Leveraging Graph Neural Networks for Botnet Detection

Ahmed Mohamed Saad Emam Saad(✉)

Texas A&M University-Corpus Christi, Corpus Christi, TX 78412, USA
asaad1@islander.tamucc.edu

Abstract. Guarding the cyberinfrastructure is critical to ensure the proper transmission and availability of computer network services, information, and data. The proliferation in the number of cyber attacks launched on the cyberinfrastructure by making data unprocurable and network services inaccessible is on the rise. Botnets are considered one of the most sophisticated cybersecurity threats to the cyberinfrastructure and are becoming more daunting with time. Developing an efficient and robust botnet detection technique is a priority to ensure the security and reachability of the cyberinfrastructure. In this research, we introduce a solution and explore the use of a novel neural network architecture leveraging a graph-based learning approach, namely Graph Neural Network (GNN) for botnet detection. GNN was used to benefit from the unique architecture of botnets and to omit the feature engineering step of the machine learning pipeline as it is a costly and cumbersome process. Additionally, we report the effectiveness of different GNN variations in terms of detecting botnets to get an insight into the performance of each model. The ISCX-Bot-2014 dataset was used to create a graph data object for the training and testing of our proposed approach. The results show our proposed GNN solution's ability to generalize to unseen botnets and perform better compared to other relevant work from the literature with an accuracy that exceeds 94%.

Keywords: Graph neural network · Representation learning · Cyberinfrastructure · Cybersecurity · Machine learning · Botnet detection system

1 Introduction

Cyberinfrastructure is a terminology that refers to every computing system including data warehouses, Internet of things (IoT) systems, and computer network services all linked together to improve research productivity and facilitate breakthroughs [1]. IoT systems and network services have the capacity for cyber vulnerabilities and are prone to cyber-attacks [2]. Distributed denial-of-service (DDoS) attacks pose a damaging threat to the cyberinfrastructure [3]. Botnets are one of the primary sources of DDoS attacks [4].

A. Ortis et al. (Eds.): ICAETA 2023, CCIS 1983, pp. 135–147, 2024.
https://doi.org/10.1007/978-3-031-50920-9_11

A botnet is a terminology that is used to refer to a collection of infected computers or bots, that are controlled by an attacker by using various command and control (C&C) channels. These channels can use various botnet topologies: centralized, distributed (e.g., P2P) or randomized [5]. It can perform a wide variety of malicious activities like DDoS attacks and phishing. In other words, for the protection and guarding of the cyberinfrastructure, there is an immanent need for an effective solution for detecting botnets. Numerous botnet detection approaches were introduced and categorized under four different categories: DNS-based, signature-based, mining-based, and anomaly-based [6].

Each of the previously mentioned approaches has a downside. The DNS-based approach is used to monitor and detect anomalies in DNS query information initiated by bots to receive commands from C&C channels. Similarly, anomaly-based detection techniques inspect network traffic for abnormalities such as high traffic volume and activities on unusual ports. Both approaches face the issue of producing high false positive rate detection results when detecting botnets since they consider any deviation as an anomaly [6]. The signature-based approach works similarly to rule-based systems such as DNS-BD [7], and it lacks the ability to detect unknown botnets since it is limited by the rules within its database. The mining-based technique originally was proposed to use data-driven solutions such as machine learning to detect anomalies in the botnet C&C communication traffic since it is difficult to identify using the other approaches. The downside of the traditional machine learning solution is that it only uses the network traffic feature to identify botnets, ignoring its architecture. In addition, the feature engineering process to set up a machine learning solution with high accuracy and a low false positive rate is a laborious task.

Therefore, creating a novel solution that is able to identify botnets from both architectures and network traffic features contributing to a better performance with high accuracy and within an acceptable false positive rate is crucial. Moreover, the solution should be scalable and able to identify unknown botnets and avoid going through the cumbersome feature engineering process. To this end, we leverage Graph Neural Network (GNN), a novel neural network architecture with a supervised representation learning approach.

