International Journal on Applications in Engineering and Technology
Volume 1 : Issue 9 : September 2015, pp 7 –14 www.aetsjournal.com
ISSN (Online) : 2455 - 0523

Fortifying Network Security: Resilient Graph Neural Networks for Intrusion Detection

Mr. S. Poorna Prakash,
Assistant Professor,Dept of Computer Science and Engineering
Rathinam Technical Campus,
Coimbatore,India
prakash.cse@rathinam.ine

KATHIRVEL M
Dept of Computer Science and Engineering
Rathinam Technical Campus,
Coimbatore,India
kathirvelm.bcs21@rathinam.in

Mr. A. Suresh kumar,
Assistant Professor,Dept of Computer Science and Engineering
Rathinam Technical Campus,
Coimbatore,India
suresh.cse@rathinam.in

NITHARSAN M
Dept of Computer Science and Engineering
Rathinam Technical Campus,
Coimbatore,India
nitharsanm.bcs21@rathinam.in

Abstract - **A network intrusion is any illegal activity carried out on a digital network.Network intrusion detection is an essential task to maintain the security and privacy of network systems. Conventional intrusion detection systems (IDS) relyon handcrafted features or signatures, which limit their adaptability to new and evolving attacks. In recent years, graph neural networks (GNNs) have emerged as a promising approach for network intrusion detection. GNNs can effectively capture the complex relationships among network entities and learn from the network structure to detect anomalous behaviours. propose a GNN-based network intrusion detection system that leverages the power of graph representations and deep learning techniques to improve the accuracy and robustness of intrusion detection.**

Index Terms – AES, spatial Database and Data Exploration

I. INTRODUCTION

An intrusion detection system (IDS) is a security tool that monitors network traffic or system activity for malicious or unauthorized behavior. Its primary function is to detect and alert security personnel to potential security threats, such as attempted unauthorized access, network scans, and other malicious activities. There are two main types of IDS: network-based and host-based. Network-based IDS monitors network traffic and analyses it for signs of malicious activity, while host-based IDS monitors activity on individual computers or servers for signs of unauthorized accessor malicious activity.

An IDS typically uses a combination of signature-based and behaviour-based detection methods. Signature-based detection involves comparing network traffic or system activity to a database of known malicious patterns or signatures. Behaviour based detection involves analyzing the behaviour of network traffic or system activity to identify potential security threats, even if the activity does not match any known signatures.

When an IDS detects a potential security threat, it generates an alert, which can be sent to security personnel for further investigation and response. An IDS can also be configured to take automated actions to block or mitigate the threat, such as blocking network traffic from a specific IP address or quarantining a compromised system.Overall, an IDS is an important tool for maintaining the security of computer networks and systems. It helps organizations detect and respond to potential security threats before they can cause significant damage or disruption.

Network security consists of the policies and practices adopted to prevent and monitor unauthorized access, misuse, modification, or denial of a computer network and network-accessible resources. Networks can be private, such as within a company, and others which might be open to public access and actually part of the knowledge discovery process. Network security is the security provided to a network from unauthorized access and risks. It is the study of network administrators to adopt preventive measures to protect their networks from potential security threats.

Computer networks that are involved in regular transactions and communication within the government, individuals, or business require security. The most common and simple way of protecting a common and simple way of protecting a network resource.

II. RELATEDWORK

I. A GNN-based NIDS that uses graph convolution networks to capture the structural relationships between network events. The proposed model achieved high accuracy and outperformed traditional machine learning approaches. “Graph Neural Networks for Network Intrusion Detection: An Evaluation of Scalability and Effectiveness" by GNN-based NIDS on large-scale network datasets. The authors propose a GNN-based NIDS that uses graph attention networks to capture the structural relationships between network events. The proposed model achieved high accuracy and scalability on large-scale datasets."Deep Learning Based Network Intrusion Detection Using Gated Graph Neural Networks". GNN-based NIDS that uses gated graph neural networks to capture the temporal relationships between network events. The proposed model achieved high accuracy and outperformed traditional machine learning approaches.

