A Novel DDoS Detection and Multi-Class Classification Method: A Graph Convolutional Network Approach

Braden J. Saunders

Submitted in partial fulfillment
of the requirements for the degree of

Master of Science

Department of Computer Science
Faculty of Mathematics and Science
Brock University
St. Catharines, Ontario

Abstract

Distributed Denial of Service (DDoS) is an attack that overwhelms the cyber critical infrastructure system with malicious packets causing it to become unresponsive, which precludes legitimate users from accessing the target system. This work leverages a deep learning method known as Graph Convolutional Network (GCN) to empower DDoS detection systems. The proposed GCN model consists of three hidden layers, each with 128 neurons. Considering the Canadian Institute for Cybersecurity CIC-IDS 2017 dataset, the proposed model achieves an overall accuracy of 99.95%, along with a value of 99.95% for each of the precision, recall, and F1-score metrics for the binary DDoS classification problem. For the multi-class DDoS classification problem, the model scores an overall accuracy of 98.94% and precision, recall, and F1-score values of over 93% for all classes. These results support the use of the proposed GCN DDoS detection method in practice.

Contents

Chapter 1

Introduction

Critical infrastructure providers across different sectors such as telecommunications, healthcare, and education, to name a few, have been victimized by cybercriminal organizations on a daily basis [2]. The constant growth of cyberspace spurred by the omnipresence of the Internet, online social networks, wireless networks, and cloud systems has paved the way for a fully connected society and thriving digital economy. Concomitantly, this has also paved the way for cybercriminal organizations to successfully wreak havoc on our society's most critical infrastructure systems.

Zaya Group, a global communications infrastructure provider, states in their annual report titled "The State of DDoS Attacks" that Q2 of 2023 saw an 387% increase in DDoS attacks compared to Q1 of the same year. Furthermore, approximately 50% of the attacks were targeted towards telecommunications companies, which totalled over 37,000 attacks in the first half of 2023 [1].

Such an increase in cyber attacks is alarming and demands innovative prevention, detection, and mitigation approaches to address these threats. Artificial intelligence (AI) assisted cyber defenses are among the most promising approaches to defeat these cyber threats. In fact, thanks to the ability to discover hidden partners from static and spatial-temporal data and to generalize learnings, i.e., to adapt to unseen data, AI cyber defenses based on Naïve Bayes, K-nearest neighbor, Logistic Regression models, and clustering techniques became widely used to defeat network attacks [3]. More recently, deep learning models have taken the leadership in detection accuracy [4].

This paper addresses the problem of a deep learning-based anomaly detection system. More specifically, we propose a Graph Convolutional Neural network (GCN) model to detect cyber attacks. GCN comprises of a deep learning technique that can learn a graph representation and aggregate or update node information from the neighborhoods in a convolutional manner [5]. Graphs are widely used in many

real-world applications, such as computer vision, fraud detection and social network analysis, to name a few [6]. We develop a GCN anomaly detection model to efficiently spot DDoS attacks, which have become one of the most destructive and effective cyber-attacks that target critical infrastructure systems [7].

To assess the performance of the proposed GCN-empowered anomaly DDoS detection model, we leveraged 2 of the DDoS subsets which are part of the UNB CIC-IDS2017 dataset to train and test it. We found that in a binary classification environment, the proposed model achieved an overall accuracy of 99.95%, along with a value of 99.95% for each of the precision, recall, and F1-score metrics for the DDoS class. We also found in a separate multi-classification test that the model achieved an overall accuracy of 98.94% and precision, recall, and F1-score values of over 93% for all classes.

The remainder of this paper is organized as follows. The next Chapter presents the proposed GCN model. Chapter 3 describes the UNB-CIC DDoS dataset and data preprocessing techniques used. Chapter 4 discusses the metrics used for evaluation, and the results. Finally, Chapter 5 concludes the paper and sheds light on future research avenues.

