Detecting Network Intrusion

Ty Misiorek (zem4by), Sanket Doddabendigere (kfr9mc), Shirley Li (fky9xf), Shawn Thomas (yzy2bt), Ansh Pathapadu (tqc7wn)

1 Problem Statement

In the modern world, nearly everything can be done online. Whether it be handling finances, work meetings, shopping, or accessing educational material, a large portion of human life has shifted towards the internet. In the digital world, network attacks are growing concern which can cause severe financial damages and shut down entire operations. Random Forest, Support Vector Machine, and Logistic Regression have been popular machine learning models for intrusion detection as they are simple to train and explain. However, with increasing complexity and multidimensionality of network traffic, these models have struggled to detect the complex relationships between data points, thereby being less effective in real-world applications [1]. Additionally, most traditional models rely on datasets that may not cover the full spectrum of current attack patterns, limiting their generalizability in unfamiliar environments [2].

In this project, we aim to address the growing complexities of network traffic by using graph-based machine learning algorithms to detect and predict network intrusions. By modeling network interactions as a graph, where nodes represent a detected activity on a port, and edges connecting the nodes with the most similar features, we will make use of traditional Graph Neural Networks such as Graph Convolutional Networks and Graph Attention Networks to make these predictions. Additionally, we will use some of the more traditionally used and less complex algorithms such as Logistic Regression and Random Forests to use as a baseline comparison for our graph models. By exploring these models, we aim to identify the highest-performing method of predicting network attacks, and to give a recommendation for what model is best suited for the task.

2 Literature Review

Increasing complexity of network traffic has made it more difficult to accurately predict network intrusions. In order to overcome such problems, researchers have turned towards graph-based models, which are more effective in representing relationships between data points. One of the breakthroughs here was Graph Convolutional Networks (GCNs) by Kipf and Welling [3]. GCNs allow the model to learn from the local nodes so that patterns that conventional models miss can be identified. Building on this, Velickovic et al. proposed Graph Attention Networks (GATs) [4], which assign different levels of importance to neighboring nodes to refine the model's ability to identify subtle anomalies. All these methods have already proven to be effective in detecting fraud and IoT security and have potential for network intrusion detection as well [5].

More recently, the k-Nearest Neighbor Learning with Graph Neural Networks (kNNGNN) introduced by Kang et al. [6] combines the classic kNN algorithm with graph-based learning. Instead of treating each data point in isolation, kNNGNN constructs a graph where each node (representing a network traffic instance) is linked to its k-nearest neighbors based on feature similarity. It enables the model to learn local relationships while using the power of GNNs to refine feature representations. This will be particularly useful in network intrusion detection because it dynamically adapts to evolving attack patterns, reducing the reliance on predefined rules or extensive hyperparameter tuning.

Despite these advances, there are still gaps in research. While Bilot et al. presented the first overview of GNN-based intrusion detection in 2023, many advanced GNN-based methods are still unexplored in this area[7]. Traditional models are still struggling against the complexity of modern network traffic, and many graph-based methods are still computationally costly and require more tuning. This paper aims to address these gaps by combining classical machine learning methods with advanced graph neural networks and comparing them in terms of precision.

3 Data Collection

The CIC-IDS2017 dataset will be used for this study, as it provides a well-structured and labeled dataset for network intrusion detection. This dataset was chosen because it includes a diverse range of real-world attack scenarios and normal network traffic, making it suitable for both supervised and unsupervised anomaly detection tasks. The data is publicly available through the Canadian Institute for Cybersecurity and Kaggle, eliminating the need for manual data collection. The dataset consists of two main components: network traffic features that capture details of connections, such as packet sizes, timestamps, and protocol types, and labeled attack categories that indicate whether the traffic is normal or malicious. If the dataset were not readily available, an alternative approach would involve collecting real-time network traffic using tools like Wireshark or Zeek in a controlled environment and labeling attacks based on known threat patterns. However, this would be incredibly labor intensive, and using this dataset allows for immediate analysis with minimal preprocessing, ensuring a reliable and standardized approach to anomaly detection in network security.

4 Methods

To develop an effective model to detect network attacks, we will employ several methods to quantify the risk of network intrusion. We will start by employing less complex methods such as logistic regression and random forests to develop a baseline predictions. Then we will test more complex models, applying graph neural networks, specifically Graph Convolutional Networks (GCNs) and Graph Attention Networks (GATs).

4.1 Logistic Regression

We will start with logistic regression for a baseline predictor of network intrusion. Through the regression, we will model the probability of an attack compared to benign traffic. Its simplicity allows for a convenient means to identify which covariates explain the highest proportion of the variance. Identifying which variables to control for is a challenge. Due to the high count of features included in the dataset (79), we plan on reducing the number of covariates to prevent possible overfitting and multicollinearity. We will run several iterations of the regression, with varying numbers and combinations of independent variables to identify which are essential in properly identifying network attacks, and which should be included in the other models.

Despite offering useful insight on feature significance, its limitations force us to look towards other methods of prediction. Performing a regression assumes a linear relationship between the dependent and the covariates, so it may fail to identify non-linear relationships contained within the data.

4.2 Random Forest

Next, we will apply a random forest classifier to predict network anomalies. Unlike logistic regression, random forest classifiers are able to capture non-linear relationships among the covariates. Additionally, using the random forest reduces the risk of multicollinearity because it uses random feature selection at splits, thus the whole feature set can be included. Similarly, the risk of overfitting is mitigated as random subset of features are included at each split, and averaging over multiple trees, so there will not be a single feature explaining all the variance.

However, the random forest model is limited in interpretability, so it may be difficult to gain insight on the specific relationship between network attacks and the covariates. Also, the distribution of classes within the dataset is moderately imbalanced. Because tree models are sensitive to class imbalance [8], it is possible that the model performs poorly.

