ISSN:1738-6764, eISSN:2093-7504

An Abnormal Pattern Detection Scheme Based on GCN and DBSCAN in a Large-Scale Graph

Christopher Retiti Diop Emane [1], Hyeonbyeong Lee [2], Dojin Choi [3], Jongtae Lim[4], Kyoungsoo Bok[5], and Jaesoo Yoo[6,*]

[1] Chungbuk National University, Korea; Student; retitidiopchristopher@gmail.com
[2] Chungbuk National University, Korea; Student; lhb@chungbuk.ac.kr
[3] Changwon National University, Korea; Professor; dojinchoi@changwon.ac.kr
[4] Chungbuk National University Korea; Professor; jtlim@chungbuk.ac.kr
[5] Wonkwang University Korea; Professor; ksbok@wku.ac.kr
[6] Chungbuk National University Korea; Professor; yjs@chungbuk.ac.kr
* Correspondence

https://doi.org/10.5392/IJoC.2022.18.4.001

Manuscript Received 14 October 2022; Received 20 December 2022; Accepted 20 December 2022

Abstract: *In recent decades, anomaly detection has undoubtedly become one of the most important areas of research. This is because applications such as financial transactions, medical fraud, and anomaly detection can be used to solve a wide range of real-life problems. Data from these applications can be modeled using large graphs of many different nodes and edges. Because of the size and heterogeneity of the data contained in the graph, it is a very difficult task to detect abnormal patterns. In this paper, we proposed a method for detecting abnormal patterns in a large homogeneous graph. The proposed method consisted of two steps. In the first step, the graph was transformed into a vector using a semi-supervised graph neural network (GCN). The second step was based on DBSCAN, an unsupervised clustering method. Various performance evaluations were performed to show the superiority of the proposed method. Experimental results showed that the proposed method could detect abnormal nodes with high accuracy in homogeneous static graphs.*

Keywords: Anomaly Detection; Abnormal Pattern Detection; GCNs; DBSCAN; Large-Scale Graph

1. Introduction

In recent years, graph data structure has been widely used in many fields to represent real-world applications. This structure can be represented statically or dynamically from a fixed sequence of nodes, also called vertices, connected by edges. A static graph can be represented as an ordered pair of nodes and edges that does not change once it is created, while a dynamic graph is a constantly changing structure that responds to the insertion of nodes or edges into the graph. Graph is particularly productive for applications that are best represented in the form of a network, such as social media, finance, and chemistry.

With the growth of communication technologies, the volume of data can reach trillions of useful information that needs to be processed efficiently. Several studies have been conducted on processing information from large graphs, especially with the goal of detecting unusual behavior in the network [1,2]. Depending on how the data is collected, there are many types of anomalies in graphs, such as anomalous nodes, anomalous edges, anomalous subgraphs, and change detection [3,4].

The development of the Internet has made searching and sharing information fast and efficient. As the world grows into a fully interconnected structure, detecting anomalies in large graphs is a major challenge. The purpose of representing data with a graph is to summarize and display the information in each data set, and in particular the key properties, while strictly maintaining the integrity of the representation of the data set. This task can be helpful both in analyzing the data and in communicating the results. Errors or mistakes in huge datasets, resulting in anomalous nodes and patterns in large graphs, can have a significant impact on the expected outcome of a given task. Detection of such anomalous patterns is necessary to ensure a high success rate for the task being performed.

Many approaches have been developed and discovered to identify abnormal behavior in large graphs. The traditional approach to identify abnormal behavior uses the concept of change detection, which consists of finding the normative patterns in the data and then detecting abnormal objects that deviate significantly from the norm [5]. Other methods have been proposed with a feature-based approach [6], which uses graph representation to extract structural graph-centric features for outlier detection. However, these conventional methods omit the graph labeled information which also provides insightful information for anomaly detection. Nowadays, the proposed methods enable the detection of network information using Deep Learning [7], which learns as well as from node features and relationships. Another recent anomaly detection method is to compute an anomaly score for each node and consider the nodes with the highest anomaly score as abnormal, while normal nodes have the lowest anomaly score [8].