Chapter 2

Proposed Model

The classification model consists of a Graph Convolutional Network (GCN), which is a deep learning approach based on convolutional neural networks specifically designed to learn using graph data [8]. The formula expression for a single GCN layer is exhibited in Eq. (2.1), where H is the node matrix, X is the node features matrix, σ is an activation function, $\tilde{A}$ is the graph adjacency matrix, $\tilde{D}$ is the graph degree matrix, and θ is the matrix of trainable data.

$$H = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}X\theta) \tag{2.1}$$

GCN models are a common choice for tasks that require graph classification, node classification, edge classification, and link/edge prediction. Because the network packets are represented as edges in the network graph, the model can be described as an edge classification problem, that is, the goal of the model is to successfully classify each edge in the graph. The model consists of 3 hidden layers where each hidden layer contains 128 features. A diagram of the model, which includes all of its layers, is presented in Figure 2.1.

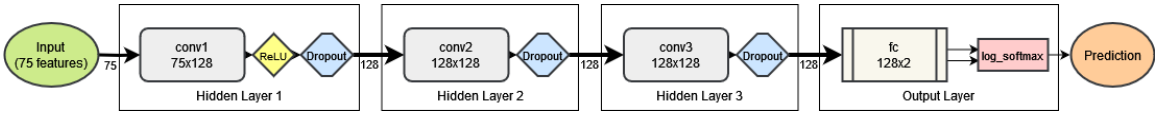

Figure 2.1: The Graph Convolutional Network model diagram for a binary classification, including all layers.

The data is passed through the model in the following manner:

- The 75 packet features are passed into the first hidden layer where they are outputted as 128 hidden features.

- The Rectified Linear Unit (ReLU) operation is performed on the outputted hidden features. The ReLU is an activation function which introduces non-linearity and is commonly used in deep learning networks [9]. The formula for the ReLU operation is defined in Eq. (2.2).

$$ReLU(x) = (x)^+ = max(0, x) \tag{2.2}$$

- The output of the ReLU activation function is then passed into a dropout operation where 50% of the data is nullified, this is done to prevent model overfitting.

- The altered data is then passed into the second hidden layer where another 128 hidden features are outputted.

- The dropout operation is once again performed on the data, before it is passed into the third hidden layer, where once again, 128 hidden features are outputted.

- The dropout operation is performed a final time, before a linear transformation is applied to convert the data from 128 features down to the number of output classes (e.g., 2 for a binary classification). The formula for the Linear transformation is defined in Eq. (2.3). In the formula, A is a randomly initialized weight matrix whose shape depends on the number of output and input features. b is a randomly initialized bias vector which is the size of the number of output features.

$$y = xA^T + b \tag{2.3}$$

- The softmax function followed by a logarithm is applied to the final layer. The softmax function will convert the linear transformation output to a probability distribution over the predicted classes. The softmax function is defined in Eq. (2.4). The logarithm function is then applied to scale the values down which makes them easier to interpret and more efficient to compute. The final probability distribution is interpreted to determine the model's prediction for each packet.

$$Softmax(x_i) = \frac{exp(x_i)}{\sum_j exp(x_j)} \tag{2.4}$$

The GCN model is trained using the Adam optimizer algorithm [10] with a training rate of 0.01, along with the CrossEntropyLoss function, which is a metric that measures the model performance by calculating the difference between the predicted class probability, which was calculated with the softmax function, to the actual class label. Through experimentation, we found that iterating the training loop 100 times (epoch=100) yielded the optimal model.

Chapter 3

Dataset & Processing

3.1 Dataset

The Canadian Institute of Cyber Security based at The University of New Brunswick generates and publicises various datasets which cover a wide range of cyber attacks. One dataset which is of particular interest is the "Intrusion Detection Evaluation Dataset" which was made available in 2017. One of the main focuses of this dataset, which is also known also as "CIC-IDS2017", was generating realistic background traffic so the data resembled true real-world data as closely as possible. The dataset was generated using an attacker network which consisted of 2 clients, which run Windows and Kali Linux respectively, and a separate victim network which consists of 12 clients running various versions of Windows, Linux, and MacOS. The data collected consists of every packet that was sent or received on the network over the stated period. Each packet was analysed using an application called CICFlowMeter and over 80 network flow features were extracted from each one. The attacks captured in this dataset include Brute Force FTP, Brute Force SSH, DoS, Heartbleed, Web Attack, Infiltration, Botnet and DDoS. The data is divided into 8 subsets which contain variations of the individual attacks that were captured. The individual data subsets are available for download in CSV format, where each row corresponds to a packet flowing between 2 clients on the network, and each column is a network flow feature [11]. The subsets of particular interest for the purpose of training and testing our model are the "DDoS LOIT" set and the "DoS / DDoS" set. The DDoS LOIT set contains 225,745 packets and the DoS / DDoS set contains 792,703 packets. Both of these sets are labelled and contain 85 columns which consist of the Flow ID, Source IP, Source Port, Destination IP, Destination Port, Protocol, Timestamp, Label and 77 numerical features extracted using the CICFlowMeter application. The

