Enhanced Cyber Threat Detection System Leveraging Machine Learning Using Data Augmentation

Umar Iftikhar, Syed Abbas Ali
Department of Computer & Information System Engineering, NED University of Engineering & Technology, Karachi, Pakistan

***Abstract*—In the modern era of cyber security, cyber-attacks are continuously evolving in terms of complexity and frequency. In this context, organizations need to enhance Network Intrusion Detection Systems (NIDS) for anomaly detection. Although the existing Machine Learning models are in place to cater to the situations but new challenges emerge rapidly which affects the performance and efficiency of existing models specifically the unreachability of large datasets and unorganized data. This results in degraded efficiency for the identification of complex attacks. In this paper, data augmentation has been done of NSL-KDD which is a standard dataset for Intrusion Detection Systems (IDS) specifically for IoT-based devices. The improvement in performance and efficiency of NIDS has been performed by training the augmented dataset using the K-Nearest Neighbor (KNN) ML model.**

Keywords*—Anomaly detection; cyber threat intelligence; generative adversarial networks; data augmentation; Wasserstein GAN with gradient penalty

I. INTRODUCTION

This section gives some background regarding the research presented in this paper along with a detailed problem statement followed by the objectives.

A. Background

The domain of Network security has become one of the most immensely important domain in the modern technological landscape because of various challenging threats, numerously increasing the risk factors. In this context, continuous counter-security measures have become the utmost need of modern-day technology that declares war against intruders and stabilizes the computer networks that are responsible for the protection of user information without compromising the entire network. Network Security has provided significant benefits with time in terms of various aspects like evaluation of security problems, practicing attack and defense, and enhancement of confrontation of network information. Network security plays a vital role and always needs room for improvement in terms of the efficiency and accuracy of intrusion detection techniques [10]. The use of wireless technology and the transmission of large amount of information over the networks has significantly increased security issues specifically in the emerging field of IoT. Network security has become the main point of concern and an effective and efficient Intrusion Detection System (IDS) has now become an essential need for traditional and IoT networks that provide countermeasures against rising threats in small and medium enterprises.

Because of these reasons, Intrusion Detection Prevention Systems (IDPS) have become crucial for network security as they play a vital role in tackling threats. IDPS have become more powerful than ever ensuring the security of networks.

B. Problem Statement

The modern-day NIDS has various crucial shortcomings and drawbacks that degrade its ability to effectively handle cyber-attacks. Majorly it lacks in authenticity and timely responsiveness of network threats in existing datasets that are used for training of NIDS. The frequent staling of these datasets is the main reason for identifying new threats as IDS models mainly rely on these datasets.

Another major limitation regarding the current deployment of NIDS is expensive middleware applications that only monitor a certain portion of the entire network, whereas neglects monitoring of other segments. Data augmentation methodologies play a pivotal role in the training of machine learning models. On the other hand, various existing research has claimed the enhancement in the adaptability of models but still, the utilization of these augmentation techniques in cyber security poses problems.

C. Objectives

The main objective of this research is to produce a vigorous cyber threat detection system equipped with the tendency of auto-encoder based data synthesis and attack surface vector modeling [11]. Generative Adversarial Networks (GANs) [2] have the ability to enhance the attack detection performance of NIDS when datasets like NSL-KDD are incorporated into it. Utilization of GANs to produce the synthetic data, NSL-KDD can be extended using more assorted and accurate examples of network traffic that enhance the quality of trained models of the provided datasets.

To authenticate the designed methodology, the NSL-KDD dataset has been selected as a vigorous baseline for this research. The dataset chosen has been proven globally in the field of networks due to its rational evaluation, reproducibility, and trustworthy capability of accomplishing theoretical amendments. The results generated by the selection of the above-mentioned dataset impact a concrete transferable theory that is not affected by dataset-specific limitations. Thus, the framework proposed in this research is more capable of being implemented on complex datasets resulting in the evolution of various network traffic scenarios.

The idea of using generative model approaches and the NSL-KDD in combination will enhance the capability to cater the challenges such as data misbalancing and overfitting in IDS. Training of augmented datasets with denser samples, the performance can be enhanced against cyber-attacks, thus enhancing the model's efficiency in detecting network threats.

II. Related Work

In this section, all the related work was discussed in detail including NSL-KDD Dataset, various ML Models, GAN for augmentation of data, KNN Model, and gaps in existing research.

A. *Overview of NSL-KDD Dataset*

The NSL-KDD dataset is an enhanced version of the KDD Cup 1999 dataset that was developed to address the problems like redundancy and class imbalance [12]. It is capable of providing improved and reliable standards for evaluating the performance of Network Intrusion Detection Systems (NIDS). The dataset in KDD Cup 1999 consists of a large number of duplicate instances that create noise while training the machine learning models [14]. Due to this reason, NSL-KDD has an edit advantage in which the majority of duplicated records are considered redundant and are eliminated. This elimination of duplicate records provides more enhanced and refined data of network traffic, improving the capability to assess the performance of NIDS.

