the globe. With all of its advantages, criminality and fraudulent conduct in this medium are on the rise. As a result, there is an urgent need to detect abnormalities in these networks before they do substantial harm. Traditional Non-Deep Learning (NDL) approaches fails to perform effectively when the size and scope of real-world social networks increase. As a result, DL techniques for anomaly detection in social networks are required. Several studies have been conducted using DL on node and edge anomaly detection. However, in the current scenario, subgraph anomaly detection utilizing Deep Learning (DL) is still in its nascent stages. This paper proposes a method called Clustering-based Deep Autoencoders (CDA) to detect subgraph anomalies in static attributed social networks. It converts the input graph into node embeddings using an encoder, clusters these nodes into communities or subgraphs, and then finds anomalies among these subgraph embeddings. The model is tested on seven open-access social network datasets, and the findings indicate that the proposed model detects the most anomalies. In the future, it is also recommended that the present experiment be aimed at dynamic social networks.

Keywords- Subgraph Anomaly Detection, Graph based Anomaly Detection, Unsupervised Learning, Attributed Social Networks, Subgraph Embeddings, Dense Autoencoders.

I. INTRODUCTION

In today's world, it is impossible to contemplate not having access to the world of internet and social media. It has evolved into a fundamental part of who people are. It was a revolution that changed the core framework of communication throughout the globe. A typical web user takes up more than two hours per day on social networks, and roughly one-third of the world's population currently uses them [1]. It offers a wide range of advantages. It enables and maintains human interactions on a worldwide scale. Additionally, it assists companies in brand promotion, therefore indirectly lowering marketing expenses. It is also an excellent teaching resource. There are many more perks of social networks, and with new users joining the social network every day, it is no more the virtual world. It has now evolved into a universe in itself, except that it is very easy to get away with things here due to its anonymity.

Social media has many advantages, but it also raises a lot of problems. Online information and data availability draw a sizable number of fraudulent users and crooks. These social networks are now used to commit various criminal acts [9][29]. Social networks are also the new tool for several current crimes as well as the production of fresh crimes, ranging from national terrorism through radicalization to violence against a single person like stalking, harassment, cyberbullying, or hate speech [18][14][16][30]. This is also because these fake individuals benefit from social networks' anonymity. Additionally, there is an urgent need to develop tools to identify these users given the degree of influence they might have on society. As crucial as it is to identify each user, there are growing social network groups of these fraudsters that adhere to organized crime and illicit activity. Therefore, identifying these fraud groups among ordinary social network users is a huge problem in today's society.

When it comes to social networks, an anomaly is defined as an unanticipated behaviour by a person or a bunch of users that deviates from the norm for network users [3]. An observation (or group of observations) that considerably differs from the other observations in the sample [12] or that doesn't seem to fit the other data can also be referred to as an outlier [2]. Finding certain substructures or patterns that are unanticipated, undesired and must be discovered to protect the networking and its users is known as anomaly detection. A danger to the network does not necessarily accompany an abnormality. To prevent the true threat among them from being undiscovered, it is vital to take note of any aberrant network units. In these circumstances, even if there are some false positives, the emphasis is on finding all anomalous units.

The most popular framework for mathematically representing a social network is a graph, therefore developing ways to spot anomalies in these networks is essential. A lot of research has been done on node and edge anomaly detection. Subgraph anomaly detection, alternatively, is a little less explored territory. This could be because determining what constitutes a subgraph and what is merely a linked component requires taking into account, both attribute closeness and structural proximity in addition to other factors. This suggests, however, that many complicated anomalies can be found via subgraphs, as nodes or edges that do not appear abnormal when viewed separately may exhibit aberrant characteristics when linked to a community. Therefore, it is essential to consider subgraphs to find hidden abnormalities that other methods miss.

The approach for subgraph anomaly detection proposed in this paper uses network encoders, dense autoencoders, and clustering techniques for graph coarsening to arrive at anomalous subgraphs using derived labels, node embeddings, and node clusters. The presented model is examined on various datasets, and the outputs are promising related to the existing modern techniques.

II. LITERATURE SURVEY

Subgraph anomaly detection is a slightly underexplored area as there haven't been many studies done on subgraph anomaly detection, and there are also fewer studies that employ DL methods. However, the existing works have significantly influenced the accomplishments in the aforementioned field up to this point. Most NDL model-based works can be largely classified into techniques that are enforced to dynamic or static graphs, along with attributed or non-attributed graphs.

[6] was one of the first efforts in this area, using minimum description length to find anomalous substructures in static unattributed networks. This technique finds frequent substructures using greedy beam search and rates them according to the minimum description length [22], considering substructures with higher description length as more anomalous ones. [10] also used minimal description length for finding the normative pattern in static unattributed graphs and developed three techniques for graph-based data fraud detection and prevention in which they categorize graph modifications into three classes, such as alteration, vertex/edge deletion and insertions, one for each of the algorithms. The first algorithm identifies patterns with a lower cost of transformation and frequency as anomalous. The second one looks for extensions of normative patterns and considers the less probable ones as anomalous. The third algorithm also uses transformation cost to check for anomalous patterns amongst the ancestors of the normal patterns as well as the highest potential substructures of it.

[24] distinguish the attacking group from a social subgraph with the presence of external triangles using randomized graph traversal in static unattributed graphs. By employing a random selection model, the attacker node selects a cluster of victim nodes for connecting inside a widespread assault variability on communication networks. The prime factor that differentiates the attacking group from a social subgraph is the presence of external triangles, which the attacker node establishes with the network balance and will comprise one attacker and two non-attacker nodes. These triangle numbers will be quite less for a malevolent node. An attacker node will be connecting to several other attacker nodes situated all over the network to avoid being found. It is improbable that several victim nodes will have access to similar good node edges in the attacker subset neighbourhood and fewer victim nodes in the neighbourhood consisting of attackers, victims, and fewer good nodes. Nodes present in the neighbouring area with influential links to the subset will therefore be a victim or attacker node. If any node consists several triangles than a particular threshold, it is possibly an attacker node.

