5G-NIDD: A Comprehensive Network Intrusion Detection Dataset Generated over 5G Wireless Network

Adit Sandeep Virkar
Masters in Software Engineering
Arizona State University
Tempe, Arizona
avirkar@asu.edu

Arnav Raviraj
Masters in Software Engineering
Arizona State University
Tempe, Arizona
araviraj@asu.edu

Shivanjay Vilas Wagh
Masters in Software Engineering
Arizona State University
Tempe, Arizona
swagh5@asu.edu

Vinay Kantilal Chavan
Masters in Software Engineering
Arizona State University
Tempe, Arizona
vchavhan@asu.edu

***Abstract*—This document is a model and instructions for LaTeX. This and the IEEEtran.cls file define the components of your paper [title, text, heads, etc.]. *CRITICAL: Do Not Use Symbols, Special Characters, Footnotes, or Math in Paper titles or abstracts.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

With the advent of fifth-generation (5G) wireless networks and the nascent development of sixth-generation (6G) technologies, the complexity and sophistication of network infrastructures have significantly increased. This evolution brings unparalleled opportunities for enhanced connectivity, higher data rates, and seamless integration of various devices and applications. However, it also introduces new vulnerabilities and security challenges, primarily due to the expanded attack surface and the incorporation of artificial intelligence (AI) and machine learning (ML) technologies. These advancements necessitate reevaluating security measures and developing innovative solutions to safeguard future networks against a broad spectrum of cyber threats.

Integrating AI and ML in network security offers promising proactive threat detection and mitigation avenues. However, the effectiveness of these AI/ML-based security solutions is contingent upon the availability of comprehensive and relevant datasets that accurately reflect the complexity and dynamism of modern networks. Additionally, the transition to 6G networks, characterized by their intelligent and highly interconnected nature, further amplifies these security concerns, necessitating advanced security mechanisms capable of addressing the sophisticated threats in these environments.

This literature survey delves into the current state of research on network security in the context of 5G and 6G technologies. It encompasses a review of eight pivotal papers exploring various aspects of network security, including developing comprehensive intrusion detection datasets, integrating AI for enhanced security measures, and exploring novel security challenges and solutions in the era of 5G and 6G networks.

II. Literature survey

The 5D-NIDD dataset underscores the complexity of securing 5G networks against sophisticated attacks. The dataset, developed in Oulu, Finland, aims to address the shortcomings of existing datasets by providing a rich source of real-world network scenarios, including various attack and benign traffic scenarios. This dataset is instrumental in applying ML techniques to develop proactive security measures against threats like Denial of Service (DoS) attacks and Port Scans, which are increasingly prevalent in 5G networks [1].

The transition towards 6G networks brings about a paradigm shift in network architecture, emphasizing the need for intelligent security mechanisms. The integration of AI into 6G security, as discussed in the second paper, offers a proactive approach to threat detection and mitigation. This paper highlights the importance of AI in achieving connected intelligence and outlines the security challenges inherent in 6G's denser deployments and reliance on AI. The proposed hierarchical security mechanisms and blockchain integration offer a blueprint for securing future networks [2]. Moreover, it provides a forward-looking perspective on 6G wireless systems and emphasizes the need for innovation in network design, application focus, and performance measurement. The recommendations for utilizing high frequencies, reimagining network architecture, and leveraging intelligent surfaces provide a comprehensive framework for developing 6G technologies [3]. The potential of Federated Learning (FL) as a defense mechanism against adversarial attacks in 5G networks. By proposing novel defense strategies and emphasizing the importance of

collaborative research, this paper contributes valuable insights into securing federated learning environments against sophisticated threats [4]. Also, focusing on the SliceSecure model addresses the unique challenges posed by network slicing in 5G architectures. The study uses deep learning techniques to offer a nuanced understanding of DDoS attacks within network slices. It proposes effective detection mechanisms, underscoring the importance of advanced intrusion detection systems in softwarized networks [5]. A comprehensive review of intrusion detection systems (IDS) highlights the evolution of detection methodologies in the face of sophisticated cyber-attacks. The discussion on the need for robust datasets and the exploration of evasion techniques offers a critical examination of the current landscape of IDS [6]. The empirical analysis compares the effectiveness of XGBoost against other machine learning classifiers. The findings underscore the efficiency and scalability of XGBoost in machine learning tasks, contributing to the broader discourse on applying ML techniques in cybersecurity [7]. Suppose we envision a 6G network architecture empowered by AI, detailing the strategic plans and technological innovations necessary for its realization. The emphasis on AI's role in enhancing situational awareness and enabling new service types highlights the transformative potential of integrating AI into wireless networks [8].