To solve the problems of the existing methods, in this paper, we propose a method based on GCN [9] and DBSCAN [10] for detecting abnormal patterns in homogeneous static large-scale graphs. In the proposed method, we first convert the graph structure into a two-dimensional space using Graph Convolutional Networks, which preserve the structural information and properties of the graph to obtain a node representation of the graph. Then, we use a DBSCAN clustering algorithm to determine anomalous nodes based on the obtained node representation.

This paper is organized as follows. Some of the existing methods on our research topic are discussed in Section 2, while Section 3 presents our proposed anomaly detection method. In Section 4 , we give an evaluation of our method from a practical and technical point of view. Finally, in Section 5, we summarize this work with a conclusion.

2. Related Works

Wang et al. proposed a framework called FdGars [3] to detect fraudsters in a large-scale online application review system. This system combines textual, behavioral, and relational features of reviewers to solve the cheater detection problem. To this end, the system extracted content and behavioral features for each reviewer and created a graph by exploiting the relational nature of scammers. They proposed a GCN method based on the limited number of labeled reviewers, typing the reviewers by a predefined labeling method, and detecting more impostors than unlabeled reviewers. The information about the reviewers is extracted from the logs to create a graph structure G that represents the reviewers as nodes, and the reviewers are connected if they reviewed the same application. A fraud measurement method is then used to classify reviewers as normal users or not by comparing two behavioral characteristics of reviewers with their respective thresholds. Finally, they trained the model to learn from the node features and graph structure according to G and a small number of labeled reviewers. After training, the learned model can detect more impostors that have similar behaviors to the labeled impostors among the unlabeled reviewers.

V. Sirisha et al. proposed a technique for detecting suspicious patterns using a graph-based anomaly detection (GBAD) approach [5]. The system can find both normative and anomalous patterns by implementing three heuristic algorithms such as GBAD-MDL, GBAD-P and GBAD-MPS. These algorithms first use the minimum description length (MDL) to find the best substructure in the graph that minimizes a given objective function. However, after discovering the normative pattern, each algorithm has its own anomaly detection method. The algorithm GBAD-MDL then searches for all substructures that are similar to the best substructure within a certain threshold, while the algorithm GBAD-P searches for all extensions of the normative substructure, extracting subgraphs with extensions that have a lower probability, and the algorithm GBAD-MPS examines all subgraphs that lack edges and nodes within a certain change threshold.

Kavehzadeh et al. proposed a deep learning approach for anomaly detection in node-based networks. Since the structural features of each node in [8] are not known, it is difficult to efficiently detect structurally anomalous nodes. This method applies the node2vec algorithm to an input graph to learn a continuous feature representation of each node. The extracted structural features and attribute vectors of each node are combined to obtain the input embedding used for anomaly detection. The proposed system also uses Variational Auto Encoder (VAE), a neural network architecture for dimensionality reduction. The general idea of VAE in Deep2vec is to use an encoder to encode the input into a latent space and a decoder to reconstruct the input. They detect anomalous nodes by ranking them based on losses. The nodes with high reconstruction errors are considered as anomalies.

Traditional methods for detecting anomalies in large graphs consist of finding a normative pattern and interpreting the anomalous pattern based on the norm. Therefore, these methods do not take advantage of the useful information contained in the nodes. The development of artificial intelligence has led to the proposal of methods that are quite efficient in obtaining the structure of the graph and capturing its features. However, the accuracy of these methods is questionable since they are based on the classification of anomalies and the interpretation of probabilities.

3. The Proposed Anomaly Detection Method

3.1 Overall structure

Existing techniques for detecting anomalies in graph data structures use a method of representing the dependencies between the nodes of the network. Although these methods require less search time and cost, there is the problem of low accuracy and reliability. In this paper, we use not only the relationships between nodes, but also the useful information they contain. A node embedding method is required to consider these two pieces of information. Node embedding refers to a low-dimensional representation of each node in a vector space [11].

Figure 1 shows the overall structure of the system. To detect anomalies in large graphs, the proposed method consists of two components such as node level embedding and anomaly detection. For node-level embedding, the proposed method uses GCN, which encodes the graph into a low-dimensional representation of the network. GCN obtains the adjacency matrix of the graph and the feature vector, a k-dimensional vector representing each node. Our method trains the model through two convolutional layers and transforms the graph into a two-dimensional vector. The proposed method performs the anomaly detection task using DBSCAN. DBSCAN is a simple algorithm for defining clusters by estimating the local density. By taking advantage of the transformed vector, this algorithm identifies dense vertex regions that we consider normal data, while the rest of the data is anomalous. However, DBSCAN requires two important parameters to be carefully set in order to efficiently detect the anomalies. In this paper, we optimize the two parameters of DBSCAN to detect the densest cluster and some smaller clusters in the data. The densest cluster detected by DBSCAN contains all normal nodes and the smallest cluster contains all anomalies.