[19] proposed a signal processing-based recognition theory for discrepancies in undirected, unweighted graphs that makes use of the L1 features of the graph's modularity matrix [20]. By projecting the large graph into its two primary eigenvector space, calculating a Chi-squared testing statistic, and associating the outcome to a threshold, the principal eigenvectors evaluation of the modularity matrix reveals the existence of a minor, firmly associated component embedded in the larger graph. The model computes the modularity matrix's eigendecomposition for the graph, determines every eigenvector’s L1 norm, and later takes away the predicted value, normalizing the output by the L1 norm. The existence of an anomalous subgraph embedding is specified if any of these altered L1 norms falls below a predetermined threshold.

[26] proposes a pre-processing model where a set of local vertices with higher similarity of including the anomalous vertex is effectively attained by subgraph search. The sparse background graph that are modelled by the Chung-Lu random graph consists a compact anomaly graph fitted with the Erdos-Renyi technique. The abnormal vertex is illustrated by the priori adjacent matrices that are utilized by the subgraph search model is constructed for condensing the global into a small vertex set. The largest abnormal coefficient initially determines the starting point of each vertex set. Later, based on the largest co-efficient amidst the revised value for every any of them, vertex from the primary vertex’s neighbouring matrices is selected, and the balance vertex are supplemented in the similar manner. A set containing largest co-efficient is selected as the utmost anomaly amidst the overall sets. This is performed for every graph snapshot for producing various local sets that are later integrated for creating the concluding set. A recognition statistic is

enforced to this concluding vertex set for determining if the graph is anomalous.

[7] recommends utilizing a weighted cumulative graph from the dataset to estimate the similarity of occurrences and discover activities of anomaly in volatile time-evolving unattributed networking using the product rule for the central limit theorem. This model primarily builds a base for standard behaviour by discovering consistent patterns amidst the vertex that are a collection of vertices which forms a related component and interconnect frequently. It later duplicates a weighted "cumulative" graph from the time-evolving network dataset that prioritizes recent edges. It compares the activities in any specified time with anticipated activities based on previous behavioural trends for recognizing anomaly.

[13] proposes a max-margin framework to determine the outliers of a subgraph which relates the margin for linked to non-linked pairing of nodes closer to a subgraph match for determining the outlier scoring that are employed for ranking like subgraph outliers. By plainly listing the overall graph edges that are covered by each match, it is easy for computing the overall induced match sets. The match's outlier score is calculated by utilizing the margin for the finest feature weight vector or the max-margin hyperplane. Afterwards the arrangement of the matches accordingly to their outlier scores in a non-increasing order, the topmost fewer matches might be sent back as outliers. [23][25] suggest a generic framework which recognizes the related anomaly property subset along with an abnormally associated subgraph. It maximizes an anomaly score function by implementing a sophisticated non-sequential non-parametric scan statistic function set that are implemented for formulating the functions utilized for estimating the anomalous subgraph behaviour and the subsequent attribute subset. The graph is approximated as the tree from a root node selected randomly by employing the tree approximation priors and finding the optimum subtree in the tree and the attributes related with it shows the majorly anomalous related subgraph and its associated quality.

[8] broadens the original Subdue approach to include numerical outliers as well as K-Nearest Neighbours by modifying the graph such that all standard edges contain a static value while anomaly edges estimate to a value collection by using K-Nearest Neighbours, it differentiates between anomaly and standard values. [5] creates a list of an overall crucial anomaly positions in a dynamic networking. It starts by outlining the usual characteristic of the network and rank edges over time accordingly to their abnormal behaviour. Provided with an edge and its weight in any exact time, the timestamp percent where the similar edge has an equal or superior weight logged on it is computed, which decreases with an increase in abnormality.

All of them, although they are classic, NDL procedures. Even though there were various researches on node and edge anomaly recognition using DL, there has been less research on subgraph anomaly recognition in social networking.

[27] proposed DeepFD, a model that learned the unusual occurrence depictions of users so that beginning users are distributed across the vector space meanwhile suspicious users who belong to a single group are closer to one another, taking into account that user nodes related with specific deceitful groups are most likely to have associations with similar item nodes. It compares two users' behavioural similarities as the fraction of common traits across all of the items they've looked at. An autoencoder is trained to create user representations. The distrusted dense blocks that are projected for producing dense fields in the feature space are later identified by employing DBSCAN [11].[31] suggested FraudNE, a model that leverages the dense block detection technique to detect dishonest individuals and related changed items in online review networks represented as bipartite graphs. It groups suspicious people and things from the same dense block together while randomly dispersing other items instead of encoding both node types into a shared latent space, as DeepFD works.

[17] Formalized the problem of anomalous subgraph detection as a binary with the null hypothesis representing a normally detected graph and the alternate hypothesis representing a background graph containing an inconsistent subgraph, and proposed a framework for recognizing subgraphs by implementing Deep Neural Networks. It comprises both an offline training stage where instances are sent to the Hidden Layer (HL) for generating feature maps to capture the graph state, and a training set is constructed based on the neural network's specific form and an online recognition phase which construct the feature vector based on the fed input, and the recognition statistic is determined. This statistic, when associated to the threshold, concludes whether or not the observed graph has an abnormal subgraph.

The proposed technique uses autoencoders on coarsened graph node embeddings to detect subgraph abnormalities in static attributed social networks.

