Scalable Network Intrusion Detection via Advanced Feature Engineering and Correlation Analysis on the CICIDS2017 Dataset

Ali Mohsin
Department of Computer Science
National University of Computer and Emerging Sciences, FAST
Lahore, Pakistan
ali787ch@gmail.com

Muhammad Zunurain Hussain
Department of Computer Science
Bahria University Lahore Campus
Lahore,Pakistan
zunnurain.bulc@bahria.edu.pk

Kabeer Ahmad
Department of Computer Science
National University of Computer and Emerging Sciences, FAST
Lahore, Pakistan
kabeeahmad.ka@gmail.com

Muhammad Zulkifl Hasan
Faculty of Information Technology
University of Central Punjab
Lahore,Pakistan
zulkifl.hasan@ucp.edu.pk

Fatima Ahmad
Department of Computer Science
National University of Computer and Emerging Sciences, FAST
Lahore, Pakistan
fatimahmad2222@gmail.com

Iqra Azam
Department of Computer Science
National University of Computer and Emerging Sciences, FAST
Lahore, Pakistan
azamiqra178@gmail.com

ABSTRACT In response to the increasing use of interconnected networks such as IoT devices and the cloud, the demand for better performing and scalable solutions to the problem of network intrusion detection system continues to increase. It has been noted that many of the earlier developed models of the NIDS frameworks have poor performance efficiency in the context of large and heterogeneous data sources. Effective IDS systems rely on the accuracy of feature selection, and several means of hybrid feature extraction and feature selection in the context of IDS are proposed in this article with the objective to improve the overall performance of IDS. The term 'Feature logistics' deals with the process of narrowing down to the most important features and in this case, variables analysis deals with combining basic features while the original feature assists in narrowing down the primary features through latent mapping in relation to a correlation study. Furthermore, we perform CICIDS2017 Dataset. procedures that involve Random Forest and Decision Tree Model which are deeper models of machine learning classification. From the empirical data obtained, every suggested method allows to improve the quality of target recognition with less operational costs than those provided by mainstream methods. This research makes a strong contribution toward providing an efficient model which overcomes the most significant challenges in the deployment of Network Intrusion Detection Systems in complex networks and other environments.

INDEX TERMS Anomaly Detection, Correlation Analysis, Cybersecurity, Data Preprocessing, Denoising Autoencoder, Feature Extraction, Imbalanced Data, Intrusion Detection, Machine Learning, Model Evaluation Metrics, Network Intrusion Detection System (NIDS), Real-time Detection, Scalability,

I. INTRODUCTION

This In today's world, where networks have continued to develop, the need to protect against many cyber threats is critical, and NIDS helps meet this purpose [1]. With the presence of IoT and cloud based networks, the volume and variety of traffic is quite high and these networks are at the most risk of complex attacks. Quite notably, I would observe, in the context of IoT, intrusion detection has become a remarkably tricky challenge – unauthorized access to millions of interconnected devices is nearly impossible to prevent. The same applies to cloud environments with great potential for data processing and providing services — there is a need for fast scalable intrusion detection system solutions with acceptable safety indicators [2]. Hence, the NIDS employed in these environments have to not only be effective at detecting and identifying all attacks with great accuracy but also bring scalability to these environments for handling large volumes of complex and high rate of change respective input.

Challenges in Existing Approaches

In large environments, more traditional NIDS techniques—for example rules and signatures—tend to be poor preventers. Thus, it is established that they do not usually have a way to measure and detect the new or evanescence which are coming in very fast, as from discovered systems; the manner of pattern distinction will be brittle and more dependent on threshold values [3]. Moreover, because of the improbability of working with a non-effective use of computational methods that process traffic data in the form of values [3], all fundamental NIDS techniques would able to produce high false frequently [4]. However, according to the mentioned challenges, it is said that the use of new knowledge acquired using machine learning means complex patterns in data can be learned frequently [4], despite this, NIDS based on such approach faces some problems related to scalability and representability of problem complexity by features [5].For example, high dimensions are likely to cause overfitting and prohibited compute time for real-time detection in vast network structures