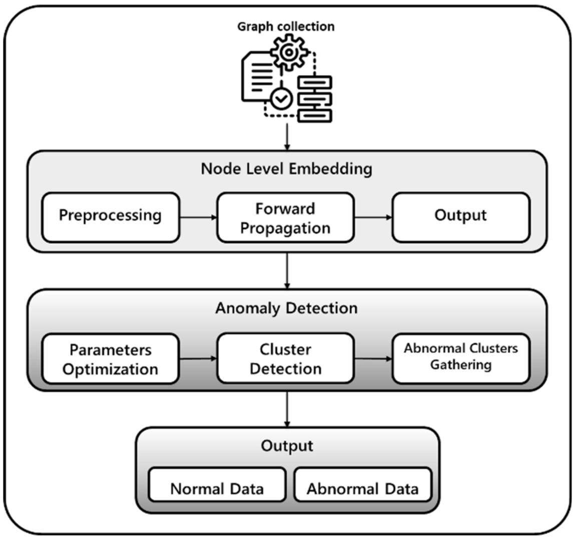
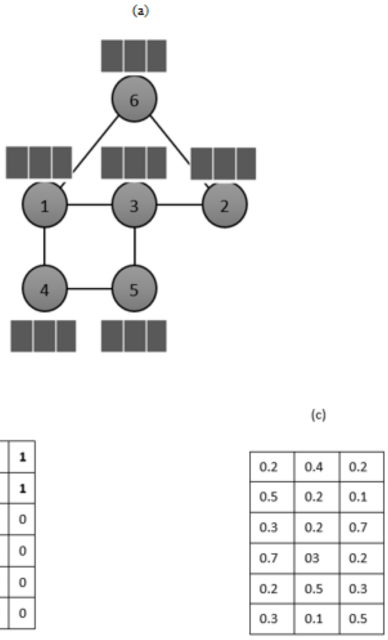


Figure 1. Overall structure of the proposed anomaly detection system

3.2. Node level embedding

A graph is a nonlinear data structure consisting of vertices and edges. Let G = {N, E, F} be a static graph, where N = $\{n_1, n_2, n_3, ...\}$ is the vertex set, E = $\{e_1, e_2, e_3, ...\}$ is the edge set, and F = $\{f_1, f_2, f_3, ...\}$ is the attribute set of each node. The node level embedding transforms the graph into a low-dimensional space [11-15]. For this purpose, we need to collect valuable information about the graph that we will use to transform the data. This includes the adjacency matrix and the feature matrix of the graph. The adjacency matrix of a graph is a two-dimensional Boolean array of dimensions N*N that contains 1 if there is an edge between two nodes and 0 otherwise, where N is the number of nodes in the graph. The feature matrix X ∈ $\mathbb{R}^{N*F}$, on the other hand, represents an ordered numerical property of each node. Figure 2 shows an example of an adjacency matrix and a feature vector obtained from a small graph.



0	0	1	1	0	1
0	0	1	0	0	1
1	1	0	0	1	0
1	0	0	0	1	0
0	0	1	1	0	0
1	1	0	0	0	0

0.2	0.4	0.2
0.5	0.2	0.1
0.3	0.2	0.7
0.7	03	0.2
0.2	0.5	0.3
0.3	0.1	0.5

Figure 2. An example of extracted data

In this paper, we use Graph Convolutional Networks (GCN) to implement the node-level embedding step. GCN is a neural network based on a graph that takes the feature matrix and the graph structure representation as inputs to learn data [16,17]. Using the preprocessed graph structure and the other information contained in each vector, GCN implements forward propagation in the hidden layers of the network. In this paper, we use two hidden layers to transform the data into a two-dimensional representation. A simple representation of the layer propagation proposed by Kipf et al [9], which is used in the learning process of the embedding component of the hidden layer, is defined by the following equation:

$$H^{i+1} = f(A, H^i, W^i) = \sigma(D^{-1/2}\check{A}D^{-1/2}H^iW^i) \tag{1}$$

For a better understanding of the forward propagation used in this paper, a nomenclature of the abbreviations used in equation (1) is presented in Table 1.