III. CLUSTERING-BASED DEEP AUTOENCODERS

Provided a network G = {V, E}, in which G is the graph denoting the social network, and V, E are the vertex and edge graph sets, the study aims to identify rare subgraphs that vary significantly from the major reference subgraphs in terms of node structure and attribute data and also to build a DL framework that can include non-Euclidean data about graph structure, used to symbolize the social network for subgraph anomaly detection. The concept is presently only applicable to static attributed graphs.

As anomalies in real-world social networks are seldom marked or labelled, unsupervised learning approaches were utilized for subgraph anomaly detection in this study. The approach embeds graphical data into low-dimensional latent depictions, clusters them, and then finds anomalies within the clusters. Figure 1 depicts the abstract framework of CDA.

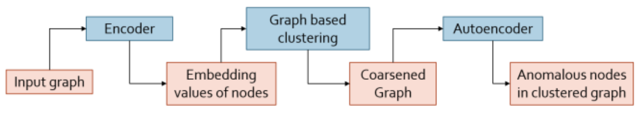

Figure 1. High-Level Abstract Representation of the CDA Model Framework

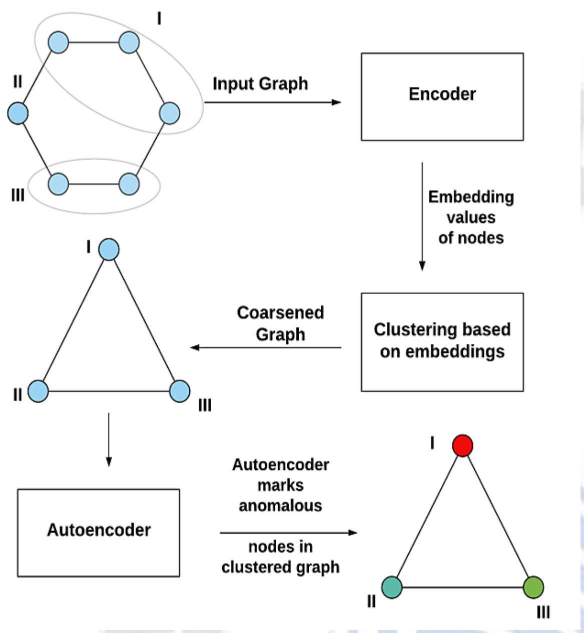
I
II
III
Input Graph
Encoder
Embedding values of nodes
Clustering based on embeddings
Coarsened Graph
Autoencoder
Autoencoder marks anomalous
nodes in clustered graph

Figure 2. The Proposed CDA Construction

CDA is broken down into three parts, an encoder, a clustering technique, and an autoencoder. To build node embeddings, the encoder takes into account both attribute information and network topology. These embeddings are used by graph-based clustering to detect similarities between nodes and cluster them into super nodes, resulting in a coarsened network. This stage effectively reduces the subgraph anomaly problem to a node anomaly problem, with each super node in the coarsened graph representing a subgraph in the original input network. These super nodes are then sent into the autoencoder, which determines anomalous super nodes, which are effectively the input's anomalous subgraphs. Figure 2 denotes the model construction.

A. Network Encoder

An encoder is a neural network that accepts an attributed graph as input and creates feature vectors or node embeddings. The objective is to learn node representations while retaining structural and attribute proximity. If a link/edge exists between two nodes, it shows structural proximity, and the attribute juncture of the two nodes denotes attribute proximity. The network encoder generates an NxF feature matrix, where N and F signifies the node number in the graph and the output feature number per node. The encoder follows the architecture of [15], where the encoder contains input, embedding, hidden, and output layers as depicted in Figure 3.

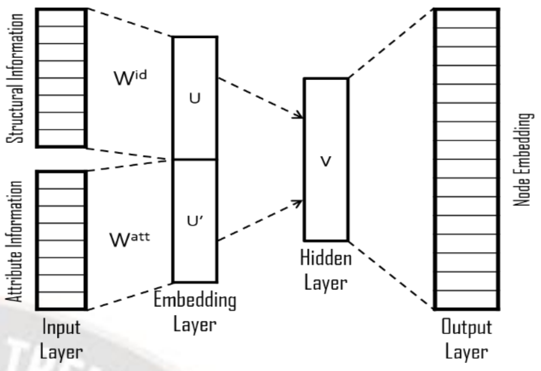

Figure 3. Network Encoder Component of CDA

The input layer is just the early fusion of the graph's structure and attribute modelling portions. It combines the structural as well as the attribute information from the input graph. The attribute information is preprocessed into an array of numeric types. This contains all processed feature information from different types of attributes like binary, numeric, categorical, discrete, continuous, etc. The structural information is given as a one-hot representation of the particular node, where each node h_i is represented using a m-dimensional array where only the *i*-th value is 1 and m denotes the node numbers in the network. Hence, for each node, the inputs in the input layer are a feature vector *f* and the one-hot encoded vector *h*.

The embedding layer encodes structural information and attributes information into compact vectors. It comprises of two elements. The first one converts the one-hot encoded vector *h* containing the node data into a dense vector *u* to capture the structural information of the input. The second part encodes the feature vector *f* into a compact vector *u'*. The weight matrix W^{id}, W^{at} corresponds to that of the structural and feature input respectively. Hence, the two output components of embedding layer *u* and *u'* are calculated using the given formula:

$$u = \sum_{m=1}^{M} h_m d_m \tag{1}$$

where h_m is the mth entry of the one-hot representation, d_m is the mth column of the weight matrix W^{id} and M is the node count in the graph as portrayed in equation (1).

$$u' = \sum_{k=1}^{K} f_k e_k \tag{2}$$

Where in equation (2), f_k is the kth entry of the feature vector, e_k is the kth column of the weight matrix W^{att} and K is the number of feature entries.