In this research, we propose an efficient solution for the botnet detection problem using representation learning combined with a novel neural network architecture, namely Graph Neural Network for the following reasons. First, it can understand and learn the structure or topology of data in form of a graph in addition to its features. Second, it automatically learns features, meaning it does not require feature engineering for the input data as it is a time-consuming and difficult process. Third, it can scale and generalize to previously unseen botnet types. The introduced approach utilizes different variations of the GNN to get an insight into the performance of each model.

Representation learning utilizes the inherent topological structure and information of the graph data object to learn its architecture and features. The problem is formalized as a node-level binary classification problem on a graph data object. We use the ISCX-bot-2014 botnet dataset [8] to build a graph data object

for the learning process. The graph data object is represented as nodes and edges. The nodes resemble each existing IP address in the dataset and the edges resemble the connection between the nodes. Each node is assigned a label depending on its IP address nature. The learning process starts by creating an initial node embedding which will later be updated as the learning process continues. Each node starts automatically learning its features by collecting information from its neighboring nodes in a process called *aggregate*. After that, each node updates its own embedding in a process called *update*. Moreover, each node embedding is trained against the label assigned to the node. Multiple aggregators were utilized to create different instances of the GNN and to get an insight into the performance of each model for botnet detection. The ISCX-Bot-2014 dataset is one of the largest and most diverse in terms of existing botnet types [8]. It is used as a benchmark to assess the performance of botnet detection solutions as it overcomes the three challenges for most of the exiting botnet datasets (i.e., generality, realism, and representatives) [8]. It is the most appropriate choice and will best serve the aim of this research.

The key contributions of this study are as follows:

1. Introducing a novel graph-based botnet detection solution utilizing the botnet's communication network traffic in addition to its architecture. The developed approach excludes the tedious feature engineering process from the machine learning pipeline.
2. Providing a graph data object for the ISCX-Bot-2014 dataset as it adds to the benchmarks used for evaluating GNN approaches for botnet detection solutions.
3. Gaining insight into the performance of multiple GNN models for botnet detection under different variations compared to other solutions from the literature.

The remaining of this paper is organized as follows. Section 2 explores the most relevant related work. A brief background is provided in Sect. 3. The followed methodology is explained in Sect. 4. In Sect. 5, evaluation and benchmarking are discussed. Finally, we conclude in Sect. 6.

2 Related Work

Several botnet detection solutions using machine learning approaches and techniques were developed.

The first attempt to use GNN for botnet detection was proposed by Zhou et al. [9], they formalized their approach as on graph node multi-classification problem. They based their botnet detection solution on the idea of recognizing the topologies of botnets (i.e., centralized and distributed). Their approach ignored the random architecture of botnets. In addition, it is computationally expensive requiring a massive dataset to train on and a large number of neighborhood hops, 12 to be exact. The dataset used for evaluation was a combination of both real and synthetic topologies of background traffic network and botnet

traffic network. The authors claimed that their approach was able to identify previously unseen botnet structures compared to other approaches.

Nguyen et al. [10] proposed an IoT botnet detection solution using a deep learning approach based on printable string information (PSI) graph in addition to the linkage between them and Convolution Neural Network (CNN) as a classifier. They demonstrated the potential of CNN as a malware classifier when trained on PSI graph links and information such as IP address, domain name, etc. Their approach was tested on the IoTPOT dataset achieving an accuracy of 92%.

Chowdhury et al. [11] suggested a botnet detection approach based on recognizing multiple topological features of nodes on a graph. Their detection approach used a clustering method known as a self-organizing map. The authors claim that the proposed approach can limit the search time of botnet nodes by its ability to cluster and isolate botnet nodes in sub-clusters of smaller sizes. They tested their approach on the CTU-13 dataset and reported their results against other classification-based detection solutions.