DDoS LOIT dataset focuses specifically on DDoS attacks and is binary, that is to say, the packets are labeled as either "BENIGN" indicating that it is regular traffic, or "DDoS" indicating that the packet is part of an ongoing DDoS attack. The DoS / DDoS set focuses on 4 types of common DoS attacks, namely, slowloris, Slowhttptest, Hulk, and GoldenEye. This set also contains labelled packets that are from another cyber attack unrelated to DDoS, known as Heartbleed. Detecting the Heartbleed attack is outside the scope of what we intend our model to accomplish, so the 11 packets labelled as Heartbleed are removed from the dataset prior to training. Each data subset has 2 CSV files available for download, that is, the "Labelled Flows" version which contains all the columns, and the "Machine Learning" version which omits the Flow ID, Source IP, Source Port, Destination IP, Protocol, and Timestamp columns. Although we are using these datasets in a machine learning context, we opted to use the "Labelled Flows" files as we require the Source and Destination IP information for each packet in order to construct a graph of the data. However, once the data is preprocessed and the graph is constructed, the Flow ID, Source Port, Destination IP, Timestamp, and Label columns are ignored by the application, and are not utilized by the model during the training process. A sample 2 rows of the LOIT dataset is presented in Table 3.1.

Table 3.1: Dataset Sample

Source IP	**Destination IP**	**Flow IAT Max**	**Flow IAT Min**	**Fwd IAT Total**	**...**	**Label**
149.202.208.196	192.168.10.15	15	15	0	...	BENIGN
172.16.0.1	192.168.10.50	1292730	2	747	...	DDoS

3.2 Data PreProcessing

A number of steps are taken to prepare and process the data prior to it being inputted into the GCN model. The data must be converted from its format in the CSV file to a graph represented by tensors that the model can utilize. The data can be structured as a graph by representing each network endpoint as a node and each packet flowing between those endpoints as edges. For the CIC-IDS2017 datsets, this means that each unique source and destination IP in the CSV file will be a graph node, and each row in the CSV file will be a graph edge. A visual representation of the graph structure that is created from the dataset is illustrated in Figure 3.1.

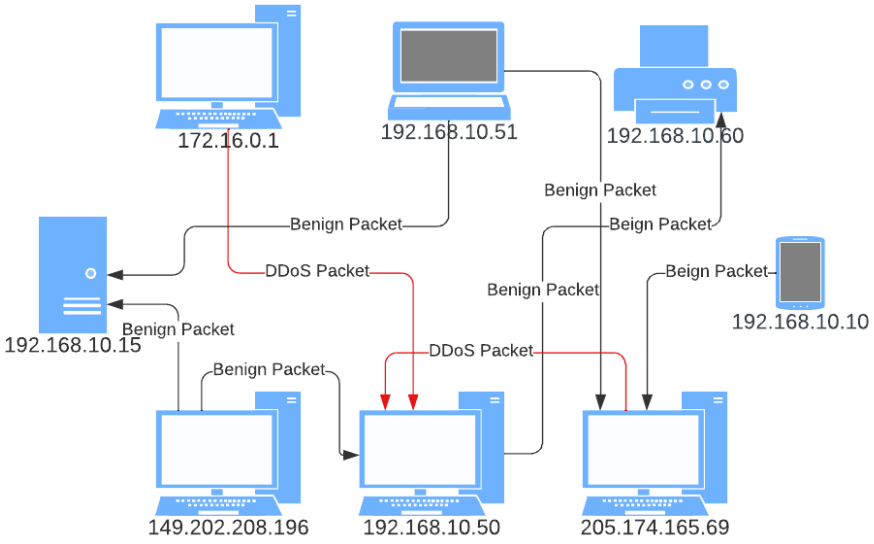

Figure 3.1: Each device has an IP address which is a node on the graph. The lines connecting devices show the packets flowing between them, each one represents a single row in the dataset.