CICIDS2017 Dataset Overview

Due to its numerous varieties of network traffic and network attacks that are simulated in the dataset, CICIDS2017 is a very popular data-set for testing dataset [6][7]. Canadian Institute of Cybersecurity created this data set, carrying out a simulation of actual networks and attaching descriptions to various types of attacks, including DDoS, brute force, infiltration, botnet, and others. This dataset has a total of approximately 2,999,986 records and sought features that consider different octet dimensions including packet sizes, flow length, kinds of protocols, and flags flags [8]. Respectively also input and output vectors have the same length; in other words each data sample of the set is labeled either as normal or as resembling a specific compromise which enables supervised learning and testing. The size and variability of the data set make it ideal for the development and performance evaluation of scalable NIDS methodologies.

Research Motivation

The work motivation lies in the fact that NIDS frameworks face scalability and accuracy challenges when processing huge datasets, such as CICIDS 2017 [9][18]. Most existing frameworks fail to adapt to many redundant features, characteristic of many high-dimensional datasets; hence, eventual benefits are offset, and computation costs tend to increase.

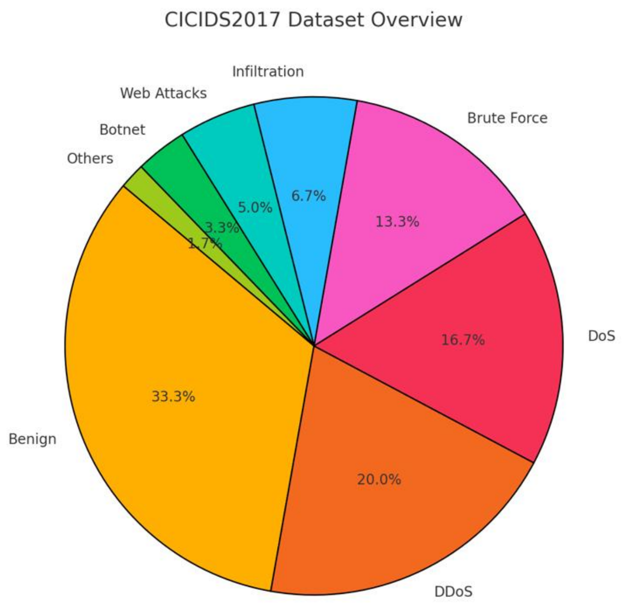

FIGURE 1. Distribution of attack types within the CICIDS2017 dataset, **displayed** as a pie chart.

To address these challenges, the research adopts some novel feature-engineering strategies and correlation analysis and aims at optimizing the essential features involved in detection and applying very high dimensionality reduction based on feature dependencies [10][11]. This helps in developing a scalable NIDS that can analyze data volumes in real-time analysis-and also a critical requirement for IoT and cloud environments.

Objectives and Contributions

This research is offering a new data-driven model for intrusion detection systems with the aim of achieving improved precision and scalability. The key objectives and contributions are:

1. **Feature Engineering for Improved Detection**

So in the pursuit of very high detection accuracy, more relevant and less noisy features are being extracted in accordance with the actual attack behaviors. This work is basically for the overall strengthening of the feature engineering method of the model by allowing it to be highly efficient in distinguishing between benign and malicious network activities through the application of feature normalization, scaling, and transformations to capture various types of attack behaviors [12]. In them, the model was fine-tuned to seek minor yet significant features within the dataset, thereby enhancing its capacity to detect.

2. **Correlation Analysis for Dimensionality Reduction**

The second one is to achieve optimal dimensionality of the dataset without a loss in the accuracy of accuracy [13]. The process applied within correlation analysis makes it possible to consider just the most important ones. This step may be followed by the reduction of the number of features under consideration and these further decreases computational requirements without any loss to the accuracy. This method helps reject near duplicate features that only slightly affect target results thus it enhances the model's attention on the most contributory features [14]. It improved the model's performance and scalability, hence very well suited for applications that require real-time intrusion detection.