Table 1. The descriptions of all notations used in GCN

Notation	Definition
A	Adjacency matrix
X	Feature matrix
I	Identity matrix
$\check{A}$	Adjacency matrix plus identity matrix
H^i	Hidden feature vector, with H^0 = X
W^i	Trainable weight matrix
$D^{-1/2}$	Inverse square root of the degree matrix
σ	Activation function

The proposed method uses 2 different hidden layers that produce an output on each layer through an activation function shown in equation (1). In this work, the result of the second hidden layer is considered as the output. This result is represented in the form of two-dimensional data.

3.3. Anomaly detection

Anomaly detection refers to the problem of distinguishing normal data from abnormal data. We use a density-based clustering algorithm, called DBSCAN for the anomaly detection. The basic idea of this algorithm is to locate the data in a high-density region of the data space and consider them as belonging to the same cluster [18,19]. Using the two-dimensional representation resulting from the node-level embedding step, we identify all nodes related to the largest cluster. The densest region, which we consider normal data, consists of objects located near a given point. Based on this objective, we define anomalies as outliers and small clusters.

Figure 3 describes the different types of clusters we can find with DBSCAN. The normal data represents the densest cluster found by our clustering method. It regroups all nodes classified as normal by the algorithm. The outlier detection cluster then regroups all nodes isolated from the densest cluster. An outlier is generally defined as a data point that is exceptionally far from other data observations. Since outlier detection is highly dependent on the data set, there are no standardized methods for this task. Therefore, determining whether an observation is an outlier or not is ultimately a subjective task. The most interesting feature of the DBSCAN clustering is its ability to detect outliers. Finally, DBSCAN can detect several small clusters whose points are not outliers and do not belong to the normal cluster. In this work, we consider all nodes in these sub-clusters as abnormal nodes.

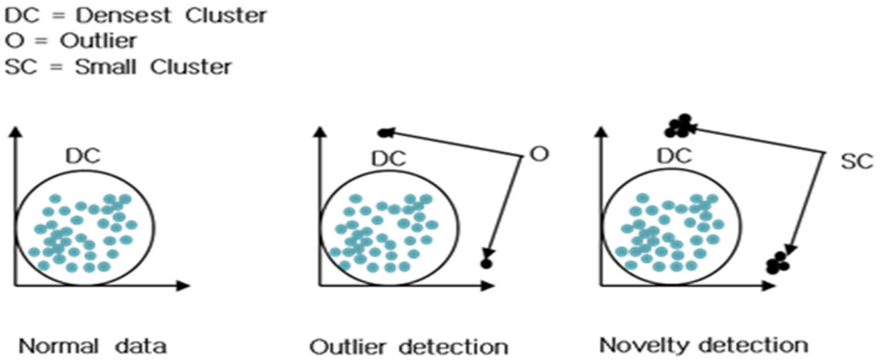


Figure 3: Description of anomaly types

The following equation shows how normal and abnormal data are represented in the system.

$$Cluster_i : \begin{cases} Normal, if \quad densest \quad cluster, \\ Anomaly, otherwise \end{cases} \tag{2}$$

Clustering is a special discipline of machine learning whose goal is to divide our data into homogeneous groups with common features. Now that we have defined the different groups of anomalies we want to detect, the next step is to find the DBSCAN parameters that can allow us to accurately detect anomalies. Of all the DBSCAN parameters, three parameters are very important for anomaly detection: the metric, epsilon (ε), and minimum point (Min_pts). The metric is a hidden parameter that represents the distance function used in calculating the distance between objects. Euclidean distance is the standard metric used in DBSCAN. On the other hand, epsilon and minimum points are two obvious parameters that we need to optimize in order to efficiently detect anomalies in the data and obtain the best result. Epsilon (ε) is the radius of the neighborhood with respect to a given point, while the minimum points parameter indicates the minimum number of points required to form a dense region.