The HL is a single-layer perceptron which combines the two components of the embedding layer. The output of the HL *v* is an abstract representation of each node, given as follows:

$$v = \begin{bmatrix} u \\ \lambda u' \end{bmatrix} \tag{3}$$

In equation (3), where the parameter λ which denotes the associative significance of the attributes concerning the structure, *u* denotes the structural component of the embedding layer output and u' denotes the attribute component of the embedding layer output.

The output layer converts the HL output v into a probability vector o that comprises each node's predicted connection capacity to all nodes in its neighbourhood. This is given by:

$$o = [p(u_1, u_i), p(u_2, u_i), p(u_3, u_i), \ldots, p(u_M, u_i)] \quad (4)$$

$$p(u_j, u_i) = \frac{exp(\tilde{u}_j.v)}{\sum_{j'=1}^{M} exp(\tilde{u}_{j'}.v)} \quad (5)$$

In equations (4) and (5), $\tilde{u}_j$ refers to u_j's embedding as a neighbour from the HL output and i is the node under consideration. The softmax probability is used to measure the structural proximity of the node concerning all other nodes.

This encoder is then trained to optimize the likelihood concerning the overall parameters, across all nodes using a softmax scheme and Adam optimizer. The output of this component of the model is an embedding matrix of 128 output features per node.

B. *Graph-based Clustering*

The technique for clustering searches the graph for highly linked components or communities using the supplied network information. The communities are discovered in such a way that the edge numbers connecting vertices inside a community is crucially bigger than the edge numbers connecting vertices across communities. Louvain algorithm [4] is a clustering technique applied to form a coarsened graph.

Algorithm 1. Louvain Algorithm

1. **Initialization**: Each discrete node present in the network is initially allocated to its community.
2. **Modularity Optimization:** The algorithm iteratively optimizes the modularity score by moving nodes between communities. Specifically, it performs the following steps:
   1) For each node, the algorithm evaluates the modularity gain resulting from moving it to each neighbouring community.
   2) The node is then moved to the community that yields the maximum modularity gain, if any.
   3) Steps (a) and (b) are repeated for all nodes in the network.
   4) If no node can be moved to a different community without decreasing the modularity score, the algorithm terminates and the current community assignments are returned as the final result.
3. **Community Aggregation:** In the final step, the algorithm aggregates the communities obtained in step 2 to form a new network. Each community is depicted by a single node, and edge's sum of the weights between the original communities determines the edge weights amongst the new nodes.
4. **Repeating Steps 2 and 3:** This is accomplished repeatedly till no additional enhancement in the modularity score can be accomplished.

Louvain is an unsupervised technique that uses modularity optimization to extract communities from networks. Modularity [21][28] is a network structural metric used to compute the strength of a network's separation into modules. High modularity networks exhibit strong and sparse connections between nodes in similar and diverse modules. In a two-phase iterative method, Louvain optimizes the modularity of a graph. It begins the initial stage by allocating a diverse community to every node in the graph. The method then examines the network's modularity change for each node when it is removed from its original community and placed into the community of its neighbouring node. This process is continued until there is no further rise in modularity and the local maximum is reached. In the second phase, a new graph is generated by combining all nodes in the same community into a single node that represents the community, with edges within the community replaced by self-loops to the node and edges outside the community replaced by weighted connections to other nodes. Once the new graph is constructed, the first phase is repeated on the new graph. This is illustrated in Figure 4.

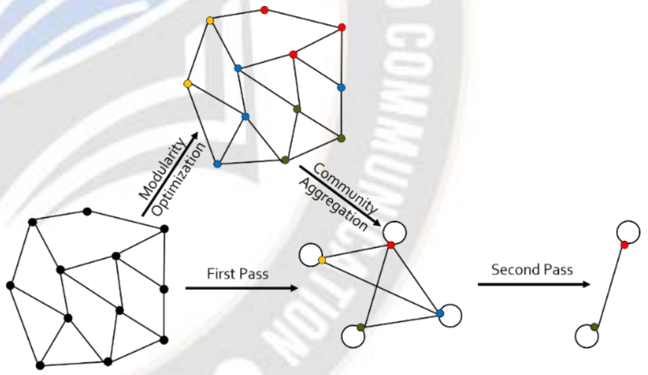

Figure 4. Louvain Algorithm for Graph-based Node Clustering

After collecting the final membership list for each community, the super node embeddings are constructed by simple aggregation functions on each output feature utilizing this community membership information and the resultant node embeddings. These coarsened graph embeddings are fed into the model's third component, the autoencoder.

C. *Autoencoder*

The autoencoder is designed as a dense autoencoder with nine dense layers that are fully connected and each layer has a ReLU activation function as shown in Figure 5. The model is given training by utilizing an Adam optimizer, using the mean absolute error as the loss function. The train-to-test split ratio is 7:3, and test data is also used for validation. Additionally, the data is normalized before being sent into the autoencoder. The autoencoder is run for 50 epochs consisting a batch size of 512. The data is normalized before being passed through the autoencoder for testing.

Table 1 mentions the specifications of the encoder. The reconstruction error is the preferred measure for distinguishing between anomalous and normal supernode embeddings. It is determined as the variance between the input data and the autoencoder output. The threshold for it is the sum of the standard deviation and mean of the loss function. All super nodes with predictions that exceed this threshold are marked as abnormal. The discovered anomalous super nodes, i.e., the anomalous subgraph, and the nodes beneath each subgraph, are therefore the output of the whole model.