Conclusively, the surveyed literature underscores the critical role of AI and ML in addressing the security challenges of 5G and 6G networks. From developing comprehensive intrusion detection datasets to exploring AI-driven security mechanisms, the research efforts highlighted in this survey offer valuable insights and directions for future work in securing next-generation wireless networks. As we advance toward implementing 6G technologies, integrating intelligent security measures will be paramount in protecting against the evolving landscape of cyber threats. This literature survey, encompassing various research perspectives and methodologies, sets the stage for continued innovation and collaboration in network security.

III. Data processing

A. Data Preprocessing

The dataset 5G Network Intrusion Detection Dataset (5G-NIDD), generated over a 5G wireless network in Oulu, Finland, provides a valuable resource for identifying patterns and potential security vulnerabilities within 5G networks. The data cleaning process for the 5G-NIDD dataset involved removing all null and zero values, a critical step in preparing the data for in-depth correlation analysis. This step was essential to ensure the integrity and accuracy of the analysis, as null and zero values can significantly skew the results and lead to misleading conclusions. It improved the dataset's quality and laid a solid foundation for generating insightful and accurate findings for detecting network intrusions in 5G networks. This preparation was crucial for ensuring the dataset's integrity and accuracy, laying a solid foundation for generating insightful and accurate findings relevant to detecting network intrusions in 5G networks.

B. Feature Selection and Extraction

Feature Selection is a fundamental step in Machine Learning as it reduces the dimensionality of the data, improves the model's performance, enhances the interpretability as the model's complexity decreases, and reduces the training time. To achieve this, it is essential to identify the relevant features of the model and select the highly correlated features. We have mainly implemented two types of correlation coefficients: Pearson and Spearman. We can comprehensively analyze the relationships between various features within the dataset based on the Pearson and Spearman correlation coefficients obtained.

1) Pearson Correlation:: The Pearson correlation coefficient measures the linear correlation between two variables, giving a value between -1 and 1. A value closer to 1 implies a strong positive correlation, while a closer to -1 indicates a strong negative correlation. A value around 0 suggests no linear correlation. We analyzed the correlation between Seq (sequence number) and most other features, which is close to 0, indicating that sequential data points do not linearly predict the behavior of different features in the dataset. Moreover, a perfect correlation (1.0) among Dur, Runtime, Mean, and Sum indicates these features move together linearly. This might suggest redundancy among these features, which could be condensed or simplified in predictive modeling to improve efficiency without losing critical information.

TABLE I
HIGHLY CORRELATED COLUMNS IDENTIFIED BY PEARSON AND SPEARMAN CORRELATION

Spearman	Pearson	Uncommon Columns
RunTime	Mean	dMeanPktSz
DstBytes	TotPkts	Loss
Rate	Load	TcpRtt
dMeanPktSz	RunTime	AckDat
Loss	SrcRate	pLoss
TotBytes	Sum	SynAck
TotPkts	DstPlts	
TcpRtt	Dur	
DstPkts	Max	
Dur	SrcPkts	
Max	DstBytes	
SrcPkts	DstLoad	
SynAck	SrcLoad	
SrcBytes	Rate	
AckDat	SrcBytes	
Min	Min	
pLoss		
Mean		
Load		
SrcRate		
Sum		
DstRate		
SrcLoad		

2) Spearman Correlation:: The Spearman correlation assesses the monotonic relationship between two variables, which is more flexible than the Pearson correlation as it can capture relationships that are not strictly linear. The Spearman correlation coefficients highlight similar trends to the Pearson correlation, with most features exhibiting low to

moderate correlation. However, the Spearman correlation identifies stronger monotonic relationships, which could be either linear or nonlinear but consistently increasing or decreasing, whereas the Pearson correlation did not. This discrepancy can indicate the presence of nonlinear relationships among certain features. For example, Rate, SrcRate, and DstRate have significantly different Spearman coefficients than their Pearson counterparts, suggesting that their relationships might be more complex and nonlinear.