There is no automatic method for determining the minimum number of points required to form a cluster in DBSCAN. In general, MinPts should be greater than or equal to the dimensionality of the dataset. Sander et al. in [20] recommends choosing MinPts = 2*dimensional data when the data has more than 2 dimensions. Ester et al. in [10] also suggests a default value of MinPts is equal to 4 for 2-dimensional data. Since we use 2-

dimensional data in this work, we optimized the Min_pts parameter to 4. However, the value of epsilon must be automatically optimized to improve the result of DBSCAN. Now that we have determined the minimum point value, we can determine the maximum distance between two points that can still belong to the same cluster. In [21], Rahmah and al. proposed a way to determine an optimal epsilon value for the DBSCAN algorithm by calculating the average distance between each point and its smallest nearest neighbors. The average of these distances is then plotted in ascending order on a k-distance plot, and the optimal value for ε is found at the point of maximum curvature. This method gives good results for clustering data.

4. Performance Evaluation

To evaluate the performance of the proposed method, we performed a simulation using the Python programming language on a personal computer with Intel® Core™ i5-4440 CPU @ 3.10GHz, RAM 16.0GB, and Microsoft Windows 11 64-bits operating system. We evaluated the proposed approach using two of the most used graph data such as Amazon-Fraud and Yelp-Fraud for anomaly detection models. The Amazon-Fraud dataset includes product reviews in the musical instrument category, while the Yelp-Fraud dataset consists of hotels and restaurant reviews. Table 2 shows two datasets for the performance evaluation. Each dataset consists of millions of edges and multiple nodes.

Table 2. Dataset of the Performance Evaluation

Dataset	Nodes	Edges	
Amazon-Fraud	11,944	U-P-U	175,608
		U-S-U	3,566,479
		U-V-U	1,036,737
		All	**4,398,392**
Yelp-Fraud	45,954	R-U-R	49,315
		R-T-R	573,616
		R-S-R	3,402,743
		All	**3,846,979**

To get a good result with our method, we need to determine the activation function of the GCN. An activation function is basically just a linear regression model. The choice of activation function in the hidden layers is a crucial part of neural network design that determines how well the network model learns from the dataset. To normalize the output of GCN, we selected four activation functions for nonlinear problems such as SoftMax, Leaky-Relu, Sigmoid, and Tanh. We evaluated the performance of each function according to an optimal value of epsilon for each dataset.

Figure 4 shows the evaluation of each activation function according to the optimal value of epsilon for the YelpChi-fraud dataset. We can see easily that the Leaky-Relu function shows the best performance.

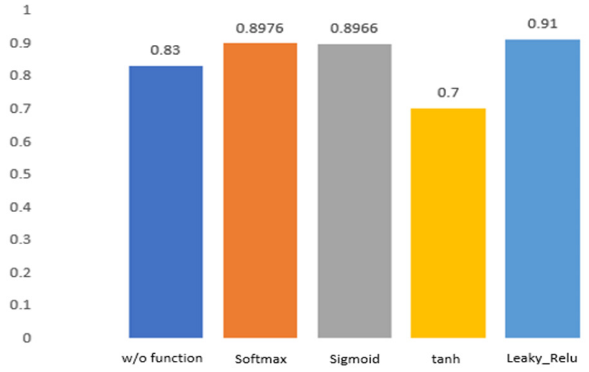


Figure 4. Evaluation for YelpChi-Fraud according to epsilon

Figure 5 shows the evaluation of each activation function according to the optimal value of epsilon for the Amazon-fraud dataset. We can also see easily that the Leaky-Relu function shows the best performance. The performance comparison of our method was performed with K-Means Clustering. K-means clustering is a vector quantization method originally from signal processing that aims to divide n observations into K clusters in which each observation belongs to the cluster with the closest cluster centroids. For using K-means clustering, we apply the output of the node level embedding step. We compared the performance of the proposed method with that of K-means clustering in terms of accuracy for Amazon-Fraud and Yelp-Fraud datasets.