To read in, prepare, process, and convert the CSV data to a graph structure, a number of steps need to be taken. Firstly, the Python *Pandas* Library is used to load the data into the Python application. Columns which contain unreadable information such as values that are Not a Number (NaN) or infinite are dropped (in our case this happens for only the "Flow Bytes/s" and "Flow Packets/s" columns). Next, the LabelEncoder class from the *sklearn* library is used to convert the source and destination IP addresses to numerical labels. *PyTorch* is then used to convert the now encoded source and destination IP labels to a tensor. With the processing of the IP addresses finished, the application then drops the unnecessary Flow ID, Source IP, Source Port, Destination IP, Protocol, and Timestamp columns, the columns remaining are converted to a tensor as these are the features which model will analyse. Lastly, the data is split into training and testing sets, where 80% of the data is used for training the model, and 20% is used for analysing the effectiveness of the model. The final data is stored in a *PyTorch Geometric* Data object before being passed to the GCN model.

Chapter 4

Evaluation & Results

4.1 Evaluation

A deep learning model's performance is typically measured using the precision, recall, and F-1 score metrics for each class, along with the overall accuracy of the model. For binary classifications, a Receiver Operating Characteristic (ROC) curve is typically used to present a visual representation of the model's performance. The area under the receiver operating characteristic curve (AUC) is also often used as a method of quantifying the curve representation. A brief description of each metric, along with its mathematical formula is provided. In the following formulas, TP = True Positive, FP = False Positive, TN = True Negative, and FN = False Negative.

4.1.1 Precision

Precision measures how many of the model's predictions for a single classification were actually correct. The precision value is a percentage which represents the number of correctly made predictions for a class, in comparison to the total number of predictions made for that class. The formula for precision is exhibited in Eq. (4.1).

$$Precision = \frac{TP}{TP + FP} \tag{4.1}$$

4.1.2 Recall

Recall measures how many data points in a class were correctly predicted by the model. The recall value is a percentage which represents the correctly predicted data points for a class, in comparison to the total number of data points in that class. The

formula for recall is exhibited in Eq. (4.2).

$$Recall = \frac{TP}{TP + FN} \tag{4.2}$$

4.1.3 F-1 Score

F-1 score is defined as the harmonic mean between the precision and recall metrics. The F-1 score is a particularly valuable metric as it encapsulates both precision and recall as a single value, which can be easily compared with other models. The formula for recall is exhibited in Eq. (4.3).

$$F\text{-}1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{4.3}$$

4.1.4 Accuracy

Accuracy measures how many of the model's predictions in total were correct. The accuracy value is a percentage with respect to the total number of predictions made. Unlike many of the other metrics, accuracy is measured across all the classes, so there is only 1 accuracy measurement per evaluation of the model. The formula for accuracy is exhibited in Eq. (4.4).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.4}$$

4.1.5 Receiver Operating Characteristic

The Receiver Operating Characteristic is a graph which visually illustrates the performance of a model on a binary classification. The graph is generated by plotting the True Positive Rate (TPR) against the False Positive Rate (FPR). The TPR is also referred to as the recall, the formula for which has already been exhibited in Eq. (4.2). The FPR formula is exhibited in Eq. (4.5).

$$FPR = \frac{FP}{FP + TN} \tag{4.5}$$

4.1.6 Area Under Curve

The Area Under Curve (AUC) is a measure of the space under the ROC curve, it is a method of quantifying the ROC graph data. The metric is a value between 0 and 1, where values closer to 1 represent a higher model performance.