II. A CNN (Convolution Neural Network) based network intrusion detection system is a type of machine learning model that is used to detect anomalous behavior in a network. It works by analyzing network traffic data and

III. identifying patterns of behavior that are consistent with known network attacks or abnormal network activity. In a CNN-based network intrusion detection system, the input data is typically represented as a series of network packets, which are then processed by a series of convolution layers. These convolution layers apply a set of filters to the input data, which are designed to identify specific patterns of activity within the network traffic. Once the convolution layers have processed the input data, the output is passed through a series of fully connected layers, which are used to classify the network traffic as either normal or anomalous. This classification is typically based on a set of pre-defined rules or machine learning algorithms that have been trained on a large dataset of known network attacks and normal network activity. One of the key benefits of using a CNN-based network intrusion detection system is that it can learn to detect new types of network attacks that have not been seen before. This is because the model is trained on a large dataset of network traffic data, which allows it to identify patterns and anomalies that may not be immediately apparent to human analysts. Overall, a CNN-based network intrusion detection system can be an effective tool for detecting and preventing network attacks, as it can analyse large volumes of network traffic data in real-time and identify potential threats before they can cause significant damage.

III. PROBLEM STATEMENT

A. **Design a system and Goals**

To overcome the problems faced in Convolution Neural Network based NIDS, the Graph Neural Network Machine learning model is used in this project. Proposed System for GNN-based NIDS using Tensor Flow and IGNITION Data Pre-processing: The first step is to pre-process the network data and convert it into graph-structured data. This involves defining nodes and edges based on network events, such as IP addresses, ports, protocols, and timestamps. The graph can be constructed using tools like NetworkX or igraph.

Graph Embedding: The next step is to embed the graph into a low dimensional vector space that preserves the structural relationships between nodes. This can be achieved using the IGNNITION library, which provides a high-level API for graph embedding using Graph Convolution Networks (GCNs). IGNNITION can be integrated with Tensor Flow, a popular deep learning framework, for efficient computation and training.

IV. B. OBJECTIVES

A novel Intrusion Detection System (IDS) based on Graph Neural Networks (GNNs) that can efficiently and accurately detect network intrusions. The specific objectives of the research include:

1. Developing a GNN-based IDS model: The first objective of the research is to propose a GNN-based IDS model that can effectively capture the complex relationships between network nodes and detect intrusions with high accuracy.

2. Evaluating the performance of the proposed model: The second objective is to evaluate the performance of the proposed GNN-based IDS model using various performance metrics such as accuracy, precision, recall, F1-score, and Area

Under the Curve (AUC) of the Receiver Operating Characteristic (ROC) curve.

3. Comparing the performance of the proposed model with existing IDS techniques: The third objective is to compare the performance of the proposed GNN-based IDS model with existing IDS techniques such as Decision Tree, Support Vector Machine (SVM), Random Forest, and Deep Neural Networks

(DNNs).

4. Investigating the robustness and generalization ability of the proposed model: The fourth objective is to investigate the robustness and generalization ability of the proposed GNN-based IDS model by performing cross-validation and analysing errors.

5. Identifying potential applications and future work: The fifth objective is to identify potential applications of the proposed GNN-based IDS model and suggest future work to improve its performance and extend its capabilities.

Overall, the main objective of the research is to propose a novel IDS system based on GNNs that can accurately and efficiently detect network intrusions and outperform existing IDS techniques.

IV. SYSTEMARCHITECTURE

The embedded graph is then used to train a GNN-based model for intrusion detection using Tensor Flow. The model architecture can be based on different GNN architectures like GCN, GAT, or GIN. The TensorFlow implementation of GNNs allows for efficient computation and training using GPUs, resulting in faster training times and higher performance.Training and Validation: The GNN-based model is trained on labeled data, which includes both normal and malicious network traffic. The model's performance is evaluated using standard metrics like accuracy, precision, recall, and F1-score. The model's hyper parameters can be tuned using techniques like grid search or random search. Testing and Deployment: Once the model is trained and validated, it can be tested on unseen data to evaluate its generalisation performance. The model can be deployed as a real-time intrusion detection system in a network environment. The system should be able to detect and alert on malicious network traffic in real-time.the proposed system for GNN-based NIDS using Tensor Flow and IGNITION involves pre-processing the network data into graph-structured data, embedding the graph into a low-dimensional vector space using IGNNITION, training a GNN-based model for intrusion detection using Tensor Flow, and testing and deploying the model as a real-time intrusion detection system. The system should be able to capture the complex relationships between network events and adapt to new and evolving attacks, resulting in a more accurate and robust intrusion detection system. The integration of Tensor Flow and IGNNITION allows for efficient computation and training, resulting in faster training times and higher performance.