There are 41 features in the NSL-KDD that are capable enough to capture the characteristics of network traffic. These characteristics include connection details, timing, transferred bytes, and payload content. These characteristics are further divided into structural, content, time-based, and host-based groups that help in attaining improved statistical and machine-learning methodologies used for intrusion detection. The enhanced features of KDD-NSL such as addressing redundancy and balanced class distribution, improve the capabilities to detect common and rare attacks in NIDS [3][4]. Vast research has been carried out using this dataset for the development and testing of new algorithms, which further makes this dataset more relevant in the field of network security and intrusion detection.

B. *Machine Learning Models for Network Security*

Machine Learning Models are now playing a vital role in the domain of network security as they provide capabilities like detection of more unusual activities, attacks, and adjustments of cybersecurity systems. There are a large number of models for anomaly detection, some of the most popular used models are Auto encoders, Support Vector Machines (SVM), K-Nearest Neighbor (KNN), Random Forest Classifiers, Convolutional Neural Networks (CNN), Deep Neural Networks (DNN), Generative Adversarial Networks (GANs).

Training the Wasserstein Generative Adversarial Network with Gradient Penalty (WGAN-GP) data augmentation [9] with the KNN model gives an enhanced result in the detection of cyber-attacks. The NSL-KDD dataset is being considered as the standard dataset for intrusion detection, sometimes agonizes from class imbalance. WGAN-GP, being a benchmark generative model addresses this issue by creating high-quality synthetic samples for diminished classes. This augmentation improves the diversity of the dataset, guaranteeing the training of ML models on illustrative data. The enhanced features of KNN such as instance classification depending upon neighborhood patterns make a perfect technique for this augmented dataset.

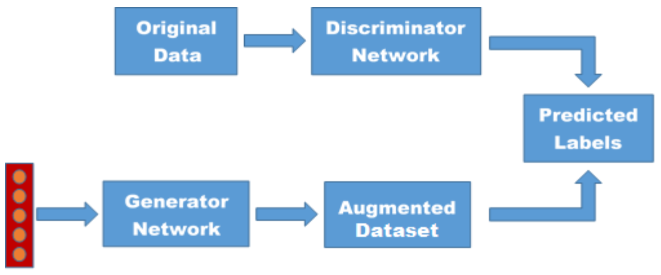

Fig. 1. Workflow of GAN.

The essence of this approach enhances its tendency to meet high accuracy and robustness in defending against cyber-attacks. WGAN-GP produces the data with very less overfittings, sustaining the integrity of the distribution, which is vital in identifying attacks. The flexibility and interoperability of KNN ensure detection in diversified attack scenarios. Combining the WGAN-GP augmented dataset with the KNN model, the system becomes capable of addressing limitations of traditional detection methods like bad generalization and imbalance bias, which leads to a more enhanced cyber-attack detection system for real-world applications.

Generative Adversarial Network (GAN) is one of the most proven model for generative learning that is divided into three main areas which are deep learning, generative models, and adversarial learning techniques [8]. GANs efficiently utilize deep learning networks for solving complex patterns.

The objective function of the original WGAN is given in Eq. (1).

$$\boldsymbol{\mathcal{L}_{WGAN}} = \mathbb{E}_{\boldsymbol{x\sim pdata}}[\boldsymbol{D(x)}] - \mathbb{E}_{\boldsymbol{z\sim pz}}[\boldsymbol{D(G(z))} \quad (1)$$

where,

x is a sample from the real data distribution pdata,

z is a noise vector sampled from a prior distribution pz,

D(x) is the output of the critic for a real sample,

D(G(z)) is the output of the critic for a generated sample.

The Gradient Penalty of WGAN-GP can be mathematically defined in Eq. (2).

$$\boldsymbol{\mathcal{L}_{GP}} = \mathbb{E}_{\boldsymbol{x\sim pz}}[(||\boldsymbol{\nabla_x D(\hat{x})}\,||_{\boldsymbol{2}} - \boldsymbol{1^2}] \quad (2)$$

where,

$\hat{x}$ is a random sample,

$\nabla_x D(\hat{x})$ is the gradient of the critic D concerning the input,

$||\nabla_x D(\hat{x})\,||_2$ is the norm of gradient.