The work that introduced the ISCX-Bot-2014 dataset used in this research was presented by Beigi et al. [8]. In addition, they developed their botnet detection approach based on classifying network traffic features as malicious or benign. They address the issue of modeling the behavior of a botnet depending solely on its flow-based features. They used a shallow learning algorithm, namely a decision tree as a classifier. The authors used a feature selection algorithm that consisted of 2 key parts: group exclusion and feature inclusion. In other words, they experimented with multiple flow-based feature groups to find the optimal feature groups that yield the highest accuracy and detection rate. Then, they eliminated feature groups that weakly contribute to the accuracy of the model. Furthermore, the features in the least performing groups were analyzed and the features that strongly enhance the accuracy of the model were selected. The authors tested their approach in seven different settings and top using the introduced dataset. The top-performing model achieved an accuracy of 75% and a detection rate of 69%.

Hossain et al. [12] proposed a feature selection algorithm combined with a supervised deep learning classifier, namely an artificial neural network (ANN). Their proposed approach integrated the inclusion-exclusion feature selection process to determine the optimal subset of features for a more accurate flow-based botnet detection solution. They introduced a heuristic algorithm that initially tests the accuracy with all existing features, and the yielded accuracy is the baseline. Moreover, an exclusion iterative process takes place where some features are excluded at each iteration step. Furthermore, the features contributing to a model with a higher botnet detection accuracy rate remain, otherwise excluded. The iterations continue till the accuracy of the remaining features model drops below a certain threshold (i.e., the previous iterations' accuracies). Moreover, their highest-performing model included training on only ten features excluding MissingBytes and AvgPBS features. They tested their approach on the ISCX-Bot-2014 dataset, similar to the work in [8]. The experiments show that their

model for botnet network traffic achieved a detection rate of 91% and an accuracy of 86.41%.

The most relevant works from the literature are [8,12] for the following reasons. First, they used a feature selection algorithm and ignored botnet structures, considering only network traffic features which we challenge with our proposed GNN approach. Second, they trained and tested their approach on the same dataset used in this research. Therefore, these research papers' results will be used later to be compared with our proposed approach.

3 Graph Neural Network

GNN is a novel neural network architecture introduced to deal with data represented in the form of graphs in graph domains [13]. It is considered the natural evolution of the existing neural network architectures. The GNN architecture is designed to work on different types of graphs such as directed, undirected, etc. Graphs G can be represented as nodes or vertices x and connections between these nodes can be represented as edges l. In GNNs, we have three prediction problem levels we can solve: node level, edge level, and graph level [14]. For the node level prediction, we can predict the label or type of each node based on the information provided by its surrounding or neighbor nodes. Moreover, the edge or link prediction level is concerned with predicting the label of the edge based on the information gathered from the two linked nodes. Furthermore, graph level prediction is a problem in which we predict a label for an entire graph, where multiple graphs exist, and each graph represents a class or a label.

Each GNN approach differs in 2 ways: the way we aggregate f_w the messages (i.e., node features in a vector format) from the neighbors and the way we update g_w the self-node embedding. This process is known as message passing, during this process each node gets information from other nodes in its neighborhood as shown in Fig. 1. The output of the message passing process for every single node is called *embedding*. The depth from which the information is collected is called a hop k, meaning if k is set to value '1', the information will be gathered from only the direct neighbors. Moreover, if k is set to value '2' the information gathered is not limited to the direct neighborhood, but it includes the neighbors' neighborhoods. Equations 1 and 2 further explain the aggregation process of node x_1 and the updating process of its self-node embedding o_n to be combined with information received from its direct neighbors (i.e. $k = 1$), respectively. Both equations are notated as follows: f_w represents the aggregation function, g_w represents the update function, l_n is the label of node n, l_{co} is the label of n edges, x_{ne} is the current embeddings of neighborhood nodes n, and l_{ne} is the label of each node n in the neighborhood.

$$\boldsymbol{x}_n = f_{\boldsymbol{w}} \left(\boldsymbol{l}_n, \boldsymbol{l}_{\mathrm{co}[n]}, \boldsymbol{x}_{\mathrm{ne}[n]}, \boldsymbol{l}_{\mathrm{ne}[n]} \right) \tag{1}$$

$$\boldsymbol{o}_n = g_{\boldsymbol{w}} \left(\boldsymbol{x}_n, \boldsymbol{l}_n \right) \tag{2}$$

In addition, different approaches can also include utilizing only node embeddings, only edge embeddings, or both for the message-passing process.