The GNN-based NIDS (Network Intrusion Detection System) proposed in this report uses a host-connection graph to represent the network traffic. This graphhas nodes for each distinct host involved in the traffic and nodes for each flow.

Two edges are created for each flow connecting it to its source host and destination host. This representation allows for the differentiation and relation of features for upstream and downstream traffic and explicitly represents the relations between different flows connected to the same source/destination hosts. The decision to add specific nodes representing each flow was driven by the way GNN models operate, which consider only the hidden states of nodes in input graphs as learnable objects.

The GNN model proposed comprises a non-standard message-passing algorithm that adapts to the needs of the network intrusion detection problem.

The hidden states of nodes representing hosts are simply encoded with all ones in the initial hidden state, while the hidden states of nodes representing flows are initialized with features of different natures, which depend on the monitoring data accessible in the network. The message-passing phase of the GNN considers the heterogeneity of the graph and comprises two learnable message functions: σsf for edges (S → f) and σfd for edges (f → D). An aggregation function is applied to the messages computed on each node, followed by an element-wise mean over the messages to normalize data across the multiple message-passing iterations.

Finally, the hidden states are updated using two different learnable functions, δh and δf, respectively applied to update the hosts' and flows' hidden states.

Graph Convolution Neural Network (GCN): In this module, the researchers propose using a GCN to learn features from the graph data. GCN is a type of neural network that operates on graph data and can learn node embedding that capture the structural information of the graph.

The researchers train the GCN to predict the labels of the nodes in the graph, which represent the device state (normal or anomalous).The Graph Convolution Neural Network (GCN) is a type of neural network that is designed to operate on graph-structured data. It has been shown to be very effective in many graph related tasks, including node classification, link prediction, and graph classification. In the proposed system "Unveiling the potential of Graph Neural Networks for robust Intrusion Detection", the GCN is used for intrusion detection. The GCN takes the graph data constructed in the previous module as input and learns to extract features from it.

The GCN consists of multiple layers of graph convolutions, each of which updates the node features based on the features of its neighboring nodes in the graph. The graph convolution operation is similar to the convolution operation in convolution neural networks (CNNs) but is defined on the graph domain.

The GCN learns to update the node features by aggregating information from its neighboring nodes. The aggregation can be performed using different methods, including mean pooling, max pooling, and attention-based pooling.

The GCN has been shown to be effective in capturing the graph structure and extracting meaningful features from it, making it a powerful tool for intrusion detection on network traffic graphs.

Anomaly Detection: This module involves using the trained GCN to detect anomalous devices in the network. The researchers use the node embedding generated by the GCN to calculate a reconstruction error for each node. An anomalous node will have a higher reconstruction error than a normal node.

They also propose using a threshold-based approach to determine whether a node is anomalous or not. Anomaly detection is the process of identifying patterns in data that deviate from the norm, indicating that they may be outliers or anomalies. In the context of intrusion detection, anomaly detection refers to identifying network traffic or behavior that deviates from the normal or expected behavior of the system.

To detect anomalies in the proposed system, the trained GCN model is used to predict the probability of each node being an anomaly. The threshold for determining whether a node is anomalous or not is set using the receiver operating characteristic (ROC) curve.