The WGAN-GP loss function is expressed in Eq. (3).

$$\boldsymbol{L} = \mathbb{E}[\boldsymbol{D(x)}] - \mathbb{E}[\boldsymbol{D(G(z))}] + \boldsymbol{\lambda}\mathbb{E}[\left(\left|\left|\boldsymbol{\nabla D(\hat{x})}\right|\right|_{\boldsymbol{2}} - \boldsymbol{1}\right)^{\boldsymbol{2}} \quad (3)$$

Eq. (3) is comprised of two parts; in the first part, the critical loss is referred while in the second part is related to the WGAN gradient penalty. This ensures maintaining stability while training. The diagram in Fig. 1 explains the architecture of a WGAN-GP [13], presenting the generator and discriminator's roles in the production of high-quality synthetic data while differentiating it from real data.

C. *Generative Adversarial Network in Data Augmentation*

When it comes to data augmentation, GANs are proven to be the most relevant technique that increases the range and volume of data serving the machine learning model.

GANs can produce suitable data distribution over the original dataset. Also, it can compensate the unusual situations where data is limited or unbalanced. Synthetic data, refers to different samples that are not present in data, for example, other viewpoint images in image data or samples of some neglected categories in categorical data. This permits to enhancement of the dataset by inculcating realistic synthetic samples and GANs in addition improves the overall performance of machine learning models and provides a better ability for generalization of data as illustrated in Fig. 1. GAN usage for augmentation of data is extensive and produced various advancements in multiple fields by synthetic data creation [6][7]. They can also be deployed in the analysis of network traffic, where they can be utilized in producing synthetic data that aids in creating enhanced models and is capable of identifying abnormalities in network systems.

D. *K-Nearest Neighbor (KNN)*

K-Nearest Neighbors (KNN) is a non-parametric, controlled learning algorithm that is popular for its classification and regression tasks. This algorithm functions by detecting the ‘k’ nearest data points known as neighbors within the featured space over a given input point. Also, it provides predictions either based on the majority class for classification or the average of their values for regression. Euclidean, Manhattan, or Minkowski distance metrics are being used to calculate the distance between two points.

KNN is specifically beneficial in the detection of abnormality as it tends to detect data points that significantly diverge from their nearest neighbors. Abnormalities usually appear as data points with lesser or more detached neighbors as compared with normal data points. KNNs are very useful in multidimensional data handling, implementation simplicity, and flexibility.

The mathematical equation of KNN can be given in Eq. (4):

$$\boldsymbol{d(x,y)=\sqrt{\sum_{i=1}^{n}(x_i-y_i)^2}} \tag{4}$$

where,

d is the distance between two points **x** and **y** in an n-dimensional space.

The formula for calculating the anomaly score is given in Eq. (5).

$$\boldsymbol{Anomaly\ Score(x)=d\left(x,x_{(k)}\right)} \tag{5}$$

where,

$x_{(k)}$ is the kth nearest neighbor of (x).

E. *Gaps in Existing Research*

In the area of GANs, a wide research gap is still there. The research performed in this paper will try to cover these areas. The first and most important part is augmentation of data using GAN. The majority of studies on GAN focused on GAN in a general manner, rather than emphasizing the functionality. This research has been carried out with a focus on how GANs can be utilized extensively to handle new types of data.

The methodology proposed in this paper focuses on practical ability by dimensionality reduction with WGAN-GP and Auto-encoder. The performed research ensures uninterrupted translation to the operational environment. Also, it addresses the real-world scenarios for network traffic data in various ways. This research will result in providing effective ways to process the real-world data, with enhanced features like reduced dimensionality, and augmentation of data, as, the WGAN-GP’s framework is specifically trained to produce realistic network traffic.

Observations show continuous improvements in the performance with the proposed methodology that is almost similar to real-world scenarios. Various network environments can be adaptable to proposed design of WGAN-GP as it allows retraining the generator with domain-specific traffic data. For example, this design can be deployed on IoT and cloud-based infrastructures by modifying the training process for reflection of their unique characteristics.

III. Methodology

This section describes the complete methodology applied to the research.

A. *Data Description*

The dataset used for experimentation is the NSL-KDD dataset. It contains 84,952 entries with 28,318 reserved entries for validation.

While considering the features and labels, the dataset contains similar properties. Every single record in the dataset has several parameters depicting multiple aspects of the network traffic. These features provide help in specific identification of the attacks in the behavior of the network. Preprocessing of data has been performed to identify the classification task to the binary decision [1] that is either “normal” or “attack”. Due to this binary transformation, intrusion detection has been more simplified specifying whether the ongoing activity is malicious or not.

B. *Data Preprocessing*

The preprocessing of data is the most important part as it transforms the data to be used by machine learning algorithms. Other processes that are associated with data preprocessing are data collection, data sanitization, and data normalization. The details of the process include data loading, data splitting, data validation, data testing, handling of numerical and categorical features, and reduction of dimensionality using auto-encoders. The data preprocessing process is further divided into the following steps.

The first step is to import the NSL-KDD dataset, which is divided into two parts. One is referred to as the original dataset containing 84,952 entries. The other one is referred to as a reserved dataset containing 28,318 entries. The remaining entries are referred to as test datasets for evaluating the model's performance on new data. This technique empowers our research to minimize overfitting effects to train models for any unseen instances.