After analyzing the correlation among the columns, we have aimed to identify the columns in the 5G-NIDD dataset that are highly correlated with each other based on a specific correlation threshold. We have set the threshold to 0.9 for both Pearson and Spearman. As a result, Pearson identified 18 different columns, and Spearman recognized 24 different columns that are highly correlated (Table 1). Spearman's method identified more correlated pairs, suggesting that some relationships between variables in the provided dataset are monotonic but not linear. Spearman correlation can capture these relationships because it is based on rank values rather than the raw data, making it sensitive to monotonic trends, whether linear or not. The columns identified exclusively through Spearman but not Pearson ("Uncommon") exhibit strongly monotonic relationships but are not strictly linear. This indicates variability in how different features relate, suggesting that some features might have a nonlinear impact on others, which Pearson's correlation fails to capture due to its focus on linear relationships. The "Uncommon" columns (e.g., dMeanPktSz, Loss, TcpRtt, etc.) are fascinating because they were only highlighted by Spearman correlation. This suggests that these features have a robust and consistent relationship with at least one other feature in a nonlinear way. These features could be crucial in specific analysis scenarios, especially in understanding complex, nonlinear dynamics within 5G-NIDD.

After conducting correlation analysis with Pearson and Spearman coefficients, we further evaluated the importance of features using the Random Forest algorithm. This approach provides additional insights into the predictive power of features by measuring their impact on the model's accuracy. Random Forest, an ensemble learning method, offers a straightforward metric to evaluate feature importance. It does so by observing how random permutations of each feature affect the model's performance, thereby indicating the significance of each feature in predicting the target variable. We applied Random Forest to extract the top 10, 15, and 25 features based on their importance scores. This process corroborated some of our findings from the correlation analysis and revealed additional features that significantly influence model prediction yet might not have shown strong linear or monotonic relationships. Selecting features based on the importance of Random Forest allows for a more nuanced approach to reducing dimensionality. It ensures that the features included in the model encapsulate the most relevant information for predicting network intrusions in 5G networks, thus enhancing model performance and interpretability while reducing training time. By leveraging correlation analysis and the importance of the Random Forest feature, we have developed a robust feature selection process that combines the strengths of statistical analysis and machine learning algorithms. This comprehensive approach ensures that our model is built on a foundation of the most relevant and impactful features, optimizing its ability to detect network intrusions within the 5G-NIDD dataset.

TABLE II
RANKING OF TOP 10, 15, AND 25 FEATURES BASED ON RANDOM FOREST

Top 10 Features	Top 15 Features	Top 25 Features
Seq	Seq	Seq
Offset	Offset	Offset
sTtl	sTtl	sTtl
sHops	sHops	sHops
sMeanPktSz	sMeanPktSz	sMeanPktSz
SrcBytes	SrcBytes	SrcBytes
Proto_tcp	Proto_tcp	Proto_tcp
TotBytes	TotBytes	TotBytes
TcpRtt	TcpRtt	TcpRtt
State_REQ	State_REQ	State_REQ
	SyncAck	SyncAck
	AckDat	AckDat
	SrcRate	SrcRate
		SrcLoad
		Cause_Start
		Cause_Status
		Proto_udp
		State_ECO
		State_INT
		Load
		Rate
		Mean
		Min
		Dur
		DstPkts

C. Data Normalization

Data Normalization is essential as the input variables need to be scaled to a standard range without distorting differences between the ranges of values. This is particularly important because machine learning algorithms that use distance metrics (such as k-nearest neighbors (KNN), k-means clustering, or gradient descent algorithms like linear regression and neural networks) are sensitive to the magnitude of the data. Without normalization, a feature with a broader range could disproportionately influence the model, leading to biased results. We have incorporated Min-Max scaling, which transforms features to have a specific minimum and maximum, which can be crucial for models sensitive to data variance. We used (1) to calculate the min-max data normalization.

$$X_{\text{scaled}} = \frac{X - \min(X)}{\max(X) - \min(X)} \tag{1}$$

X is the original value, and Xscaled is the scaled value. Also, Xmin and Xmax are the maximum and maximum of the feature across all the data points. After applying the formula for each value of each feature to the Xscaled formula, we transform the original feature values with their scaled versions. This helps optimize the performance and enhances the effectiveness of the machine learning algorithms.