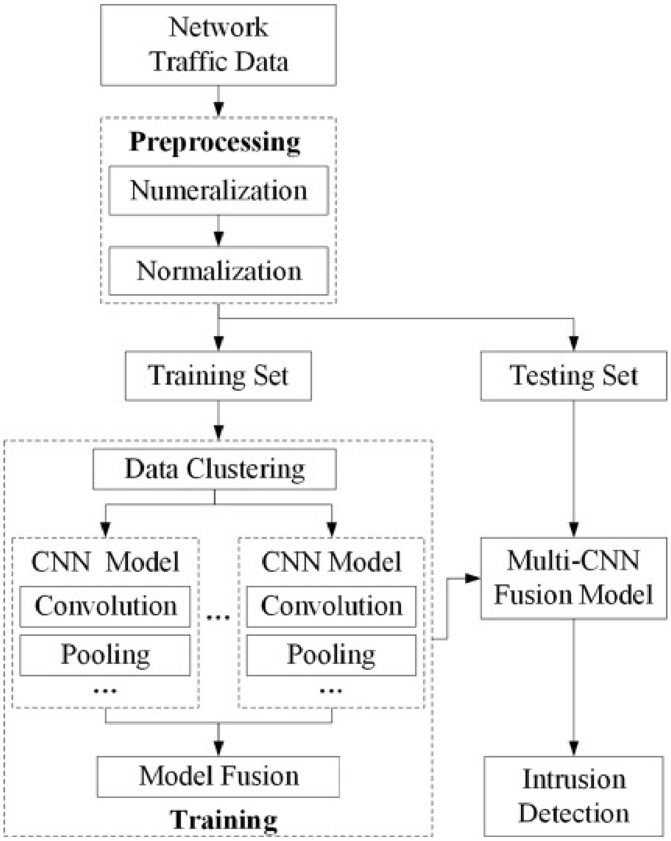


Fig -1: System Architecture

The ROC curve is a plot of the true positive rate (TPR) against the false positive rate (FPR) for different threshold values. The TPR is the proportion of true anomalies that are correctly identified as anomalies, while the FPR is the proportion of non-anomalies that are incorrectly identified as anomalies. The threshold value that maximizes the TPR while minimizing the FPR is chosen as the threshold for anomaly detection. Once the anomalous nodes have been identified, they are evaluated based on various metrics, such as precision, recall, and F1-score. Precision measures the proportion of true anomalies among all nodes identified as anomalies, while recall measures the proportion of true anomalies that are correctly identified by the system.

The F1-score is the harmonic mean of precision and recall and provides a balanced measure of the system's performance. These metrics are used to assess the effectiveness of the proposed system in detecting anomalies.

Evaluation: In this module, the researchers evaluate the performance of the proposed system using several metrics such as precision, recall, F1-score, and area under the receiver operating characteristic curve (AUC-ROC). They compare the performance of their proposed system with several other state-of the- art intrusion detection systems.

The evaluation metrics used in the proposed system are as follows:Accuracy: It measures the proportion of correctly classified instances to the total number of instances. It is calculated as follows:

Accuracy = (TP + TN) / (TP + TN + FP + FN)

where,

TP is the number of true positives, TN is the number of true negatives,

FP is the number of false positives, and FN is the number of false negatives.

Precision: It measures the proportion of true positives to the total number of instances predicted as positive. It is calculated as follows:

Precision = TP / (TP + FP)

Recall: It measures the proportion of true positives to the total number of positive instances. It is calculated as follows:

Recall = TP / (TP + FN)

F1-Score: It is the harmonic mean of precision and recall. It is calculated as

follows:

F1-Score = 2 * (Precision * Recall) / (Precision + Recall)

Area Under the Receiver Operating Characteristic Curve (AUC-ROC): It measures the ability of the model to distinguish between positive and negative

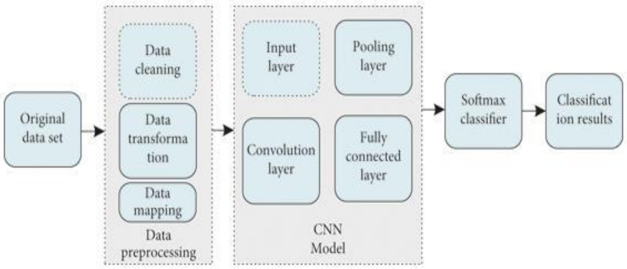


Fig -2: CNN-based models

V. MATHEMATICALMODEL

# ATTACKS IDS 2017

attack_names = ['SSH-Patator', 'DoS GoldenEye', 'PortScan',

'DoS Slowhttptest', 'Web Attack Brute Force', 'Bot', 'Web

Attack Sql Injection', 'Web Attack XSS', 'Infiltration', 'DDoS',

'DoS slowloris', 'Heartbleed', 'FTP-Patator', 'DoS

Hulk','BENIGN'] indices = range(len(attack_names))