The next step is scaling the features for numeric features and one-hot encoding for categorical features. In numeric features, the median strategy was used for imputing missing values. This was done to refrain from the mean value of numeric data so that it does not bend towards outliers. In categorical features, a one-hot encoder class was used for each feature. The preprocessing technique was used to assign the use of a column transformer which introduces automation of their application to training and test datasets.

The encoding dimension is constant throughout the experiment, to preserve the true spirit of feature representation and analyze the behavior of varying factors. Out of 100%, a 30% augmentation level has been represented to create a balance between sufficient variability and training data, preserving the integrity of the original labels. Also, diverse representations will be learned by the model without being overwhelmed with noise at 30%. This will further enhance the generalization capabilities. The quality of the synthetic data is evaluated using the Classification Model Utility approach. This method involves training a model on the original dataset and another on the combined original + synthetic dataset, then testing both models on the same validation dataset. Improved performance on the validation dataset indicates the synthetic data's quality and utility [5]. The augmentation level increased to 50% enhances variability without introducing out-of-distribution issues.

The next step is the generation of data. It comprises three phases that include optimization of the constructed model, new synthesized data was produced using WGAN-GP.

The next step is the setup of the WGAN-GP model. This setup comprises two major components, the generator and the discriminator. The generator produces data that resembles with input dataset. The discriminator determines whether the generated data is fake or real. While training the WGAN-GP, numerous iterations were performed through many epochs. The generated data was still transformed and in scaled format, it was necessary to untransformed it to bring it back into its original feature space.

The final step is the merging of synthesized data with existing data.

The hyper-parameters and their functionality are as follows:

- Learning Rate: Controls the weight update per iteration, enhancing the speed and stability of training.
- Num Epochs: Determines the count for seeing the entire dataset by the model of the training process.
- Critic Iterations: Number of updates for the critic (discriminator) per generator update, affecting model stability.
- Lambda GP: It is the coefficient of gradient penalty that ensures that makes sure normalization of the gradient to stabilize the training.

The deviation in hyper-parameters shows a major difference in WGAN-GP performance. Table I depicts the response of the model over the selected algorithm and parameter. The fluctuated parameters during the experiment are shown in Table I.

TABLE I. VARIATION IN HYPER-PARAMETER

Instance	**Hyper-Parameter Configuration**		
	Learning Rate	***Num of Epochs***	***WGAN Augment %***
X1	2.00E-04	100	30%
X2	1.00E-04	100	30%
X3	5.00E-05	100	30%
X4	2.00E-04	50	30%
X5	1.00E-04	50	30%
X6	5.00E-05	50	30%

C. Performance Metrics

The parameters applied on WGAN-GP for examining the performance are as follows:

1) Accuracy (ACC): It is considered the most important parameter in evaluating the model's performance. This metric evaluates the number of samples for correct prediction over the number of all samples. The formula for calculating this metric is given in Eq. (6).

$$\boldsymbol{ACC = \frac{TP+TN}{TP+TN+FP+FN}} \tag{6}$$

2) Recall: This parameter refers to the ability of the model to predict positive samples. This is calculated by dividing the number of samples that are categorized as true positive overall positive samples. The formula for calculating this metric is given in Eq. (7).

$$\boldsymbol{Recall = \frac{TP}{TP+FN}} \tag{7}$$

3) Precision: In this parameter, true positive identified the number of samples over several samples that are predicted as positive. Eq. (8) calculates the precision.

$$\boldsymbol{Precision = \frac{TP}{TP+FP}} \tag{8}$$

4) F1_score: In this parameter, the recall and precision are combined into a single metric. This is called the harmonic mean of recall and precision. Eq. (9) calculates the F1_score.

$$\boldsymbol{F1\_score = \frac{2 \times Precision \times Recall}{Precision+Recall}} \tag{9}$$

5) Matthews correlation coefficient (MCC): This performance parameter is considered the best metric for binary classification. It combines all parts of the confusion matrix. The equation for calculating this metric is given in Eq. (10).

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP+FP)\times(TP+FN)\times(TN+FP)\times(TN+FN)}} \quad (10)$$

D. Model Training

While training the models, various ML Models have been considered on encoded information generated from auto-encoder. K-Nearest Neighbor (KNN) has been shortlisted from various models. The parameters for selecting this model were its applicability in classifying the tasks.