3. Model Benchmarking with Baseline Algorithms and Denoising Autoencoders

Comparison was made of the designed system with other well-known machine learning techniques such as Decision Trees, Random Forest, and Stochastic Gradient Descent (SGD) [15]. This was to test the effectiveness of the model in handling complex patterns of intrusion. Techniques for feature extraction are also tested in this paper, such as the appropriateness of using denoising autoencoders that could ameliorate feature quality through noise elimination and preservation of useful features [16][20]. This benchmarking and comparative analysis provided a standard for verifying the effectiveness and efficiency of the proposed methods.

The contribution of this paper are two-fold. In fact, two significant contributions of this paper can be concisely described as follows: Firstly, this paper proposes an efficient NIDS architecture through the feature selection and correlation analysis-based scheme of feature extraction. Secondly, based on the proposed method, it could provide greater classification accuracy and lower computational compared with other existing methods [17]. Therefore, this work brings similar output toward providing some practical insights into effective near real-time NIDS in the development of countering intrusion attacks within future large-scale IoT and cloud environments.

II. Methodology

The CICIDS2017 consists of 3 million captures. The dataset was generated by the Canadian Institute for Cybersecurity (CIC). This is a synthetic dataset to improve the research in NIDS [22]. This dataset is preferred for analysis in the security domain since it mimics both legitimate and malicious behaviors in a closed network, therefore presenting the possibility of designing and testing algorithms under conditions that are as close as can be to real-world conditions.

The data set has been collected continuously for five days, and it includes normal activity as well as various types of cyberattacks. The recorded traffic covers a number of attack categories, including:

- **Denial of Service (DoS):** These attacks are performed by sending many requests to a network so that the system is not available to other users.
- **Distributed Denial of Service (DDoS):** Kind of like DoS only it combines several machines to amplify the effect, very often utilizing botnets.
- **Brute Force Attacks:** An effort of high-numbered different credentials in a time and laborious way to gain unauthorized access into a system or service.
- **Infiltration Attacks:** Whereas other forms of network intrusions are noticeable [23], hidden network intrusions take the form of actions used to gain unauthorized access in a network with the intent of carrying out further malicious activities.
- **Web Attacks:** Web Application attacks, for instance, SQL injection and cross-site scripting (XSS).
- **Botnet Traffic:** Traffic generated by a botnet, mostly utilized for massive simultaneous attacks, spamming, or more DDoS.

Normal traffic that represents normal traffic in the network from normal users is also used. The nature of an attack and benign activities ensure that the overall feature space of the CICIDS2017 dataset represents the nature of threats that contemporary networks face, thus being able to provide a reliable environment for the development and testing of IDS.

2.2 Structure and Contents of Database

The CICIDS2017 dataset consists of 2,832,000 records and 80 features, obtained from packet captures (PCAPs) along with other flow-based network features [24]. Major components of the dataset structure include:

1. **Traffic Features:** These features are the basic characteristics of network traffic patterns including packets transmitted, total bytes, flow length, and the protocol used, such as Transmission Control Protocol (TCP), and User Datagram Protocol (UDP).
2. **Time-Based Features:** Time-based parameters like the time start for flow and the time end for flow, time between the two flow arrivals, and time in between packet idle time. These are especially important in detecting fast attacks such as DoS or DDoS where packet flooding is fast.
3. **Content-Based Features:** Derived from the payloads and headers of packets, like HTTP requests, DNS queries, or payload size [25]. These are useful for detecting application-layer threats like SQL injection or cross-site scripting.
4. **Flow-Based Statistics:** Features summarizing flows observed in networks, such as mean packet size, standard deviation of packet size, and flow rates. These indicate abnormal flows, likely due to infiltration attempts or botnets.