zip_iterator = zip(attack_names,indices)

attacks_dict = dict(zip_iterator)

chosen_connection_features = ['Source Port', 'Destination Port',

'Bwd Packet Length Min', 'Subflow Fwd Packets',

'Total Length of Fwd Packets', 'Fwd Packet

Length Mean', 'Total Length of Fwd Packets',

'Fwd Packet Length Std', 'Fwd IAT Min', 'Flow

IAT Min', 'Flow IAT Mean', 'Bwd Packet Length Std',

'Subflow Fwd Bytes', 'Flow Duration', 'Flow

IAT Std', 'Active Min','Active Mean', 'Bwd IAT Mean',

'Subflow Bwd Bytes', 'Init_Win_bytes_forward',

'ACK Flag Count','Fwd PSH Flags','SYN Flag Count',

'Flow Packets/s', 'PSH Flag Count', 'Average Packet Size']

indices = range(len(chosen_connection_features))

zip_iterator = zip(chosen_connection_features, indices)

chosen_features_dict = dict(zip_iterator)

possible_protocols = {'6':[0.0,0.0,1.0],'17':[0.0,1.0,0.0],'0':[1.0,0.0,0.0],'':[0.0, 0.0,0.0]} # ----------------------------------

def normalization_function(feature, name):

if name in chosen_connection_features and

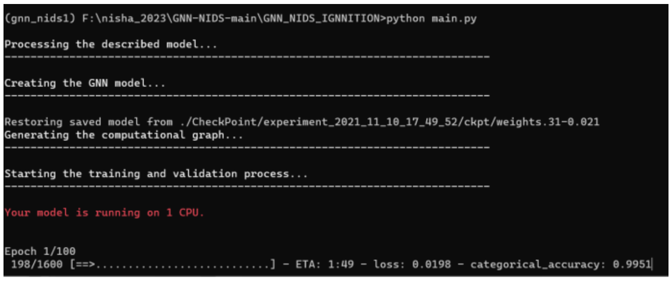
(name+'_mean') in params_norm['PARAMS'] and

float(params_norm['PARAMS'][name + '_mean']) != 0: feature

= (feature - float(params_norm['PARAMS'][name + '_mean']))

/ float(params_norm['PARAMS'][name + '_std'])

VI. SYSTEMOVERVIEW



```
(gnn_nids1) F:\nisha_2023\GNN-NIDS-main\GNN_NIDS_IGNNITION>python main.py

Processing the described model...
---------------------------------------------------------------------------

Creating the GNN model...
---------------------------------------------------------------------------

Restoring saved model from ./CheckPoint/experiment_2021_11_10_17_49_52/ckpt/weights.31-0.021
Generating the computational graph...
---------------------------------------------------------------------------

Starting the training and validation process...
---------------------------------------------------------------------------

Your model is running on 1 CPU.


Epoch 1/100
 198/1600 [==>...........................] - ETA: 1:49 - loss: 0.0198 - categorical_accuracy: 0.9951
```

Fig -4: GNN Model

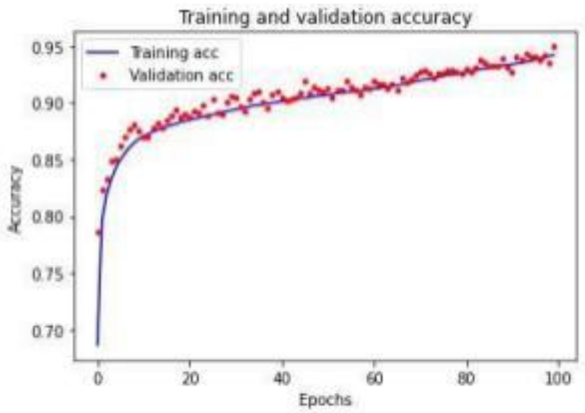


Fig -5: training and validation accuracy

VII. CONCLUSION

Motivate the use of Graph Neural Networks (GNNs) to develop accurate and robust NIDS. I argue that, to achieve effective MLbased NIDS, it is essential not only to collect relevant patterns on individual flow features, but also to capture meaningful structural flow patterns that characterise different attacks (e.g., DDoS, port/network scans, brute force