Algorithm 1: Machine Learning Model for Predictions

```
Initialize (train_set, validate_set)
Step 1: Import K-Nearest Neighbor (KNN) instances from the scikit-learn library
Step 2:
        If (train_set: == X_train_encoded) then
            Train model on X_Train_Encoded
        Else
            Train model on X_Train_Augmented
        End
Step 3: Predict the imported models on Validate_Set
Step 4: Evaluate the models on the following performance metrics
        a.  Accuracy
        b.  Precision
        c.  Recall
        d.  F1 score
        e.  MCC
Step 5: Export the performance metrics in the CSV file
End
```

The machine learning model can be categorized into two stages. First is training of the model and second is validation of the model. The step-by-step processing of the research algorithm is presented in Algorithm 1. This algorithm gives a method for comparing the accuracy of the models and various parameters for evaluation using the encoded validation dataset to assess the possibility of identifying cyber-attacks.

K-Nearest Neighbor (KNN) ML model is used in this research. KKN Algorithm is known for its simplicity and instance-based machine learning which makes it suitable for tasks like classification and regression. This algorithm operates in a manner that compares a new data point with its nearest neighbors in the featured space. The "K" is the number of considerable neighbors. The label of the new point is determined by classifying the most common class from corresponding K neighbors. Regression is achieved by averaging the neighbors' values that are used to predict the output.

Another reason for using KNN was its non-parametric feature. In this feature, assumptions were not made regarding data distribution. The other features of this model like simplicity and effectiveness for handling non-linear relationships make this algorithm the most suitable choice for tasks like pattern recognition, recommendation systems, and anomaly detection. However, the performance of KNN depends on the selection of K and the distance metric (e.g., Euclidean, Manhattan), along with the size and quality of the dataset.

E. WGAN-GP Augmented Data Training and Validation

The blending of WGAN-GP synthesized data with the NSL-KDD dataset [30], poses a great impact on the performance of the model. The model training starts with the preprocessing of raw datasets, which is the most important process that ensures that the data is ready for model training. Also, further processes like imputation, scaling, and encoding were performed on the augmented data.

After the completion of preprocessing on augmented data, the next step was to update of machine learning model. The subset of the model used in the training process is replicated for the update process. The training of the model was performed on the same metrics used in the dataset training process. Those are accuracy, precision, recall, F1-score, and Mathews Correlation Coefficient (MCC). Using these metrics, performance was assessed after training the model.

Once the KNN model training is completed, the next step is to apply the trained model to encoded validation data. The metrics for analyzing the performance are the same as in previous steps. The use of the validation model trained from the WGAN-GP augmented dataset is the most important factor in assessing the generalization capabilities of the trained model from the augmented datasets.

By observing the performance of the model on the validation dataset, the quality of the synthetic dataset was accounted for.

IV. Experimental Results and Discussion

Table II shows the performance evaluation of the trained model with original data and augmented data. The datasets were evaluated on five performance metrics as described in Algorithm 1 step 4. The values of performance parameters present significant and effective results. These results can be used to draw a meaningful conclusion.

On the original dataset, the provided metrics reflect the mixed performance of the model. 64.77% of accuracy specifies the classification of the instances in the model is an average of two-thirds. Nevertheless, considering accuracy alone can mislead, especially while dealing with imbalanced datasets. The precision value of 0.7376 depicts that the prediction of a positive outcome by the model is 73.76% correct, resulting in a reduced false positive rate. As far as recall is considered, the 0.6477 value highlights the capturing of actual positive cases by the model up to 64.77%, leaving a significant number of false negatives. The F1 score, which provides a balanced measure between precision and recall, with 0.5594, reflects an imbalance relation between these two metrics. This reflects the struggle of the model to maintain an optimal trade-off between precision and recall. Moreover, MCC which covers all aspects of the confusion matrix, is 0.26467. This depicts the model has limitations in prediction, performing slightly better than random guessing.

With an augmented dataset, the performance metrics show variable performance for various hyper-parameter configurations. These performances are presented in the form of graphs in Fig. 2 to 6. Instance X1 provides average recall at 59.11%, but looking at precision (45.12%) and F1 score (44.69%) gives a high false positive rate, whereas negative MCC (-0.0328) means that the performance of the model is overall weak. On the other hand, instance X2 results as the most improved, enhanced, and efficient model. The highest level of accuracy (79.74%), precision (0.8352), recall (0.7974), and F1 score (0.7988) having the strongest MCC (0.6297), depicts the overall best performance providing balanced and reliable predictions. X3 performance is considered to be an average performance having an accuracy of 60.21%, precision at 0.5895, and recall at 0.6021. However, lower F1_score (0.4886) and MCC (0.0722) show that there is room for improvement.

By analyzing the provided parameters of X4, the performance of the model seems to be poor with an accuracy of 38.21%, precision at 0.3656, and recall at 0.3821. The results of F1_score (0.3728) and negative MCC (-0.3152), indicate the predictions very near to random guessing. The worst performance results can be seen in the X5 instance with very low accuracy (13.09%), precision of 0.1489, and recall (0.1309). The F1_score (0.1373) and highly negative MCC (-0.7379) show extremely unreliable predictions. The X6 instance results same as X1 depicting average recall (0.5899), low precision (0.4625), F1 score of 0.4485, and nearly zero MCC (-0.0321), indicating poor predictive power.