4.2 Results

The proposed model was trained and evaluated on the CIC-IDS2017 LOIT dataset. The results of the evaluation, which is comprised of the metrics discussed, are presented in Table 4.1.

Table 4.1: Classification Report

	Precision	**Recall**	**F1-score**	**Support**
Benign	0.9994	0.9994	0.9994	19523
DDoS	0.9995	0.9995	0.9995	25703
Accuracy			0.9995	45226
Macro avg	0.9995	0.9995	0.9995	45226
Weighted avg	0.9995	0.9995	0.9995	45226

As can be seen in Table 4.1, the overall model accuracy is 99.95%, and the precision, recall, F1-Score for the DDoS and Benign classes are all 99.95% and 99.94% respectively. For further context, the confusion matrix for this evaluation is provided as well in Figure 4.1. If we represent benign labels as "True" and DDoS labels as "False", then the values in Figure 4.1 represent (from left to right, top to bottom) the true positive (TP) predictions, the false negative (FN) predictions, the false positive (FP) predictions, and the true negative (TN) predictions.

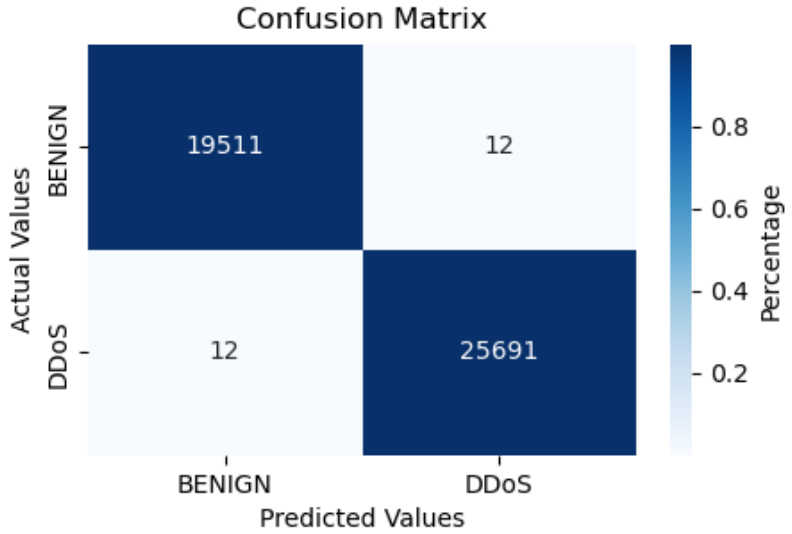

Figure 4.1: Confusion matrix for LOIT dataset, represented as a percentage heat map.

As this is a binary classification, an ROC curve can be generated, and an AUC value calculated. The ROC curve, including AUC value, is presented in Figure 4.2.

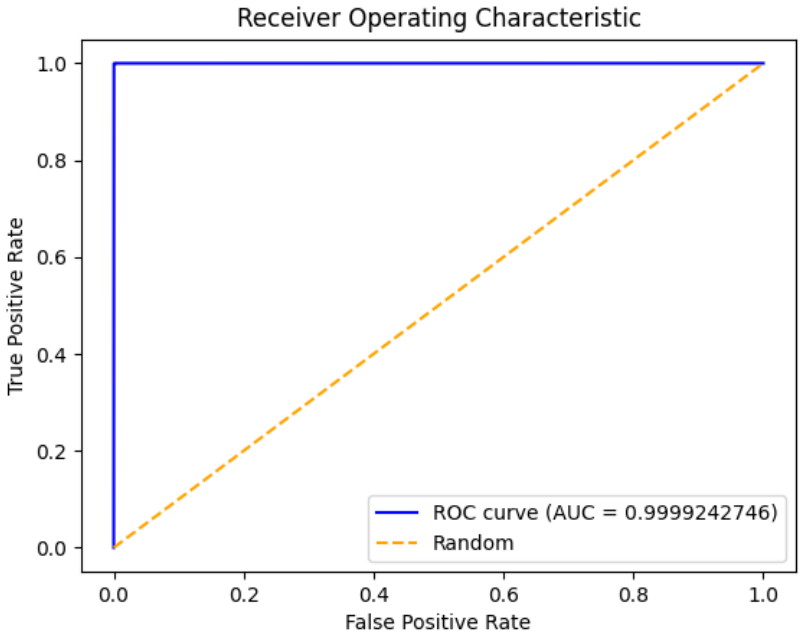

Figure 4.2: The ROC curve representing the model's performance in a binary classification setting.