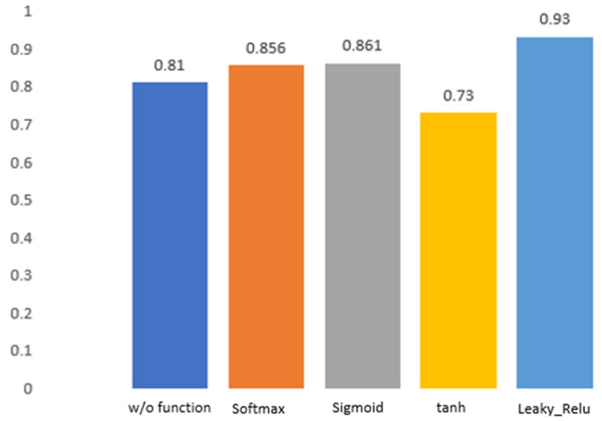


Figure 5. Evaluation for Amazon-Fraud according to epsilon

Figure 6 shows the performance evaluation results on accuracy using each dataset. The proposed method showed 86% and 89% accuracy for Amazon-fraud and YelpChi-fraud, respectively. However, the existing method with node level embedding and K-means clustering method showed 22% and 31% for the two datasets, respectively. As a result, it was shown through the performance evaluation that the proposed method achieved about 60% better accuracy than the existing method on average.

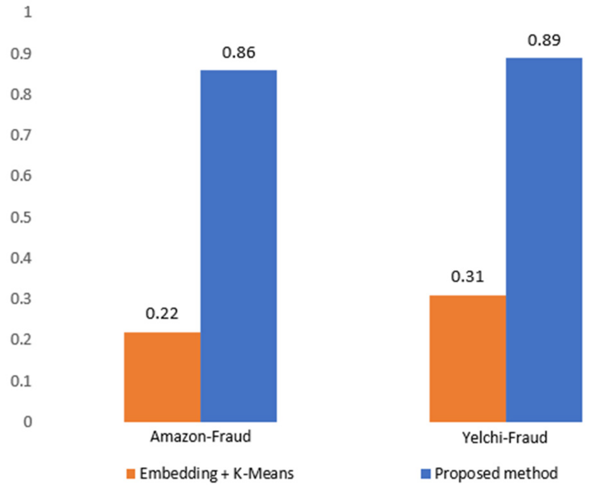


Figure 6. Accuracy rate on each of the data sets

5. Conclusions

In this paper, we have proposed a semi-supervised method for detecting anomalous nodes in static large data. The proposed method first converts a graph into vectors using Graph Convolutional Networks, which preserve the structural information and properties of the graph to obtain a node representation of the graph. Then, we use a DBSCAN clustering algorithm to determine anomalous nodes based on the obtained node representation. In this paper, a node is considered anomalous if it deviates significantly from the norm or belongs to the smallest clusters that are close to the densest cluster. It was shown through various performance evaluations that the proposed method outperforms the existing method in terms of accuracy. Future studies will investigate anomalies on edges and subgraphs in dynamic network environments.

Acknowledgments: This work was supported , by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) ((No. 2022R1A2B5B02002456), by the MSIT (Ministry of Science and ICT), Korea, under the Grand Information Technology Research Center support program (IITP-2022-2020-0-01462) supervised by the IITP (Institute for Information & communications Technology Planning & Evaluation), by the Cooperative Research Program for Agriculture Science and Technology Development (Project No. PJ01624701) Rural Development Administration, Republic of Korea, and by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2021-0-02082, CDM_Cloud: Multi-Cloud Data Protection and Management Platform).

Conflicts of Interest: The authors declare that they have no potential conflicts of interest.

References

[1] G. Malewiczm M. H. Austern, A. J. C. Bik, J. C. Dehnert, I. Horn, and G. Czajkowski, “Pregel: a system for large-scale graph processing,” *In: Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pp. 135-146, 2010. doi: https://doi.org/10.1145/1807167.1807184.

[2] R. F. Erbacher, K. L. Walker, and D. A. Frincke, “Intrusion and misuse detection in large-scale systems,” *IEEE computer graphics and applications*, vol. 22, no 1, pp. 38-47, 2002. doi: https://doi.org/ 10.1109/38.974517.

[3] J. Wang, R. Wen, C. Wu, Y. Huang, and J. Xiong, “Fdgars: Fraudster detection via graph convolutional networks in online app review system," *Companion Proceedings of the 2019 World Wide Web Conference*, 2019. doi: https://doi.org/10.1145/3308560.3316586.