In a nutshell, instance X2 is considered to be the most significant as compared to other hyper-parameter configurations providing a strong balance between precision and recall. X1 and X6 can be considered as average while, X4 and X5 require major improvements.

TABLE II. OVERALL PERFORMANCE OF DIFFERENT HYPER-PARAMETERS WITH KNN AS A CLASSIFIER

Instance	Data Model	Hyper-Parameter Configuration				
		Accuracy	Precision	Recall	F1 Score	MCC
Original Dataset	Original Dataset	64.77%	0.737635	0.647655	0.559395	0.264674
X1	Augmented Dataset	59.11%	0.451186	0.591118	0.446918	-0.03282
X2		79.74%	0.835227	0.797432	0.798767	0.62967
X3		60.21%	0.589498	0.60214	0.488586	0.072174
X4		38.21%	0.365578	0.382094	0.372804	-0.31518
X5		13.09%	0.148879	0.130908	0.137281	-0.73789
X6		58.99%	0.462506	0.589905	0.44854	-0.03214

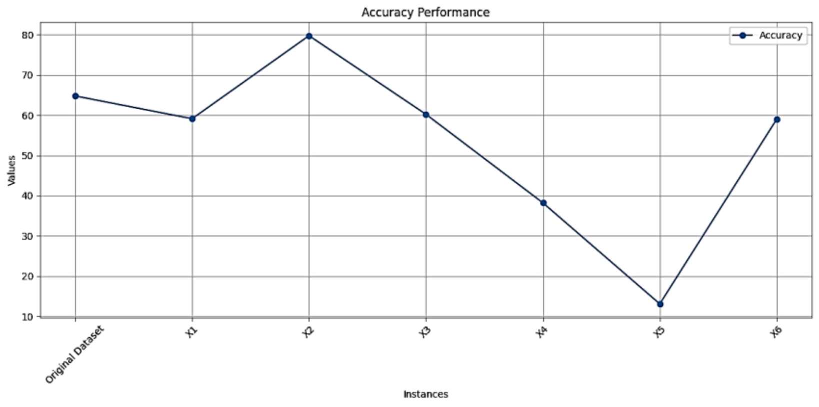

Fig. 2. Accuracy performance.

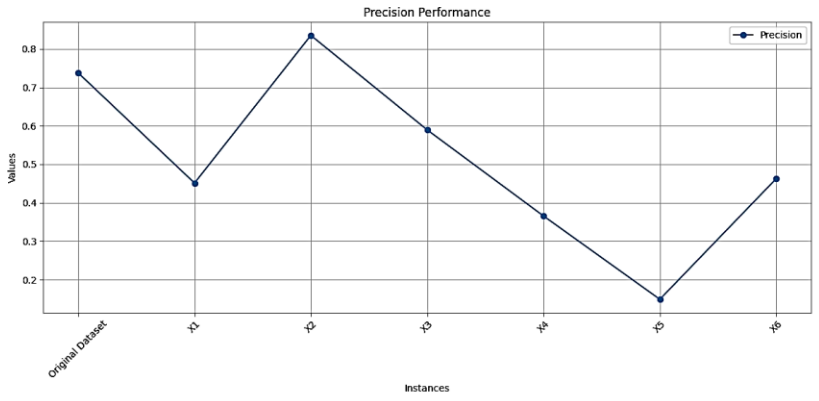

Fig. 3. Precision performance.

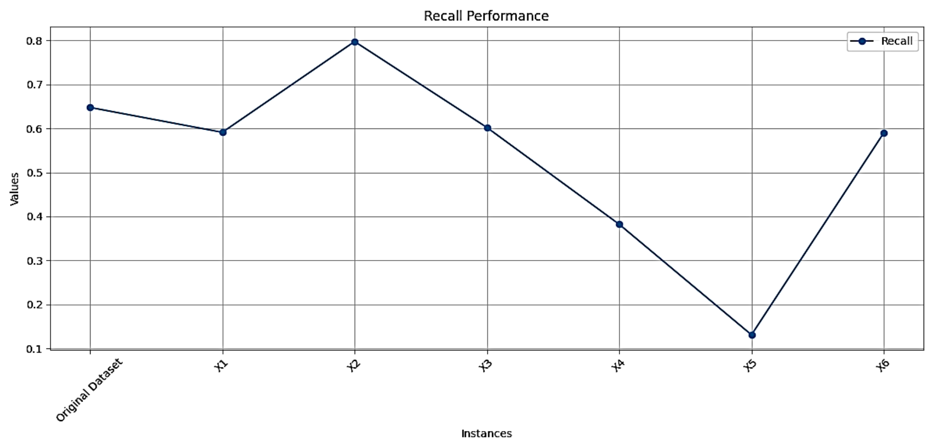

Fig. 4. Recall performance.