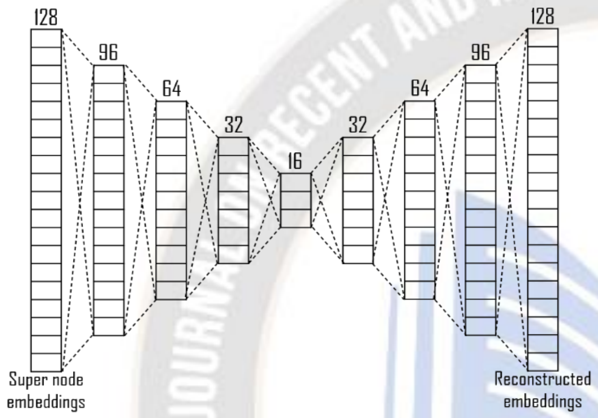

Figure 5. Autoencoder Architecture

TABLE I. AUTO ENCODER SPECIFICATIONS

Model	Sequential
Layer	Dense
No. of Layers	9
Activation Function	ReLU
Optimizer	Adam
Loss Function	Mean Absolute Error
Train-to-Test Split Ratio	7:3
Epochs	50
Batch Size	512

Hence, the anomaly is defined as all those super nodes or subgraphs that cross the reconstruction error threshold. It can hence be mathematically defined as:

$$Anomaly = \begin{cases} True, & r \geq \gamma \\ False, & r < \gamma \end{cases}$$

Where r is the reconstruction error and γ is the threshold values defined as:

$$\gamma = \Sigma(Loss) + \sigma\,(Loss)$$
$$Loss = \Delta(Output, Input)$$

IV. RESULTS AND DISCUSSION

The proposed method is tested using seven publicly accessible open-domain datasets from the SNAP library (Stanford Large Network Dataset Collection). All of the datasets chosen are undirected, non-temporal, and have node properties, as shown in Table 2.

Due to the scarcity of datasets containing actual anomalies, anomalies are synthesized and injected into the aforementioned datasets while adhering to the concept of an anomaly for this study. In this approach, any super node that surpasses the reconstruction error is deemed unusual. As a result, anomalies are created that exceed the given threshold. Nodes are therefore chosen randomly from the original datasets, and the attribute values are adjusted to differ greatly from the original values, implying a high likelihood of being an anomalous unit. This adjusted dataset is fed into the model to evaluate its performance. Because the changed values differ greatly from the original values, they are predicted to exceed the reconstruction error threshold and hence be identified as abnormal.

TABLE II. DATASET DESCRIPTION

Dataset	Nodes	Edges	Description
Github	37700	289003	The social network of GitHub developers.
Giraffe	982	9952	Wikipedia page-page network on Giraffes.
Chameleon	1655	25390	Wikipedia page-page network on Chameleons.
Facebook	22470	171002	Facebook page-page network with page names.
Twitch	6549	112666	The social network of Twitch users.
Deezer	28281	92752	The social network of Deezer users from Europe.
LastFM	7624	27806	The social network of LastFM users from Asia.

The final model was tested on all datasets, each with different anomaly levels, that is, 10, 15, 20, 25, and 30 percent of the original dataset size were made anomalous. The results are observed alongside various performance measures such as Precision, F1 Score, Recall, Accuracy, etc. The same is represented in Table 3.

In Table 3, different percentages of anomalies were injected into each dataset, and the model's performance was monitored in each case. Precision represents how much of the discovered anomalies are genuine anomalies, whereas Recall denotes how many of the genuine abnormalities were detected by the model. As seen in the table, this is known as the True Positive Rate (TPR). Because the goal is to maximise the number of anomalies found, recall is a greater weightage measure because a few normal nodes being marked as anomalous have less influence in applications than true abnormal nodes staying unnoticed. The False Negative Rate (FNR) or miss rate reflects the number of abnormalities that the model does not accurately identify. The model is trained to maximise TPR while minimising FNR.

TABLE III. EXPERIMENTAL RESULTS

Dataset	Anomaly Ratio	Precision	Recall	TPR	FNR	Accuracy
Chameleon	10%	0.5	1	1	0	0.9166667
	15%	0.3333333	1	1	0	0.8333333
	20%	0.6666667	1	1	0	0.9166667
	25%	0.6666667	0.6666667	0.6666667	0.3333333	0.8333333
	30%	1	0.6666667	0.6666667	0.3333333	0.9166667
Deezer	10%	0.4375	0.875	0.875	0.125	0.8863636
	15%	0.6875	0.8461538	0.8461538	0.1538462	0.9204545
	20%	0.6842105	0.7647059	0.7647059	0.2352941	0.8863636
	25%	0.9285714	0.5909091	0.5909091	0.4090909	0.8850575
	30%	0.9375	0.5769231	0.5769231	0.4230769	0.8636364
Giraffe	10%	0.5	1	1	0	0.9411765
	15%	0.5	1	1	0	0.8823529
	20%	0.6666667	0.6666667	0.6666667	0.3333333	0.8823529
	25%	1	0.5	0.5	0.5	0.8823529
	30%	1	0.6	0.6	0.4	0.8823529
LastFM	10%	0.3333333	1	1	0	0.862069
	15%	0.6666667	1	1	0	0.9310345
	20%	0.8333333	1	1	0	0.9655172
	25%	1	0.7142857	0.7142857	0.2857143	0.9310345
	30%	1	0.75	0.75	0.25	0.9310345
Twitch	10%	0	UNDEF	UNDEF	UNDEF	0.875
	15%	0.5	1	1	0	0.875
	20%	0.5	1	1	0	0.875
	25%	1	0.5	0.5	0.5	0.875
	30%	1	0.5	0.5	0.5	0.875
Facebook	10%	0.5454545	1	1	0	0.9230769
	15%	0.75	1	1	0	0.9538462
	20%	1	0.9230769	0.9230769	0.0769231	0.9846154
	25%	1	0.75	0.75	0.25	0.9384615
	30%	1	0.6315789	0.6315789	0.3684211	0.8923077
Github	10%	0.3333333	0.6666667	0.6666667	0.3333333	0.8611111
	15%	0.5	0.8	0.8	0.2	0.8611111
	20%	0.7142857	0.7142857	0.7142857	0.2857143	0.8888889
	25%	0.7142857	0.5555556	0.5555556	0.4444444	0.8333333
	30%	0.8333333	0.5	0.5	0.5	0.8333333

To observe the techniques accomplishment and its variation with diverse datasets, True and False Positive Rate, Accuracy, Precision, and Recall are plotted for the model against each dataset as shown below. Each of these values is measured when the datasets had 0%, 5%, 10%, 15%, 20%, 25%, 30%, 35%, 40%, 45% and 50% of the data as anomalous.