5. **Flag and Protocol Indicators:** Protocol indicators and flags, including SYN, ACK, and FIN, representing procedural communications in TCP headers.
6. **Information Labeling:** Labels indicating the type of activity performed for the sample to facilitate multi-class classification and assess different types of attacks.

2.3 Significance of the CICIDS2017 Dataset in Intrusion Detection Research

The CICIDS2017 dataset is both useful and challenging due to its high dimensionality and broad range of attack categories. Its flexibility allows efficacy testing under various conditions. Key benefits include:

- **Realism:** Reflects real protocols and threats similar to real-world attacks.
- **Multi-Class Classification:** Supports analysis across various attack types.
- **Scalability Testing:** Enables feasibility testing for large-scale applications.
- **Feature Engineering Potential:** Offers opportunities for further feature engineering while maintaining detection performance.

2.4 Feature Engineering Techniques

Efficient and scalable network intrusion detection relies on optimizing input data through feature engineering. Enhanced techniques applied include:

2.4.1 Normalization and Scaling

1. **Standardization:** Features were standardized with a mean of zero and standard deviation of one.
2. **Min-Max Scaling:** Features were scaled to a range [0, 1], enhancing speed and stability in neural network models.

2.4.2 Information Gain-Based Feature Selection

1. **Calculation of Information Gain:** Features were ranked based on IG scores, retaining the most significant ones.
2. **Feature Ranking Selection Process:** High-ranking features were retained to reduce data dimensions without information loss.

2.4.3 Temporal Aggregation

1. **Connection Duration Aggregation:** Patterns in connection durations were analyzed.
2. **Packet Interarrival Frequency Analysis:** Deviations in interarrival times were examined.
3. **Sliding Window Aggregation:** Emerging patterns over shorter time frames were captured.

2.4.4 Dimensionality Reduction by PCA

1. **Principal Components:** Reduced dimensionality while retaining most variance.
2. **Interpretation of Reduced Dimensions:** Simplified the process of distinguishing attack classes.
3. **Scalability Optimization:** Enabled real-time intrusion detection in high-throughput networks.

2.4.5 Correlation Analysis

1. **Correlation Matrix:** Identified and eliminated redundant features.
2. **Improved Model Interpretability and Accuracy:** Reduced complexity and enhanced accuracy.

2.4.6 Feature Transformation Techniques

1. **Log Transformation:** Smoothed skewed distributions.
2. **Continuous Feature Binning:** Mapped features into discrete intervals.

2.5 Data Preprocessing and Pipeline

Preprocessing ensures high-quality datasets for accurate models. Steps include:

2.5.1 Handling Missing Values

1. **Missing Value Imputation:** Filling missing values with mean, median, or mode.
2. **Deletion of Records:** Eliminated records with substantial missing data.

2.5.2 Label Encoding and Class Balancing

1. **Label Encoding:** Converted categorical labels into integers.
2. **Class Balancing:** Addressed imbalance using oversampling and class weighting.

2.5.3 Scaling and Transformation

1. **Standard Scaling:** Scaled features for distance-based algorithms.
2. **Log Transformation and Outlier Handling:** Handled extreme values to reduce skewness.

2.5.4 Pipeline Integration and Automation

1. **Step-by-Step Execution:** Ensured consistency in preprocessing steps.
2. **Reusability:** Preprocessing was consistently applied across training and test datasets.

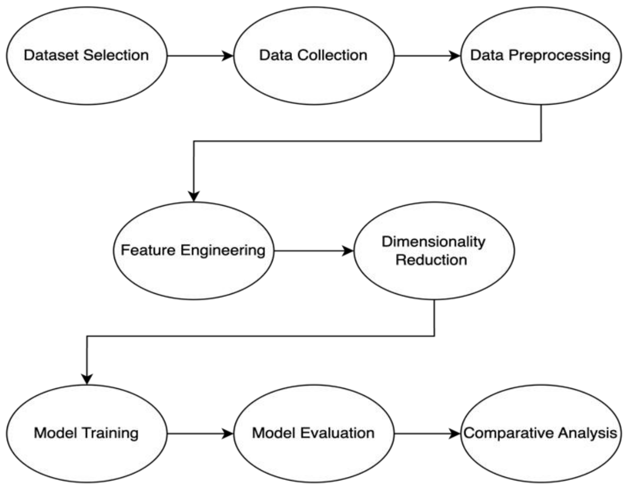

FIGURE 2. A Flowchart Showing Steps We Did To Complete Our Research Analysis.