Based on the results presented in Table 4.1, the proposed model performs exceptionally well in a binary classification environment, with all of the metrics reporting results in the 99.94%-99.95% range. The ROC curve is a sharp corner shape with an AUC value of 0.99992, this indicates a near perfect model accuracy, which is consistent with the results in Tables 4.1. The confusion matrix in Figure 4.1 shows that the model correctly predicted 25,691 packets as DDoS, and only incorrectly predicted 12 packets to be DDoS when they were actually benign. Similarly, it can be seen that the model correctly predicted 19,511 packets to be benign and only incorrectly 12 packets to be benign when they were actually DDoS.

The proposed model was also trained and evaluated in a multi-classification environment, specifically, on the CIC-IDS2017 DOS / DDoS dataset. The results of this evaluation, which are comprised of the same metrics, apart from thead ROC and AUC which are only applicable to binary classifications, are presented in Table 4.2.

Once again, further context on the results is provided with the confusion matrix for this evaluation in Figure 4.3

As can be seen from the metrics illustrated in Table 4.2 along with the confusion

Table 4.2: Classification Report

	Precision	Recall	F1-score	Support
Benign	0.9976	0.9870	0.9923	87916
DoS GoldenEye	0.9917	0.9574	0.9742	2113
DoS Hulk	0.9764	0.9984	0.9873	46118
DoS Slowhttptest	0.9347	0.9355	0.9351	1117
DoS slowloris	0.9472	0.9236	0.9353	1126
Accuracy			0.9894	138390
Macro avg	0.9695	0.9604	0.9648	138390
Weighted avg	0.9896	0.9894	0.9894	138390