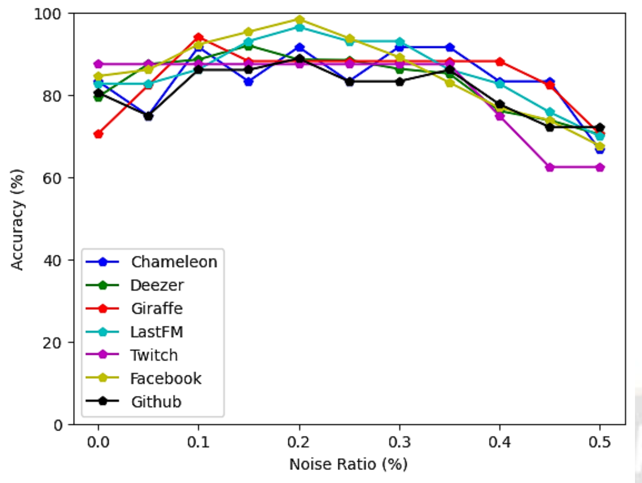

Figure 6. Accuracy Against Noise Ratio for all Datasets

Figure 6 depicts the variation of accuracy against various levels of noise ratio for all datasets. From the plot, it can be observed that the accuracy is not largely affected by the anomaly percentage. However, there is a slight in the accuracy when the anomaly ratio is the highest. This could be due to the imbalance in the ratio of anomalous and non-anomalous nodes.

Figure 7 depicts the variation of precision against various levels of noise ratio for all datasets. From the plot, it can be noted that the precision surges with an increase in anomaly percentage. This is possible because the count of anomalies that goes undetected when the total anomaly density is small, which is largely visible compared to the count of anomalies that go undetected. Except for the Twitch dataset, all other datasets have steady progress. Twitch shows fluctuation due to the small dataset size.

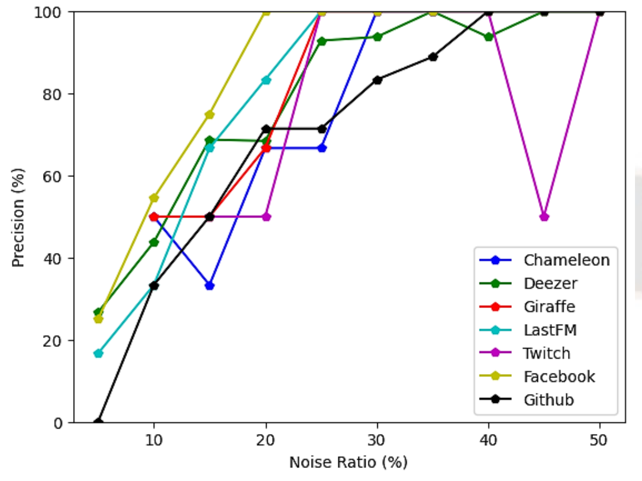

Figure 7. Precision Against Noise Ratio for all Datasets

Figure 8 depicts the variation of recall against various levels of noise ratio for all datasets. From the plot, it can be observed that the recall decreases with an increase in anomaly percentage. This is however acceptable as such high anomaly density is very unlikely in real-world datasets. Nevertheless, the accuracy remains pretty much the same and does not show a lot of fluctuation with changes in anomaly ratio. This shows the stability of the model.