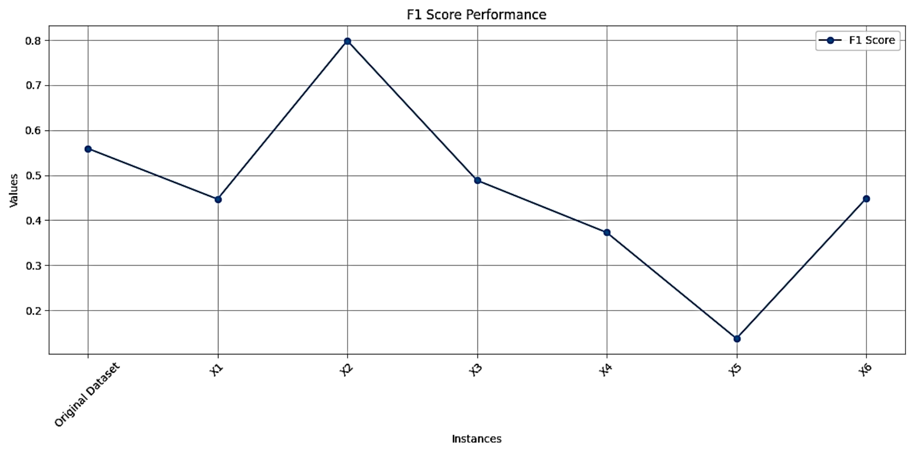

Fig. 5. F1_score performance.

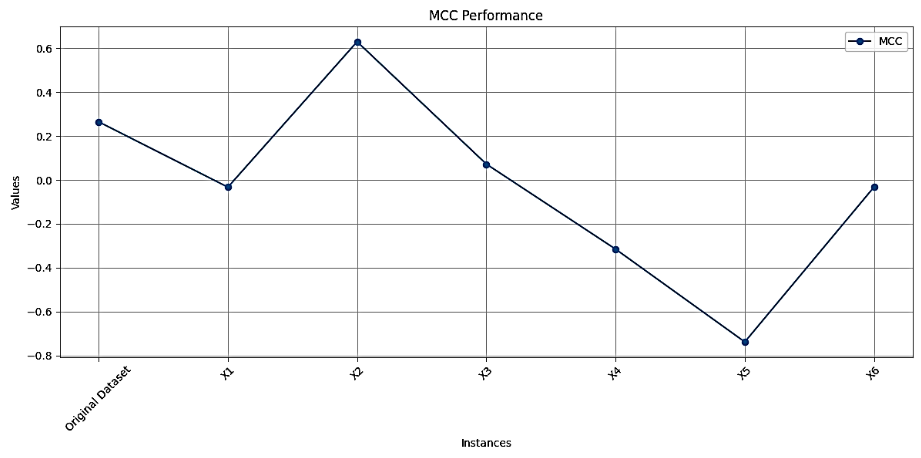

Fig. 6. MCC performance.

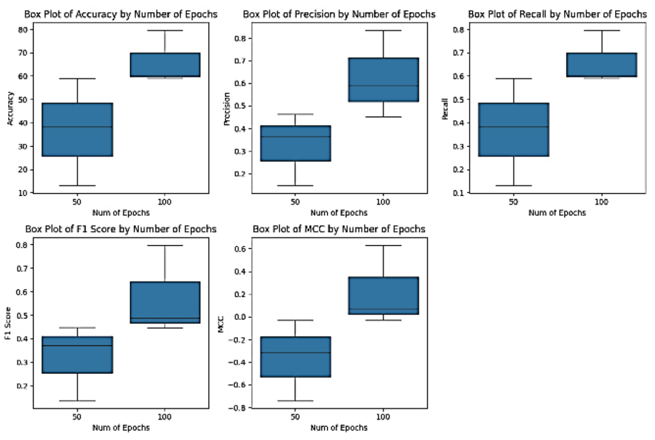

Fig. 7. Correlation heatmap between epoch and hyperparameters.

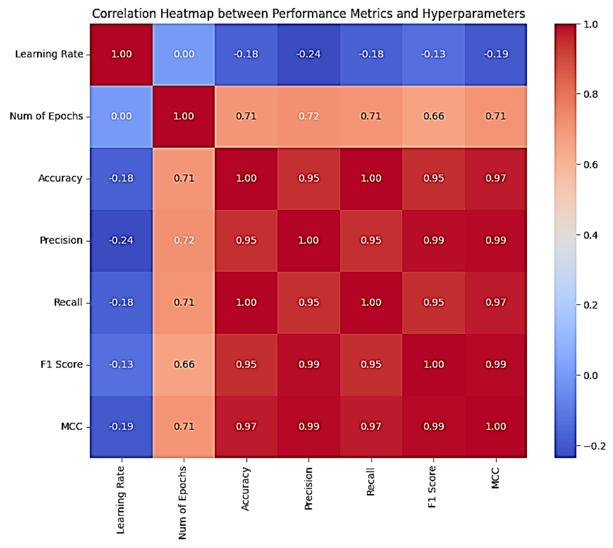

Fig. 8. Correlation heatmap between performance metrics and hyperparameters.