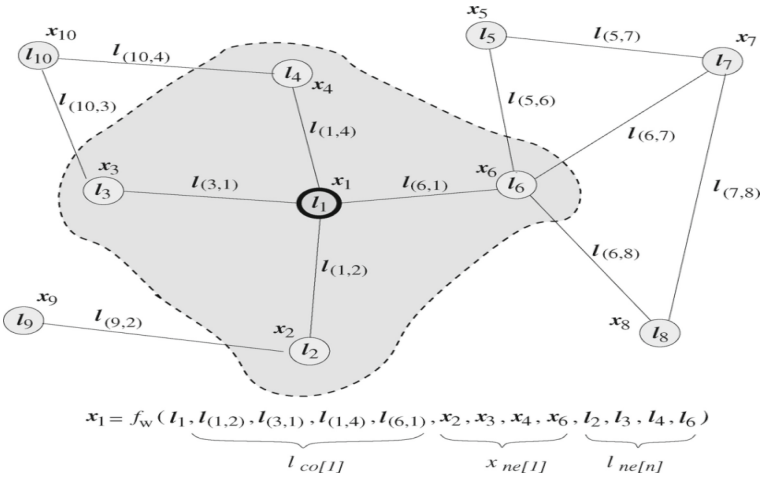

Fig. 1. Graph and the neighborhood of a node [13]

Furthermore, depending on the data, we first structure the data into either a single graph for node and edge level predictions or multiple graphs for graph-level prediction. Moreover, for graph-level prediction, each graph will include its nodes and edges if multiple data objects exist and will be used for classification or clustering.

4 Methodology

This study aims to propose a solution for the botnet detection problem, which utilizes a representation learning algorithm in addition to the topological structure and network traffic presented in the ISCX-Bot-2014 dataset. The problem will be considered a binary classification node-level problem since we attempt to classify whether some nodes (i.e., bots on a graph or network) belong to a botnet. In addition, the omission of the feature engineering process from the machine learning pipeline is considered to provide an efficient end-to-end fashion botnet detection solution. In the developed approach, GNN is used under different variations to enrich the node embeddings in the message-passing process to make the botnet node more recognizable to the classifier from normal nodes. The ISCX-Bot-2014 botnet dataset was selected as it contains many feature groups that make feature engineering a difficult, time-consuming, and computationally expensive process. Therefore, it is used to prove that our approach can perform well without the feature engineering step, hence overcoming the feature engineering process.

4.1 Dataset

The dataset ISCX-Bot-2014 was provided by the Canadian Institute for Cybersecurity, University of New Brunswick [8]. The dataset was obtained from its

official sources and is divided into training and test datasets that include 7 and 16 types of botnets, respectively. The data has unlabeled training and testing sets with a record count of 331,851 and 345,746, respectively. Moreover, each record has 82 features of network traffic.

4.2 Dataset Preprocessing

The data comes in PCAP format, hence there is a need to convert it into a convenient format to be manipulated. CICFlowMeter [15] was used to convert the network traffic from PCAP into a CSV file format. The training and testing dataset were combined in one CSV file to follow the transductive learning approach [16] followed by GNN (i.e., train and test data must be batched as one graph). Each network traffic was labeled based on its IP address status (i.e., malicious or benign) using a labeling function since a list of all malicious IP addresses was provided by [8]. The malicious traffic label was given a value of "1", and the benign traffic was given a value of "0". The dataset contains some features that will not be utilized in the training process such as *('Src IP', 'Src Port', 'Dst IP', 'Dst Port', and 'Label')* and therefore dropped.