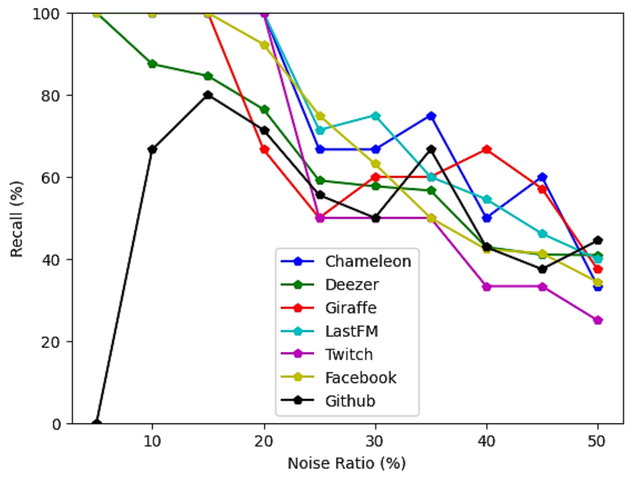

Figure 8. Recall Against Noise Ratio for all Datasets

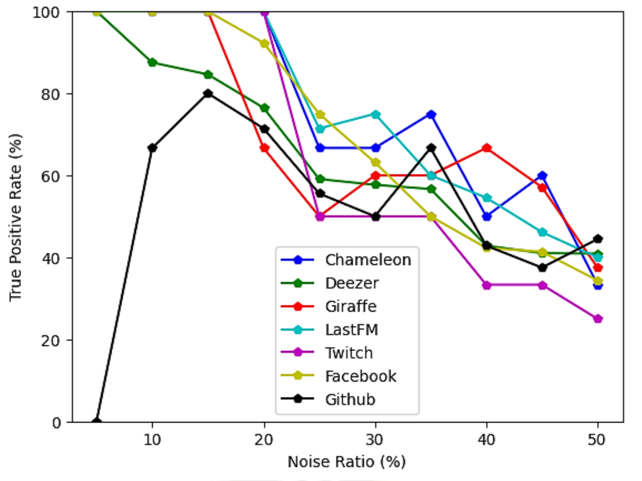

Figure 9. TPR Against Noise Ratio for all Datasets

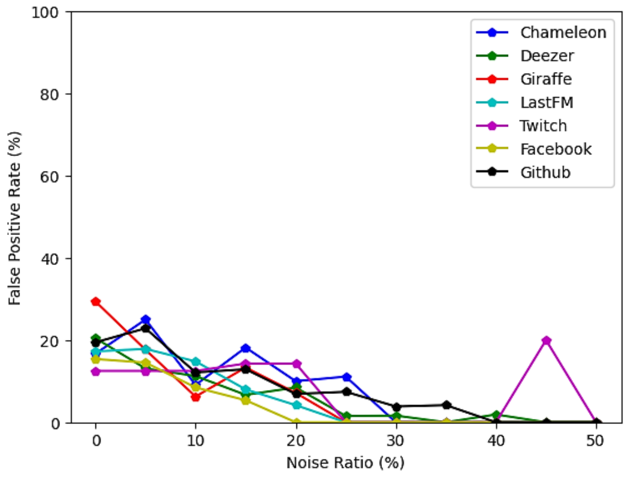

Figure 10. FPR against Noise Ratio for all Datasets

***Article Received:** 05 July 2023 **Revised:** 25 August 2023 **Accepted:** 15 September 2023*

Figure 9 depicts the variation of the true positive rate against various levels of noise ratio for all datasets. From the plot, it can be observed that the true positive rate decreases with an increase in anomaly percentage which is possibly because the varieties of anomalies increase with an increase in anomaly density and becomes hard for the model to detect.

Figure 10 depicts the variation of the false positive rate against various levels of noise ratio for all datasets. From the plot, it can be observed that the false positive rate decreases with an increase in anomaly percentage.

Table 3 displays the model's findings on several datasets with varying anomaly densities. The model's average accuracy across all datasets and noise levels is 89.41%. With more datasets and more training time, this accuracy might increase. A variety of factors might influence the model's performance. The type of datasets and attribute information in them may have an impact on the model since skewed information will influence the model during training to be predisposed to a certain type of data that dominates the dataset. Furthermore, because all anomalies have been artificially introduced into the dataset, its performance may decline to some extent because these anomalies are not necessarily equivalent to what true anomalies would be like. Because all information in node attributes is anonymous and without context, anomalies injected are simply dependent on assumptions about the weightage of each feature, which may or may not be the ones utilized by the model to identify anomalies. The model performs admirably with the datasets under consideration. However, there is room for development in the suggested model design, both in terms of performance and the incorporation of dynamic networks.

V. CONCLUSION

This work provides a methodology for detecting subgraph anomalies in social networks using DL techniques, in which the input graph is coarsened to get super nodes using embedding and clustering algorithms, and anomaly detection is conducted using an autoencoder on this coarsened graph. The results of the experiments reveal that the model performed well. At this time, the model is only intended for static attributed graphs and will not operate on dynamic graphs. In future iterations, the model may also be extended to accommodate volatile real-time networks. In addition, the present model takes as input a sparse feature matrix. Because the attribute list of most real-world datasets or networks is far more complicated, further iterations of the model may be constructed to contain characteristics of other sorts, such as categorical, continuous, numerical values, and so on. Along with the structural information, this information may be employed in the clustering component.

There are several difficulties encountered while attempting to solve a subgraph anomaly detection problem. The issue's complexity is increased by the range of graph types and their high dimensionality. The lack of labels or ground truth for anomalies in the dataset makes evaluating performance and training even more challenging. Anomaly detection is now a critical requirement in all social networks since it has become the fundamental and default form of interaction in practically all spheres of society.

REFERENCES

[1] Ang, C., & Lu, M. (2021, December 6). Ranked: The World's Most Popular Social Networks, and Who Owns Them. Visual Capitalist. https://www.visualcapitalist.com/ranked-social-networks-worldwide-by-users/.

[2] Balakrishnan, N., Barnett, V., & Lewis, T. (1995, March). Outliers in Statistical Data. Biometrics, 51(1), 381. https://doi.org/10.2307/2533352.

[3] Bindu, P., &Thilagam, P. S. (2016, June). Mining social networks for anomalies: Methods and challenges. Journal of Network and Computer Applications, 68, 213–229. https://doi.org/10.1016/j.jnca.2016.02.021.

[4] Blondel, V. D., Guillaume, J. L., Lambiotte, R., & Lefebvre, E. (2008, October 9). Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment, 2008(10), P10008. https://doi.org/10.1088/1742-5468/2008/10/p10008.

[5] Bogdanov, P., Faloutsos, C., Mongiovì, M., Papalexakis, E. E., Ranca, R., & Singh, A. K. (2013). NetSpot: Spotting Significant Anomalous Regions on Dynamic Networks. In Proceedings of the 2013 SIAM International Conference on Data Mining. https://doi.org/10.1137/1.9781611972832.4.

[6] Cook, D., & Holder, L. (2000, March). Graph-based data mining. IEEE Intelligent Systems, 15(2), 32–41. https://doi.org/10.1109/5254.850825.

[7] DAPA-V10: Discovery and Analysis of Patterns and Anomalies in Volatile Time-Evolving Networks. (2009).