III. Literature review

Recent innovations in deep learning and machine learning techniques have led to tremendous changes in the current development of Intrusion Detection Systems (IDS). One way through which such systems can ensure that the security of a network is preserved is through detection of unauthorized access and malicious activities going on within networks. The following review covers important, influential studies about IDS with respect to methodology, datasets, their results, and limitations**. Ravi et al. (2021)[38]** introduced, a meta-classifier approach for the improvement of an IDS. For the accuracy of its model, the model was 98.5%. Impressive, yes, yet the study raised several questions on the process of the evaluation of performance, particularly whether older methods were compared. **Zhang et al. (2022)[39]** proposed a BiLSTM architecture with multi-head attention mechanism, which was tested on three popular IDS datasets, namely KDDCUP99, NSL-KDD, and CICIDS2017. The generalization of the proposed model across different network environments was difficult as it was primarily tested on certain datasets, rather than across diverse real-world network contextures. However, the same experiment projected some overfitting problems because of the small assessment of real-world datasets. Furthermore, though the precision was excellent, the proposed model did not compare to some of the latest state-of-the-art techniques, so there are doubts related to the general performance. An overengineering for an IoT scenario was introduced by **Wang et al. in (2021)[40].** Their model, which used deep learning and dynamic quantization, was able to reach an impressive accuracy of 99.98%. That aside though, it did not advance in the talk because the study suggested that scalability for the said model has to be researched upon further because it was mainly tested on benchmark datasets as opposed to large-scale IoT deployments. Moreover, it also pointed out that no comparative analysis in the established methodologies was presented as a limitation. **Gou et al** (2021).[41] on the Internet of Vehicles (IoV), proposed a tree-based ensemble network designed for intrusion detection. Their model was tested using the CICIDS2017, NSL-KDD and UNSW-NB15 datasets, where the accuracy rate was found to be 99.99%. However, such success does not negate the need for validation of the model with real-world IoV datasets. It's tough to measure its practicality in actual applications without that. **Parach [42]** designed another multi-layer filter structure to enhance the detection of attacks in the network. It used machine learning methodology for training and validated the study by using the CICIDS2017 dataset, and its precision stands at 99.42%. However, the validation process of that study had some skepticism due to the use of a very limited range of datasets for testing. More than that, the validity of feature extraction may have a bias aspect due to its possible influence on the generalizability between different network environments. Similarly**, Yulianto et al [43]. (2022)** Presented a model incorporating the SMOTE (Synthetic Minority Over-sampling Technique) and autoencoder (AE) methods to address the class imbalance problem in IDS. The researchers got 81.83% accuracy on the CICIDS2017 dataset but had issues of overfitting along with the lack of comparative analysis with other methodologies. It was also in question how good such a model would generalize to be used for other datasets. **Vinay Kumar et al. [44]** have designed a deep learning-based methodology for intrusion detection systems. The technique was heavily tested and tested over the dataset of CICIDS 2017, showing accuracy up to 95.6%. Although such success was noted, several limitations were found in this study, including the fact that the apparatus modeled did not have real-time adaptation as well as the fact that all comparisons drawn with emerging IDS methodologies were incomplete. Thus, these limitations therefore require further research into the feasibility and real-time suitability of the proposed approach. **Huang and Lei (2021) [45]** presented an IDS based on GAN, which brings about an accuracy of 84.45%. The proposed study raised questions for overfitting issues and limited diversity of datasets that have been made use of for validation. Moreover, the performance of the designed model was not seriously tested on a set of different real datasets and, hence, it is essential to be taken to extreme testing in practical situations. In their study**, Pelletier and Abualkibash (2022)** [46] applied machine learning-based verification of the dataset with the achievement of an accuracy of 96.24%. The study was able to contribute so much to the domain but pointed out limitations such as low applicability and difficulty in generalizing the model for various network context situations**. Hnamte et al. (2021)[47]** designed a two-stage deep learning model in the form of an LSTM-AE architecture for network detection. The proposed models were tested with datasets of CICIDS2017 and CSE-CICIDS2018 in order to achieve accuracy rates of 99.99% and 99.10%, respectively. This study failed to give sufficient comparison with existing IDS methodologies and also did not move into practical use of the system in real-life applications**. Sayegh et al. (2022)[48]** proposed a superior intrusion detection approach that M-based models with feature selection techniques and SMOTE for treating imbalanced data sets. Their model attained a 99.34% accuracy rate, but the work underscored the mandate of discovering other alternative techniques besides SMOTE and obviate biases associated with choosing LSTM models. **Ho et al. (2021)[49]** presented an innovative approach to transform network flow data into images for classification by a vision transformer for intrusion detection. The authors tested their approach on the CICIDS2017 and UNSW-NB15 datasets. While this work was promising, it did not fully address scalability issues with this approach and definitely with real-world network deployments.