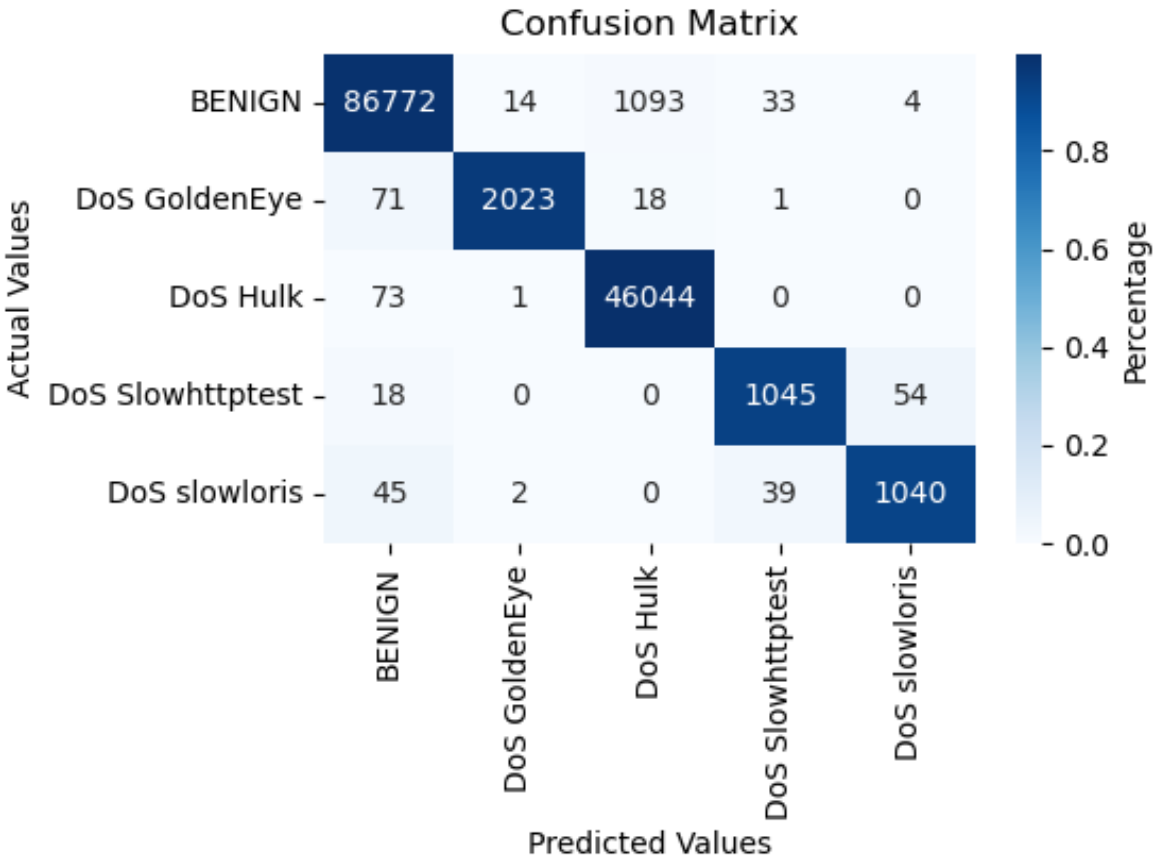

Figure 4.3: Confusion matrix for DoS / DDoS dataset, represented as a percentage heat map.

matrix presented in Figure 4.3, the model fairs quite well in a multi-classification environment. The metric values are all over 92%, with many in the 98%-99% range. The metrics indicate that the model is exceptional at correctly identifying packets

labelled as Benign, DoS GoldenEye, and DoS Hulk with F-1 scores of 99.23%, 97.37%, 98.74% respectively for each of these classes. The model is less accurate but still adept at correctly identifying packets labelled as DoS Slowhttptest and DoS slowloris, with F-1 scores of 93.15% for both. Whereas the classification report in Table 4.2 showcases the metric percentages, the confusion matrix in Figure 4.3 shows specifically how the model classified each packet. For an example, it can be seen that a notable number of benign packets (1088) were incorrectly predicted to be DoS Hulk. Overall, the confusion matrix confirms the model's high level of accuracy. The diagonal values in a confusion matrix represent the correctly classified data points, and the diagonal values in Figure 4.3 are the highest of all the values by a significant margin.

4.3 Comparison with Related Works

In [12], Sabeel et al. proposed a Deep Neural Network (DNN) model and a Long Short Term Memory (LSTM) model, which the authors trained and tested in a multi-classification environment with the CIC-IDS2017 DOS / DDoS dataset. The DNN and LSTM model achieved an overall accuracy of 99.67% and 99.34% respectively. These accuracy values are slightly higher, but still comparable to the values that our GCN model was able to achieve on that same dataset.

In [13], R. Doriguzzi-Corin et al. proposed a model using a Convolutional Neural Network (CNN) which the authors trained and tested on 3 datasets, including the CIC-IDS2017 dataset. Specifically, the LOIT subset was used, which provides a binary classification environment. That model was able to achieve an accuracy of 99.67%. The precision, recall, and F1-scores are also provided which are 99.39%, 99.94%, and 99.66% respectively. While these results are comparable to ours, our GCN model does achieve a have a higher performance with this dataset across all metrics.

In [14], Lucky et al. proposed a DDoS dection solution using a Decision-Tree (DT) model. The authors trained and tested their model on 2 CIC datasets, one of which was the IDS2017 dataset. Although it is unclear which subset was used, the authors report that their model was able to achieve and accuracy of 99.69%, the recall metric appears to be approximately 99.7% based on the result graph provided.

Chapter 5

Conclusion

In this work, we have proposed a method of positively identifying packets that were maliciously sent to form part of a DDoS attack. Our proposed method utilizes a Graph Convolutional Network deep learning model. We tested our model on the CIC-IDS2017 LOIT dataset, and the results proved that the model performs exceptionally in a binary classification setting, with an overall accuracy of 99.95% and F-1 scores of over 99.9% for both classes. We also tested our model on the CIC-IDS2017 DOS / DDoS dataset, and the results indicated that the model is also effective at identifying packets in a multi-classification setting, with an overall accuracy of 98.94% and F-1 scores of over 93% for each class. As critical infrastructure providers continue to be victimized by cybercriminal organizations, it is important for system administrators to have access to the tools and technology needed to be able to detect and prevent malicious packets from entering their infrastructure. In the future, we would like to continue fine tuning the model to potentially further increase its accuracy. Furthermore, we would like to implement this detection system into a real-world application that can analyse, detect, and block network traffic in real time.