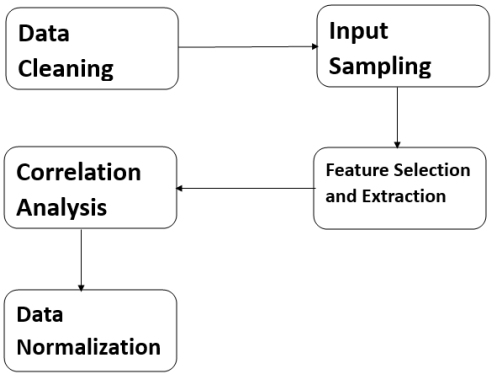

Fig. 1. Data Processing

D. File Description

IV. Analysis and Discussion

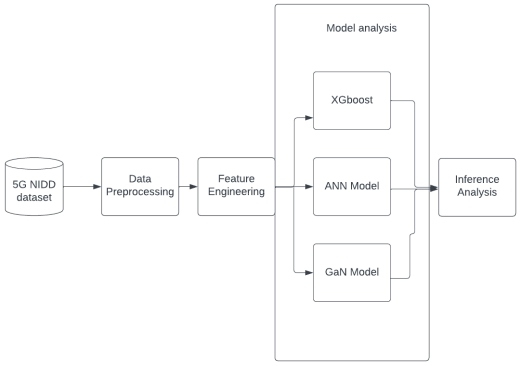

Fig. 2. Workflow of our ML models

A. Boosting

1) XGBoost: XGBoost, short for eXtreme Gradient Boosting, is a powerful and versatile boosting algorithm widely used in machine learning and data science applications. We found that xgboost had the second fastest training time and the best accuracy. However, we fear the xgboost model is overfitting.Despite being outperformed marginally in speed by a single competitor, XGBoost's robust performance solidified its position as a top contender in our evaluation. Its ability to handle large datasets with remarkable efficiency further underscores its suitability for real-world deployment. Our analysis corroborates XGBoost's reputation as a reliable and high-performing algorithm, offering practitioners a formidable tool for various predictive modeling tasks.

2) AdaBoost: We implemented the AdaBoostClassifier algorithm from the sci-kit-learn library for multi-class classification. It begins by preprocessing the data, including handling categorical variables using one-hot encoding and filling null values with zeros. The dataset is then split into training and testing sets. The AdaBoostClassifier is instantiated with 1000 estimators and trained on the training data. After training, predictions are made on the test set, and a classification report is generated to evaluate the model's performance, including precision, recall, and F1-score for each class. Additionally, the training time is recorded for performance analysis.

3) CatBoost: CatBoost stands out for its ability to handle categorical variables seamlessly without preprocessing, overfitting robustness, and GPU acceleration support. It's often used in various machine learning tasks, especially when dealing with tabular data containing categorical features. CatBoost is applied to a dataset containing network-related features for classification tasks. Initially, the data undergoes preprocessing steps, including dropping irrelevant columns, imputing missing values, and encoding categorical variables. The CatBoostClassifier is then initialized with parameters specifying the number of trees, learning rate, tree depth, loss function, and random seed. The model is trained on the preprocessed training data using the fit() method, and its performance is evaluated on the test data. CatBoost's ability to handle categorical features seamlessly and its robustness to overfitting make it well-suited for this classification task.

B. Artificial Neural Network

A fully connected artificial neural network has been implemented to model the data. The network architecture consists of an input layer, three hidden layers, and an output layer. The hidden layers are composed of densely connected neurons, allowing for complex nonlinear transformations and feature extraction. The rectified linear unit (ReLU) activation function is employed in the hidden layers to introduce non-linearity and enable the network to learn intricate patterns. The Adam optimizer, a state-of-the-art adaptive learning rate optimization algorithm, has been chosen to train the network efficiently. The Adam optimizer adaptively updates the learning rate for each weight based on the gradients, providing faster convergence and improved generalization performance compared to traditional stochastic gradient descent methods. Currently, the model is being trained using the specified architecture and optimizer to minimize the defined loss function and achieve accurate predictions on the target variable. More research is needed to find the correct number of hidden layers and to avoid overfitting. Another Artificial Neural Network (ANN) implementation uses TensorFlow and Keras for binary classification. It begins with data preprocessing steps, including handling missing values, normalization, and label encoding. The ANN architecture consists of four fully connected layers with ReLU activation in hidden layers and sigmoid activation in the output layer for binary classification. It employs a Stochastic Gradient Descent (SGD) optimizer with a learning rate of 0.01 and a Sparse Categorical Cross entropy loss function. The model is trained on the training data with a batch size of 32 for ten epochs, with a validation split of 10

C. GANBased Network

The generator network creates new data samples, such as images, by transforming random noise into realistic-looking