attacks). To this end, I will first present a graph representation that properly represents the properties of flows and their relationships in the network. Then, I present a novel GNN architecture specifically designed to learn and generalize over the previous graph-structured information. First, I have tested the accuracy of our model, showing comparable results to state-of-the-art ML-based NIDS in the well-known CIC-IDS2017 dataset. Then, we have tested the solution against two common adversarial attacks that intentionally modify relevant flow features on attack-related flows (packet size and inter-arrival times) to evade detection.

The results show that, while the proposed GNN model is completely robust to these attacks, state-of-the-art ML models for NIDS degrade their accuracy by up to 50%. This is mainly thanks to the capability of the proposed GNN model to learn the inherent structural flow patterns that compose different attacks. These structural patterns represent a deeper knowledge about attacks, as they typically remain unchanged over time, and across different networks

REFERENCES

[1]. IBM and the Ponemon Institute. Cost of a Data Breach Report 2020. https://www.ibm.com/security/databreach.

[2]. Ansam Khraisat, Iqbal Gondal, Peter Vamplew, and Joarder Kamruzzaman. 2019.Survey of intrusion detection systems: techniques, datasets and challenges. Cybersecurity 2,1 (2019),1–22.

[3]. Paulo Angelo Alves Resende and Andr´e Costa Drummond. 2018. A survey of random forest based methods for intrusion detection systems. ACM Computing Surveys (CSUR) 51, 3 (2018),1–36.

[4]. Robin Sommer and Vern Paxson. 2010. Outside the closed world: On using machine learning for network intrusion detection. In IEEE symposium on security and privacy. 305–316.

[5]. G. Cong, C. Jensen, and D. Wu. “Efficient retrievalof the top-k most relevant spatial web objects”. In: Proc. VLDB Endow. 2.1 (2009), pp. 337–348.

[6]. Igino Corona, Giorgio Giacinto, and Fabio Roli. 2013. Adversarial attacks against intrusion detection systems: Taxonomy, solutions and open issues. Information Sciences 239 (2013).

[7]. David Pujol-Perich, Jose Suarez-Varela, Miquel Ferriol Games, Shihan Xiao, Bo Wu, Albert Cabellos Aparicio, and Pere Barlet-Ros. 2021. IGNITION: A framework for fast prototyping of Graph Neural Networks. MLSys Workshop on Graph Neural Networks and Systems (GNNSys).

[8]. Iman Sharafaldin, Arash Habibi Lashkari, and Ali A Ghorbani. 2018. Toward generating a new intrusiondetection dataset and intrusion traffic characterization. ICISSp 1 (2018), 108–116

[9]. G. R. Hjaltason and H. Samet. “Distance browsing in spatial databases”. In: TODS 2 (1999), pp. 256–318.

[10]. Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. IEEE transactions on neural networks and learning systems 32, 1 (2020), 4–24..

[11]. N. Mamoulis and D. Papadias. “Multiway spatial joins”. In: TODS 26.4 (2001), pp. 424–475.

[12]. D. Papadias, N. Mamoulis, and B. Delis. “Algorithms for querying by spatial structure”. In: VLDB (1998), p. 546.

[13]. D. Papadias, N. Mamoulis, and Y. Theodoridis. “Processing and optimization of multiway spatial joins using R-trees”. In: PODS (1999), pp. 44–55.

[14]. J. M. Ponte and W. B. Croft. “A language modeling approach to information retrieval”. In: SIGIR (1998), pp. 275–281.

[15]. Jo˜ao B. Rocha-Junior et al. “Efficient processing of top-k spatial keyword queries”. In: Proceedings of the 12th international conference on Advances in spatial and temporal databases. 2011, pp. 205–222.

[16]. Gerard Biau and Erwan Scornet. 2016. A random forest guided tour.Test25,2(2016),197–227.