Bibliography

[1] "The State of DDoS Attacks," Zayo Group Holdings, Inc., Accessed: Jan. 31, 2024. [Online]. Available: https://go.zayo.com/zayo-ddos-protection-ebook/

[2] M. V. Pawar and J. Anuradha, "Network Security and Types of Attacks in Network", Procedia Comput Sci, vol. 48, no. C, pp. 503–506, Jan. 2015, doi: 10.1016/J.PROCS.2015.04.126.

[3] M. Aljabri, S.S. Aljameel, R.M.A. Mohammad, S.H.Almotiri, S. Mirza, F.M. Anis, M. Aboulnour, D.M. Alomarii, D.H., Alhamed, and H.S. Altamimi, "Intelligent Techniques for Detecting Network Attacks: Review and Research Directions", Sensors (Basel). 2021 Oct 25;21(21):7070. doi: 10.3390/s21217070. PMID: 34770375; PMCID: PMC8587628.

[4] M. S. Elsayed, N. -A. Le-Khac, S. Dev and A. D. Jurcut, "DDoSNet: A Deep-Learning Model for Detecting Network Attacks", 2020 IEEE 21st International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM), Cork, Ireland, 2020, pp. 391-396, doi: 10.1109/WoWMoM49955.2020.00072.

[5] S. Zhang, H. Tong, J. Xu, and R. Maciejewski, "Graph convolutional networks: a comprehensive review," Comput Soc Netw, vol. 6, no. 1, Dec. 2019, doi: 10.1186/S40649-019-0069-Y/FIGURES/1, pp. 1-23.

[6] A. Majeed and I. Rauf, "Graph Theory: A Comprehensive Survey about Graph Theory Applications in Computer Science and Social Networks", Inventions, 5(1), p.10, February 2020

[7] M. A. Al-Shareeda, S. Manickam, and M. A. Saare, "DDoS attacks detection using machine learning and deep learning techniques: analysis and comparison", Bulletin of Electrical Engineering and Informatics, vol. 12, no. 2,, 2023, doi: 10.11591/eei.v12i2.4466, pp. 930-939.

[8] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks", CoRR, abs/1609.02907, 2016, http://arxiv.org/abs/1609.02907

[9] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for Activation Functions", CoRR, abs/1710.05941, 2017, https://arxiv.org/abs/1710.05941

[10] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization", CoRR, abs/1412.6980, 2017, https://arxiv.org/abs/1412.6980

[11] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization", 4th International Conference on Information Systems Security and Privacy (ICISSP), Portugal, January 2018

[12] U. Sabeel, S. S. Heydari, H. Mohanka, Y. Bendhaou, K. Elgazzar and K. El-Khatib, "Evaluation of Deep Learning in Detecting Unknown Network Attacks," 2019 International Conference on Smart Applications, Communications and Networking (SmartNets), Sharm El Sheikh, Egypt, 2019, doi: 10.1109/SmartNets48225.2019.9069788, pp. 1-6.

[13] R. Doriguzzi-Corin, S. Millar, S. Scott-Hayward, J. Martinez-Del-Rincon, and D. Siracusa, "LUCID: A Practical, Lightweight Deep Learning Solution for DDoS Attack Detection," 2019, IEEE Transactions on Network and Service Management, https://doi.org/10.1109/TNSM.2020.2971776

[14] G. Lucky, F. Jjunju and A. Marshall, "A Lightweight Decision-Tree Algorithm for detecting DDoS flooding attacks," 2020 IEEE 20th International Conference on Software Quality, Reliability and Security Companion (QRS-C), Macau, China, 2020, pp. 382-389, doi: 10.1109/QRS-C51114.2020.00072.