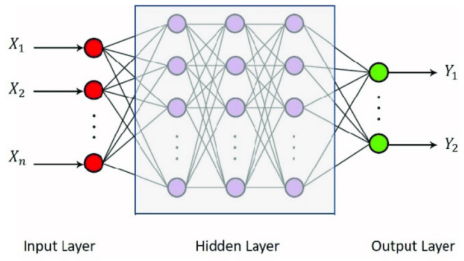

Fig. 3. Simple Ann Architecture

outputs. On the other hand, the discriminator network tries to distinguish between actual data samples and those generated by the generator. Through this adversarial process, both networks improve over time, with the generator learning to produce more convincing samples and the discriminator becoming better at distinguishing real from fake. GANs have been used for various applications, including image generation, data augmentation, style transfer, and even generating music and text. They have shown remarkable capabilities in producing high-quality, realistic outputs, sometimes indistinguishable from accurate data. We have applied the GAN to the processed code. The code demonstration is below.

1) Data Preprocessing:: The code starts by reading a CSV file containing the dataset. It drops columns with high missing values or is deemed irrelevant for classification. Missing values in certain columns are imputed using mean values or specific strategies. Label' 's target variable is encoded using label encoding, where 1 represents malicious, and 0 represents benign.

2) Feature Engineering:: Categorical variables are one-hot encoded to prepare the data for modeling.

3) GAN Architecture:: The GAN consists of a generator and a discriminator. The discriminator takes the real and generated data samples as input and tries to distinguish between them. The generator takes random noise as input and tries to generate synthetic data resembling real data. The GAN combines the generator and discriminator to train them simultaneously in an adversarial manner.

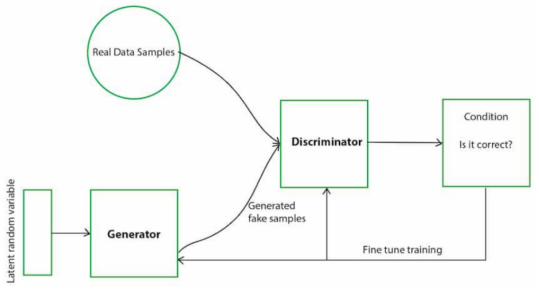

Fig. 4. Architecture of GAN-based model

4) Training the GAN:: The GAN is trained in a loop for several epochs. In each epoch, synthetic data is generated by the generator using random noise. Real data samples are randomly selected from the training dataset. The discriminator is trained using real and synthetic data, with labels indicating whether the data is real or fake. The generator is trained through the GAN model to generate data that can fool the discriminator into predicting it as real.

5) Generating Synthetic Data:: Synthetic data is generated using the trained generator after training the GAN. The number of synthetic samples generated is specified as num synthetic samples.

6) Downstream Classification Task:: The generated synthetic data is combined with the original training data. A CatBoost Classifier is trained on the combined dataset, including real and synthetic data. The classifier is trained to predict the binary labels (benign or malicious) based on the input features. The classifier's accuracy is evaluated on the test data after training.

7) Accuracy Evaluation:: The accuracy of the classifier on the synthetic data is computed using the test dataset. A confusion matrix is also generated to assess the classifier's performance regarding true positives, false positives, and false negatives. We used GAN to generate synthetic data to augment the training dataset for a downstream classification task. The classifier's accuracy on the synthetic data provides insights into the effectiveness of GAN-generated data for improving the performance of machine learning models.

TABLE III
COMPARISON OF MACHINE LEARNING MODELS FOR ENCODED DATA

Model	Accuracy	Training Time(s)	Prediction Time(s)
Decision Tree	99.90%	4.32	0.03
Random Forest	99.95%	8.76	0.22
KNN	99.87%	2.21	338.52
Naive Bayes	96.35%	1.24	0.06
MLP	99.85%	150.20	0.26
XGBoost	100%	50.26	1.24
AdaBoost	99.56%	212.34	1.7
CatBoost	99.98%	256.52	0.39
ANN (PyTorch)	61%	5185.08	0.49
ANN (TensorFlow)	63%	1853.67	0.54
GAN	99.97%	75.77	0.16

TABLE IV
COMPARISON OF MACHINE LEARNING MODELS USING PEARSON CORRELATION

Model	Accuracy	Training Time	Prediction Time
XGBoost			
AdaBoost			
CatBoost			
ANN (PyTorch)			
ANN (TensorFlow)			
GAN			

REFERENCES

TABLE V
COMPARISON OF MACHINE LEARNING MODELS USING SPEARMAN CORRELATION

Model	Accuracy	Training Time	Prediction Time
XGBoost			
AdaBoost			
CatBoost			
ANN (PyTorch)			
ANN (TensorFlow)			
GAN			