[4] R. Yu, H. Qiu, Z. Wen, C. Y. Lin, and Y. Liu, “A survey on social media anomaly detection,” ACM SIGKDD Explorations Newsletter, vol. 18, no 1, pp. 1-14, 2016. doi: https://doi.org/10.1145/2980765.2980767.

[5] S. Velampalli, L. Mookiah, and W. Eberle “Discovering suspicious patterns using a graph-based approach,” *In: The Thirty-Second International Flairs Conference*, 2019.

[6] L. Akoglu, M. Mcglohon, and C. Faloutsos, “Anomaly detection in large graphs," School of Computer Scuence Carnegie Mellon University Pittsburgh, PA, 2009.

[7] X. Ma, J. Wu, S. Xue, J. Yang, C. Zhou, Q. Z. sheng, H. Xiong, and L. Akoglu, “A comprehensive survey on graph anomaly detection with deep learning,” *IEEE Transactions on Knowledge and Data Engineering*, 2021. doi: https://doi.org/10.1109/TKDE.2021.3118815.

[8] P. Kavehzadeh, M. Samadi, and M. A. Haeri, “Unsupervised Anomaly Detection on Node Attributed Networks: A Deep Learning Approach,” *4th International Conference on Information Science and Systems, ICISS 2021. Association for Computing Machinery (ACM)*, 2021. doi: https://doi.org/10.1145/3459955.3460597.

[9] T. N. Kipf and M. Welling “Semi-supervised classification with graph convolutional networks,” *Conference paper at ICLR 2017*, 2017. doi: https://doi.org/10.48550/arXiv.1609.02907.

[10] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise,” *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pp. 226-231, 1996.

[11] W. Yu, W. Cheng, C. C Aggarwal, K. Zhang, H. Chen, and W. Wang, “Netwalk: A flexible deep embedding approach for anomaly detection in dynamic networks,” *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 2018. doi: https://doi.org/10.1145/3219819.3220024.

[12] E. A. Manzoor, M. Sadegh, V. N. Venkatakrishnan, and L. Akoglu, “Fast memory-efficient anomaly detection in streaming heterogeneous graphs,” *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016. doi: https://doi.org/10.1145/2939672.2939783.

[13] S. Bhatia, M. Wadhwa, K. Kawaguchi, N. Shah, P. S. Yu, and B. Hooi, “Sketch-Based Streaming Anomaly Detection in Dynamic Graphs,” computer science, vol. 2, pp.1-12, 2022. https://doi.org/10.48550/arXiv.2106.04486.

[14] M. Xu, “Understanding graph embedding methods and their applications,” SIAM Review, vol. 63, no. 4, pp. 825-853, 2021. doi: https://doi.org/10.1137/20M1386062.

[15] J. Chen, T. Ma, and C. Xiao, “Fastgcn: fast learning with graph convolutional networks via importance sampling,” *Conference: International Conference on Learning Representations*, 2018. doi: https://doi.org/10.48550/arXiv.1801.10247

[16] science, , 2018, https://doi.org/10.48550/arXiv.1801.10247.

[17] S. Wang and P. S. Yu, “Graph Neural Networks in Anomaly Detection,” *In: Graph Neural Networks: Foundations, Frontiers, and Applications. Springer, Singapore*, pp. 557-578, 2022. doi: 10.1007/978-981-16-6054-2_26.

[18] https://signing.tistory.com/125

[19] D. K. Xu and Y. G. Tian, “A comprehensive survey of clustering algorithms,” Annals of Data Science, vol. 2, pp.165-193, 2015. doi: https://doi.org/10.1007/s40745-015-0040-1.

[20] A. Sreenivasulu, “Evaluation of Cluster Based Anomaly Detection,” Master degree project, University of SKOVDE, 2019.

[21] J. Sander, M. Ester, H. P. Kriegel, and Xiaowei Xu, “Density-based clustering in spatial databases: The algorithm gdbscan and its applications,” Data mining and knowledge discovery, vol. 2, no. 2, pp. 169-194, 1998. https://doi.org/10.1023/A:1009745219419.

[22] N. Rahmah and I. S. Sitanggang, “Determination of optimal epsilon (eps) value on dbscan algorithm to clustering data on peatland hotspots in Sumatra,” In: IOP conference series: earth and environmental science, vol. 31, p. 012012, 2016. https://doi.org/10.1088/1755-1315/31/1/012012.