4.3 Graph Data Object Construction

Creating a graph data object was a challenging process given that the dataset is not represented as a graph and that we need to batch the training and testing dataset as one graph. Moreover, tailoring a GNN model to work on both network traffic features and structure for botnet detection was another challenge. The construction of the graph data object will follow the convention of PyTorch Geometric [17] since it is the framework used to develop and test the GNN approach. In order to construct a graph data object, the following steps were followed.

We start by constructing two different matrices: a connectivity matrix and nodes feature matrix combined with a labels list to train against. Moreover, we construct other components necessary for the representation learning process (i.e., training and testing masks, data batch, and class count), which all are discussed below.

Connectivity Matrix. The connectivity or adjacency matrix was constructed by using two features that determine the existence of a connection between two nodes, namely source IP and destination IP. After running some analysis on the data, some nodes were observed to be disconnected from the rest of the nodes, which may decrease the training efficiency. Networkx Python library [18] was used to visualize the entire graph to confirm the analysis results of disconnected nodes. The presence of disconnected nodes was confirmed and then removed from the dataset, and some tests were conducted before and after the removal of those nodes. The accuracy of the GNN model increased after removing the disconnected nodes, and the number of records was reduced to 330,133 for training and 337,907 for testing.

Labels List. Labeling the dataset was done in the data preprocessing step, and all the labels were converted into a list for further usage during training.

Node Features. Assigning initial features to nodes was a cumbersome process since the number of nodes is double the number of features provided (i.e., network traffic records or edge features). In addition, several cases were observed in the data. For example:

* Source node is benign, and the destination node is benign.
* Source node is malicious, and the destination node is benign.
* Source node is benign, and the destination node is malicious.
* Source node is malicious, and the destination node is malicious.

The problem was assigning the network traffic features to which node, since assigning the existing features to source nodes will leave the destination nodes featureless. In addition, if the all ones feature vector $x_n = \{1, ..., 1\}$ method [19] was used to initialize the initial nodes' embeddings for the featureless nodes, it may result in low accuracy. This is because the initial features will be the same for benign and malicious destination nodes.

To address the aforementioned problem, four solutions were proposed and tested:

1. Assigning all one constant vectors to all destination nodes and assigning network traffic features to all source nodes (to confirm our theory).
2. Assigning all one constant vectors to all source nodes and assigning network traffic features to all destination nodes.
3. Duplicating source nodes features and assigning them to all destination nodes.
4. * Duplicating the features for both the source and destination node if both source and destination nodes are benign or malicious.
   * Assigning the network traffic features to the source node, copying a random benign node's traffic features, and assigning them to the destination node if the source is malicious and the destination is benign.
   * Assigning the network traffic features to the source node, copying a random malicious node's traffic features, and assigning them to the destination node if the source is benign and the destination is malicious.

Moreover, the dimension of all one constant vectors is set to be equal to the number of network traffic features. Experiments show that the fourth solution resulted in higher accuracy than the others.

Training and Testing Masks. Graph-based learning follows the convention of transductive learning [16] (i.e., the entire dataset including training and testing sets is fed during the training and testing processes as one graph). To separate the training nodes from the testing nodes, we use another concept special to graph-based learning called masking. A mask is a Boolean array indicating which nodes to be used during training, and those are assigned a Boolean value *'True'*.

Otherwise, it is assigned a Boolean value *'False'*. Similarly, during testing, the test nodes are assigned a Boolean value of *'True'*, and other nodes are assigned a Boolean value of *'False'*. During training, we use a training mask in which the nodes in the training dataset are masked in and test nodes are masked out. Furthermore, during testing, we mask in the nodes in the testing dataset, and we mask out the other nodes.