[8] Davis, M., Liu, W., Miller, P. D., & Redpath, G. (2011). Detecting anomalies in graphs with numeric labels. In Conference on Information and Knowledge Management. https://doi.org/10.1145/2063576.2063749.

[9] Drury, B., Drury, S. M., Rahman, A., & Ullah, I. (2022). A social network of crime: A review of the use of social networks for crime and the detection of crime. Online Social Networks and Media, 30, 100211. https://doi.org/10.1016/j.osnem.2022.100211.

[10] Eberle, W., & Holder, L. (2007). Anomaly detection in data represented as graphs. Intelligent Data Analysis, 11(6), 663–689. https://doi.org/10.3233/ida-2007-11606.

[11] Ester, M., Kriegel, H., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial Databases with Noise. In Knowledge Discovery and Data Mining (pp. 226–231). https://www.aaai.org/Papers/KDD/1996/KDD96-037.pdf.

[12] Grubbs, F. E. (1969, February). Procedures for Detecting Outlying Observations in Samples. Technometrics, 11(1), 1–21. https://doi.org/10.1080/00401706.1969.10490657.

[13] Gupta, M., Mallya, A., Roy, S., Cho, J. H. D., & Han, J. (2014). Local Learning for Mining Outlier Subgraphs from Network Datasets. In SIAM International Conference on Data Mining. https://doi.org/10.1137/1.9781611973440.9.

[14] Keyvanpour, M. R., Moradi, M. H., &Hasanzadeh, F. (2014). Digital Forensics 2.0. In Studies in computational intelligence (pp. 17–46). Springer Nature. https://doi.org/10.1007/978-3-319-05885-6_2.

[15] Liao, L., He, X., Zhang, H., & Chua, T. S. (2018, December 1). Attributed Social Network Embedding. IEEE Transactions on Knowledge and Data Engineering, 30(12), 2257–2270. https://doi.org/10.1109/tkde.2018.2819980.

[16] Liu, Y., & Chawla, S. (2015). Social Media Anomaly Detection. In Knowledge Discovery and Data Mining. https://doi.org/10.1145/2783258.2789990.

[17] Luan, M., Wang, B., Zhao, Y., & Hu, F. (2021). Anomalous Subgraph Detection in Given Expected Degree Networks With Deep Learning. IEEE Access, 9, 60052–60062. https://doi.org/10.1109/access.2021.3073696.

[18] Malla, N., & Poutintsev, F. (2020, April 26). 13 Advantages and Disadvantages of Social Networking Sites. Honest Pros and Cons. https://honestproscons.com/social-networking-advantages-and-disadvantages/.

[19] Miller, B. L., Bliss, N. T., & Wolfe, P. J. (2010). Subgraph Detection Using Eigenvector L1 Norms. In Neural Information Processing Systems (Vol. 23, pp. 1633–1641). https://papers.nips.cc/paper/4044-subgraph-detection-using-eigenvector-l1-norms.pdf.

[20] Newman, M. E. J. (2006, September 11). Finding community structure in networks using the eigenvectors of matrices. Physical Review E, 74(3). https://doi.org/10.1103/physreve.74.036104.

[21] Newman, M., & Girvan, M. (2004). Finding and evaluating community structure in networks. Physical Review E, 69(2). https://doi.org/10.1103/physreve.69.026113.

[22] Rattigan, M. J., & Jensen, D. (2005, December). The case for anomalous link discovery. ACM SIGKDD Explorations Newsletter, 7(2), 41–47. https://doi.org/10.1145/1117454.1117460.

[23] Shao, M., Li, J., Chang, Y., Zhao, J., & Chen, X. (2021). MASA: An efficient framework for anomaly detection in multi-attributed networks. Computers & Security, 102, 102085. https://doi.org/10.1016/j.cose.2020.102085.

[24] Shrivastava, N., Majumder, A., & Rastogi, R. G. (2008). Mining (Social) Network Graphs to Detect Random Link Attacks. In International Conference on Data Engineering. https://doi.org/10.1109/icde.2008.4497457.

[25] Stanford Large Network Dataset Collection. (n.d.). Stanford Large Network Dataset Collection. https://snap.stanford.edu/data/.

[26] Wang, B., Zhao, Y., Hu, F., & Liang, Y. (2018). Anomaly Detection With Subgraph Search and Vertex Classification Preprocessing in Chung-Lu Random Networks. IEEE Transactions on Signal Processing. https://doi.org/10.1109/tsp.2018.2866847.

[27] Wang, H., Zhou, C., Wu, J., Dang, W., Zhu, X., & Wang, J. (2018). Deep Structure Learning for Fraud Detection. In International Conference on Data Mining. https://doi.org/10.1109/icdm.2018.00072.

[28] Wang, Y., & Zhang, Y. (2013). Nonnegative Matrix Factorization: A Comprehensive Review. IEEE Transactions on Knowledge and Data Engineering, 25(6), 1336–1353. https://doi.org/10.1109/tkde.2012.51.

[29] Ye, F., Chen, C., & Zheng, Z. (2018). Deep Autoencoder - like Nonnegative Matrix Factorization for Community Detection. In Conference on Information and Knowledge Management. https://doi.org/10.1145/3269206.3271697.

[30] Yu, R., He, X., & Liu, Y. (2015, October 26). GLAD. ACM Transactions on Knowledge Discovery From Data, 10(2), 1–22. https://doi.org/10.1145/2811268.

[31] Zheng, M., Zhou, C., Wu, J., Pan, S., Shi, J., & Guo, A. L. (2018). FraudNE: a Joint Embedding Approach for Fraud Detection. In International Joint Conference on Neural Network. https://doi.org/10.1109/ijcnn.2018.8489585.