V. Conclusion

The performance analysis of KNN using various hyper-parameter configurations shows unique trends in deciding which configuration is suitable for enhanced performance of machine learning classifier. KNN was found to be more sensitive in selecting hyper-parameters, resulting in great variability, with some configurations resulting in poor performance. Dataset augmentation improves performance in general. However, it is essential to how hyper-parameter values are combined. The optimistic combination of hyper-parameter values for data augmentation will leverage the performance of the machine learning algorithm. Also, from the graphs shown in Fig. 2 to 6, it can be concluded that the increment in epoch value increases the chance of performance as compared to lower epoch values. By analyzing the heat map correlation matrix in Fig. 7 and 8, a value of 0.7 implies a strong positive correlation between the two variables. This concludes that variables have a direct relationship: as one variable increases, the other tends to increase as well.

The research in this paper presents a focused approach to enhancing the capabilities of Network Intrusion Detection Systems (NIDS) through improvement in learning capacity using advanced generative methods like WGAN-GP for augmentation of the dataset. In addition, resource-constrained environments can be considered for future experiments that will help in determining the practical implementation of these techniques in real-time IoT-based applications. Future enhancement of this research can be done by comparing different augmentation techniques on diversified datasets that will enhance the adaptability and robustness of NIDS.

References

[1] C. Strickland, “DRL-GAN: A hybrid approach for binary and multiclass network intrusion detection,” Sensors, vol. 24, no. 9, p. 2746, 2024.

[2] A. Mari, D. Zinca, and V. Dobrota, “Development of a machine-learning intrusion detection system and testing of its performance using a generative adversarial network,” Sensors, vol. 23, no. 3, p. 1315, 2023.

[3] M. Arafah, “Evaluating the impact of generative adversarial models on the performance of anomaly intrusion detection,” IET Networks, vol. 13, no. 1, pp. 28–44, 2023.

[4] E. Goud, “Enhancing DDoS attack detection in SDNs with GAN-based imbalanced data augmentation,” International Journal on Recent and Innovation Trends in Computing and Communication, vol. 11, no. 9, pp. 541–551, 2023.

[5] Z. Wang, C. Han, W. Bao, and H. Ji, “Understanding the effect of data augmentation on knowledge distillation,” arXiv preprint, arXiv:2305.12565, 2023.

[6] S. Bourou, A. Saer, T. Velivassaki, A. Voulkidis, and T. Zahariadis, “A review of tabular data synthesis using GANs on an IDS dataset,” Information, vol. 12, no. 9, p. 375, 2021.

[7] Y. Sun, M. Li, L. Li, H. Shao, and Y. Sun, “Cost-sensitive classification for evolving data streams with concept drift and class imbalance,” Computational Intelligence and Neuroscience, vol. 2021, 2021.

[8] A. Aggarwal, M. Mittal, and G. Battineni, “Generative adversarial network: An overview of theory and applications,” International Journal of Information Management Data Insights, vol. 1, no. 1, p. 100004, 2021.

[9] A. Shafee, M. Baza, D. A. Talbert, M. M. Fouda, M. Nabil, and M. Mahmoud, “Mimic learning to generate a shareable network intrusion detection model,” in 2020 IEEE 17th Annual Consumer Communications & Networking Conference (CCNC), IEEE, 2020, pp. 1–6.

[10] R. Vinayakumar, M. Alazab, K. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, “Deep learning approach for intelligent intrusion detection system,” IEEE Access, vol. 7, pp. 41525–41550, 2019.

[11] L. Xu, M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni, “Modeling tabular data using conditional GAN,” in Advances in Neural Information Processing Systems, 2019, pp. 7335–7345.

[12] J. Lee and K. Park, “GAN-based imbalanced data intrusion detection system,” Personal and Ubiquitous Computing, pp. 1–8, 2019.

[13] R. Abdulhammed, M. Faezipour, A. Abuzneid, and A. AbuMallouh, “Deep and machine learning approaches for anomaly-based intrusion detection of imbalanced network traffic,” IEEE Sensors Letters, vol. 3, no. 1, pp. 1–4, 2018.

[14] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, “A detailed analysis of the KDD CUP 99 data set,” in Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA), Ottawa, ON, Canada, 2009, pp. 1–6.