4.3 Graph Convolutional Network (GCNs)

Graph convolutional networks (GCNs) are designed to operate on graph-structured data, where each node represents an entity, and the edges represent relationships between them. Although our dataset is not explicitly represented as a graph, we will transform the data into a graph structure using the K-Nearest Neigbors (kNN) algorithm [6]. Despite using the logistic regression to help select which covariates to use in the model, there still may be dimensionality concerns, so we plan on using principal component analysis (PCA) to perform dimensionality reduction. After, we can run a logistic regression on the transformed data to ensure there is not a large loss in explanatory power. Then, we will create the graph structure using kNN, defining nodes as a row in the dataset, and computing distances using Euclidean distance between each node pair.

A key operation of GCNs is neighborhood aggregation, where nodes gain information from its neighbors, learning from it's own features and its neighbors. This will allow the network to not only learn from individual attributes, but how similar nodes are related. A common issue in GCNs is oversmoothing, which occurs when layers are added and the embeddings of nodes become too similar, so to account for this, we will test the GCN with multiple different layers.

4.4 Graph Attention Network (GATs)

Graph Attention Networks are similar to GCNs, with the key distinction being an attention mechanism, which by stacking layers in which nodes can consider their neighbor's features, different weights can be specified for different nodes in the neighborhood, without being computationally expensive. [4]. With the density of features in this dataset, even after dimensionality reduction, there will likely be multiple features that aren't strong predictors of an attack, so having the ability to weigh feature importance will be useful. Ensuring to test out different parameters like number of attention heads and layers, we will be able to have the model perform to its best ability.

5 Evaluation

Our project will be evaluated using key metrics such as the confusion matrix, F1 score, AUC-ROC, and cross-entropy loss. The confusion matrix will help reveal how many true positives, true negatives, false positives, and false negatives that our model will produce. F1 score will measure the balance between the precision and recall metrics. The AUC-ROC will evaluate how well our model is able to distinguish between classes, and the cross-entropy loss metric will measure how accurate our predicted probabilities will be. Furthermore, we will compare our advanced graph-based methods against standard methods like logistic regression and random forest classifiers to evaluate the improvements in detecting network intrusions by using more advanced approaches. In addition, when we drop covariates with PCA for dimensionality reduction, logistic regression will be a useful means of determining whether any predictive power is lost.

6 Timeline

We will aim to finish the project by May 1st. In the first two weeks, we will conduct a literature review on graph-based intrusion detection, explore works on GCNs, GATs, and kNNGNN applied to cybersecurity and fraud detection. We will also identify gaps in research and formulate the hypothesis for the study. In week 3 we aim to reprocess our dataset by handling missing values, normalizing numerical features, and encoding categorical variables. We will transform the data into a graph structure using k-Nearest Neighbors (kNN) to establish relationships between nodes. In weeks 4 and 5, we will implement graph-based models including Graph Convolutional Networks (GCNs), Graph Attention Networks (GATs), and kNNGNN. We will also train and validate these models using the processed graph dataset. In week 6 we will optimize hyperparameters of the graph-based models to enhance accuracy and efficiency. Experiment with different configurations for the number of layers, dropout rates, and attention heads. In the following week we will evaluate the performance of the model using confusion matrix, F1 score, AUC-ROC, and cross-entropy loss. We will then compare the performance of our graph-based models against the baseline models (Logistic Regression, Random Forest), and note our findings. For weeks 8 and 9, we will finalize the report by summarizing findings, discussing insights, and preparing the final presentation. The project report will be submitted by May 1st.

7 Expected Results

By the end of the project, we expect to develop a graph-based intrusion detection model capable of identifying network anomalies with high accuracy. Specifically, we anticipate that Graph Convolutional Networks (GCNs), Graph Attention Networks (GATs), and kNNGNN will effectively capture complex relationships between network traffic patterns, improving the ability to detect cyber threats. Additionally, we expect that the graph-based models will outperform traditional methods like Logistic Regression and Random Forest in terms of precision, recall, and AUC-ROC scores, demonstrating their superiority in handling high-dimensional network data.

References

[1] Sommer, R., & Paxson, V. (2010). Outside the Closed World: On Using Machine Learning for Network Intrusion Detection. In *IEEEXplore*. https://ieeexplore.ieee.org/document/5504793

[2] Liu, H., & Lang, B. (2019). Machine Learning and Deep Learning Methods for Intrusion Detection Systems: A Survey. *Applied Sciences*, 9(20), 4396. https://www.mdpi.com/2076-3417/9/20/4396

[3] Kipf, T. N., & Welling, M. (2017). Semi-Supervised Classification with Graph Convolutional Networks. In *arXiv*. https://arxiv.org/abs/1609.02907

[4] Velickovic, P., et al. (2018). Graph Attention Networks. In *arXiv*. https://arxiv.org/abs/1710.10903

[5] Zhou, J., et al. (2020). Graph Neural Networks: A Review of Methods and Applications. *ScienceDirect*. https://www.sciencedirect.com/science/article/pii/S2666651021000012

[6] Kang, J., et al. (2020). k-Nearest Neighbor Learning with Graph Neural Networks. In *MDPI*. https://www.mdpi.com/2227-7390/9/8/830

[7] Bilot, T., El Madhoun, N., Al Agha, K., & Zouaoui, A. (2023). Graph Neural Networks for Intrusion Detection: A Survey. *IEEE Access*, 11, 49114-49139. https://ieeexplore.ieee.org/document/10123384

[8] Is Random Forest a Good Option for Unbalanced Data Classification?, *Stack Exchange*, (2016), https://stats.stackexchange.com/questions/242833/is-random-forest-a-good-option-for-unbalanced-data-classification, Accessed: Feb. 19, 2025.