Data Batch. In GNN, a batch indicates which graph each node belongs to. For example, if we have multiple graphs for a graph classification level problem, then multiple nodes can belong to a different graph. To address this, a batch array is constructed by assigning a unique number to all the nodes belonging to the same graph. In our case, we only have one graph for node classification, and the batch array will contain the same number indicating that all the nodes belong to the same graph.

Class Count. The class count is an integer assigned to indicate the number of classes that exist in the data. In the context of the presented problem, the botnet detection problem is a binary node classification problem in which we have only to classify whether a node belongs to a botnet. Moreover, the class count will be an integer of value "2", indicating the two classes (i.e., malicious and benign).

The final representation of the graph data object was visualized using Networkx [18] and can be viewed in Fig. 2.

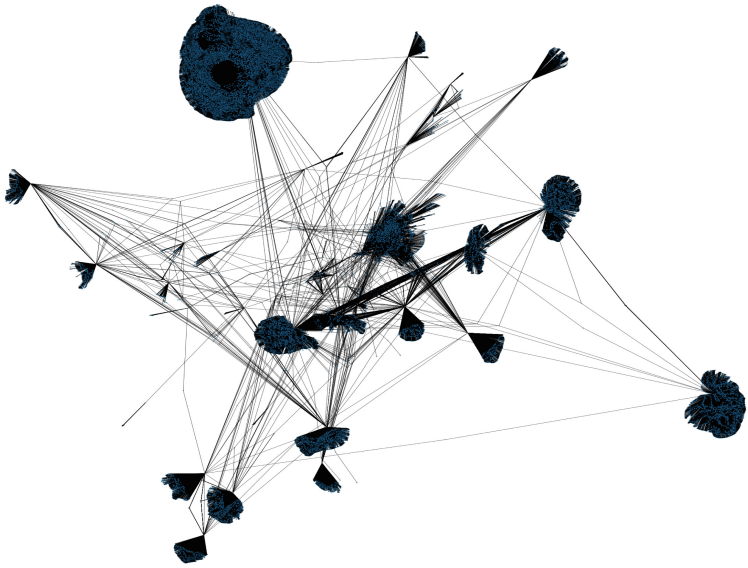

Fig. 2. Graph Representation

5 Evaluation

The method used to evaluate, measure, and describe the performance of the proposed approach is the confusion matrix [20]. It is considered the optimal evaluation method for classifiers. In addition, it shows the actual classification or labels against the predicated ones. Since the developed approach in this study is a binary classifier, the confusion matrix is the selected method for evaluation. The metrics inferred from the confusion matrix include F1 score, accuracy, precision, and recall (detection rate). Those metrics are represented in terms of prediction evaluation categories such as True Positive (TP), False Negative (FN), False Positive (FP), and True Negative (TN). Given the nature of the problem, the evaluation will be heavily reliant on the recall (detection rate) in addition to the accuracy.

5.1 Graph Neural Network Experimental Results

The performance of the proposed GNN approach was examined under four different variations. Each variation utilizes a different operator (i.e., aggregation and update function). Further experiments were conducted, but only top-performing ones are reported. The other experiments were conducted to explore the effect of different operators and neighborhood hop values k on the performance of the proposed approach. In all reported experiments, we used 150 training iterations, a unit size of 136, and a learning rate of 0.001. The top performing experiment is "GNN - Experiment 4" as it scored the highest in every performance metric. As observed, the best results were achieved by using a k value of "2". Moreover, there was no further improvement shown when the k value was increased to "3". Although the training data has only 7 types of botnets as opposed to 16 in the testing set, the results show the GNN model's ability to generalize to unseen botnet types. In addition, the high precision indicates the model's ability to perform with a low false positive rate. An aggregation of the performance metrics of the reported experiments is shown in Table 1.

Table 1. Aggregated Results for the Reported Experiments

Experiment	Operator	k-value	F1 (%)	Recall (%)	Precision (%)	Accuracy (%)
GNN - Experiment 1	GCNConv	2	80.69	80.26	81.12	86.48
GNN - Experiment 2	SAGEConv	2	77.92	73.39	83.05	85.36
GNN - Experiment 3	SGConv	2	81.73	79.17	84.48	87.55
GNN - Experiment 4	TAGConv	2	95.73	93.79	97.75	94.38
GNN - Experiment 5	GCNConv	3	80.75	74.23	88.52	87.54
GNN - Experiment 6	SAGEConv	3	80.43	73.82	88.34	87.36
GNN - Experiment 7	SGConv	3	82.02	76.67	88.18	88.17

5.2 Comparison with Relevant Work

Table 2 is an aggregation of the top-performing GNN model in addition to the performance results of relevant work from the literature. The relevant studies [8,12] have been carried out to find the optimal feature set to improve botnet detection based solely on network traffic features. They evaluated their approach using the same ISCX-Bot-2014 dataset. Both relevant papers suggest a feature selection approach using an inclusion-exclusion algorithm for optimal results. On the other hand, our approach provides an alternative solution based on the topological structure in addition to network traffic features and omits the feature engineering step. A comparison with the relevant studies is shown in Table 2 to emphasize that our approach exceeds relevant solutions in terms of accuracy and detection rate and to show the contribution added by the proposed approach.

Table 2. Comparison with Others' Work

Approach	Detection rate (%)	Accuracy (%)
Our approach (Developed in this research)	93.79	94.38
Group-based Feature selection, decision tree [8]	69	75
Inclusion & exclusion for Feature Selection, feed forward ANN [12]	91	86.41

6 Conclusion

The growth and expansion of the cyberinfrastructure starting from data centers to IoT systems uncovered numerous vulnerabilities in our cyber defense capabilities. Those vulnerabilities can pose a threat to the entire cyber domain. One of the most brutal cyber attacks that threaten the cyberinfrastructure is DDoS. Botnets are considered the origin of most DDoS attacks. The aforementioned problem produced the necessity for a reliable and efficient botnet detection solution. Different botnet detection solutions were developed using mining-based approaches such as machine learning algorithms. However, most of the developed approaches are reliant on detecting botnets using network traffic features and ignoring their topological structure in addition to the tedious task of feature engineering for optimal results.

In this paper, we presented a botnet detection solution by utilizing a representation learning approach, namely graph neural network. We developed a novel approach for botnet detection that utilized the inherent structure of botnets alongside their network traffic features. In addition, we introduced the first graph data object based on the ISCX-Bot-2014 dataset. The aforementioned

dataset was used as it is the most diverse in terms of botnet types. The graph object was used to train, evaluate, and test the effectiveness of the developed GNN approach. In addition, we tested multiple GNN variations to get an insight into the performance of different models for botnet detection. The performance results obtained reflect the dominance of the proposed approach when compared to other related solutions from the literature. The proposed approach outperformed other solutions, achieving an accuracy of more than 94%. Multiple experiments were conducted using different GNN operators and hyper-parameters to obtain the optimal model. The results and settings of the experiments conducted were reported, compared, and discussed.

In future work, we can consider the temporal aspect of the network traffic behavior for botnet detection and integrate it with the proposed approach to add further improvements in the performance.

References

1. Stewart, C.A., Simms, S., Plale, B., Link, M., Hancock, D.Y., Fox, G.C.: What is cyberinfrastructure. In: Proceedings of the 38th Annual ACM SIGUCCS Fall Conference: Navigation and Discovery, pp. 37–44 (2010)
2. Djenna, A., Harous, S., Saidouni, D.E.: Internet of Things meet internet of threats: new concern cyber security issues of critical cyber infrastructure. Appl. Sci. **11**(10), 4580 (2021)
3. Kaur Chahal, J., Bhandari, A., Behal, S.: Distributed denial of service attacks: a threat or challenge. New Rev. Inf. Netw. **24**(1), 31–103 (2019)
4. Hoque, N., Bhattacharyya, D.K., Kalita, J.K.: Botnet in DDoS attacks: trends and challenges. IEEE Commun. Surv. Tutor. **17**(4), 2242–2270 (2015)
5. Abu Rajab, M., Zarfoss, J., Monrose, F., Terzis, A.: A multifaceted approach to understanding the botnet phenomenon. In: Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement, pp. 41–52 (2006)
6. Feily, M., Shahrestani, A., Ramadass, S.: A survey of botnet and botnet detection. In: 2009 Third International Conference on Emerging Security Information, Systems and Technologies, pp. 268–273. IEEE (2009)
7. Alieyan, K., Almomani, A., Anbar, M., Alauthman, M., Abdullah, R., Gupta, B.B.: DNS rule-based schema to botnet detection. Enterp. Inf. Syst. **15**(4), 545–564 (2021)
8. Beigi, E.B., Jazi, H.H., Stakhanova, N., Ghorbani, A.A.: Towards effective feature selection in machine learning-based botnet detection approaches. In: 2014 IEEE Conference on Communications and Network Security, pp. 247–255. IEEE (2014)
9. Zhou, J., Xu, Z., Rush, A.M., Yu, M.: Automating botnet detection with graph neural networks. arXiv preprint arXiv:2003.06344 (2020)
10. Nguyen, H.T., Ngo, Q.D., Le, V.H.: IoT botnet detection approach based on PSI graph and DGCNN classifier. In: 2018 IEEE International Conference on Information Communication and Signal Processing (ICICSP), pp. 118–122. IEEE (2018)
11. Chowdhury, S., et al.: Botnet detection using graph-based feature clustering. J. Big Data **4**(1), 1–23 (2017). https://doi.org/10.1186/s40537-017-0074-7
12. Hossain, M.I., Eshrak, S., Auvik, M.J., Nasim, S.F., Rab, R., Rahman, A.: Efficient feature selection for detecting botnets based on network traffic and behavior analysis. In: 7th International Conference on Networking, Systems and Security, pp. 56–62 (2020)

13. Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.: The graph neural network model. IEEE Trans. Neural Netw. **20**(1), 61–80 (2008)
14. Zhou, J., et al.: Graph neural networks: a review of methods and applications. AI Open **1**, 57–81 (2020)
15. Draper-Gil, G., Lashkari, A.H., Mamun, M.S.I., Ghorbani, A.A.: Characterization of encrypted and VPN traffic using time-related. In: Proceedings of the 2nd International Conference on Information Systems Security and Privacy (ICISSP), pp. 407–414 (2016)
16. Rossi, A., Tiezzi, M., Dimitri, G.M., Bianchini, M., Maggini, M., Scarselli, F.: Inductive–transductive learning with graph neural networks. In: Pancioni, L., Schwenker, F., Trentin, E. (eds.) ANNPR 2018. LNCS (LNAI), vol. 11081, pp. 201–212. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99978-4_16
17. Fey, M., Lenssen, J.E.: Fast graph representation learning with PyTorch Geometric. In: ICLR Workshop on Representation Learning on Graphs and Manifolds (2019)
18. Hagberg, A., Swart, P., Chult, D.S.: Exploring network structure, dynamics, and function using NetworkX. Technical report, Los Alamos National Lab. (LANL), Los Alamos, NM, United States (2008)
19. Lo, W.W., Layeghy, S., Sarhan, M., Gallagher, M., Portmann, M.: E-GraphSAGE: a graph neural network based intrusion detection system for IoT. In: NOMS 2022–2022 IEEE/IFIP Network Operations and Management Symposium, pp. 1–9. IEEE (2022)
20. Vihinen, M.: How to evaluate performance of prediction methods? Measures and their interpretation in variation effect analysis. BMC Genomics **13**, 1–10 (2012)