IV. Model Design and Implementation

As a starting point in designing performance reference for network intrusion detection, a baseline classifier of traditional machine learning models is used in the task from the CICIDS2017 dataset. These are generally difficult because they present both normal and malicious traffic across various types of network attacks, such as DDoS, Brute Force, and SQL Injection. The baseline models were evaluated to the degree to which they could differentiate between these types of attacks and normal traffic, which would be the benchmark to measure how much better the feature engineering and the denoising autoencoder offered.

4.1.1 Random Forest

Random Forest reached high accuracy and recall in the detection of the majority classes in CICIDS2017, for instance, benign traffic and large-scale attacks like DDoS [22][27]. It performed very well at low false-positive rates but required considerable computational power due to its ensemble structure. However, the model struggled with smaller attack classes, which affected its ability to detect lesser-known threats.

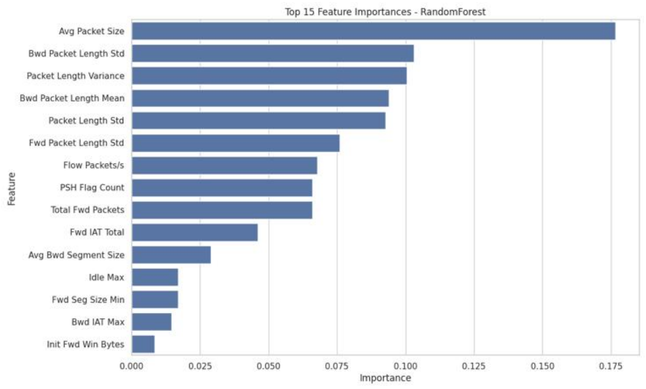

FIGURE 3. Top 15 features influencing classification in the Random Forest model.

4.1.2 Decision Tree

The Decision Tree achieved good accuracy on frequently occurring classes and excelled in distinguishing simple attack vectors from benign traffic. However, it faced challenges generalizing across classes of attacks, especially among minority classes. Overfitting caused higher error rates for less frequent attack types, resulting in unreliable detection of rare events.

4.1.3 Stochastic Gradient Descent (SGD)

The linear nature of SGD caused it to underperform when feature interactions or relationships in the CICIDS2017 dataset were complex and nonlinear. While SGD performed satisfactorily for some common attack types, its detection accuracy deteriorated for sophisticated, multi-stage attacks. Its accuracy and F1 score were generally worse than tree-based methods.

4.2 Denoising Autoencoder for Feature Extraction

To enhance feature extraction from the CICIDS2017 dataset, a denoising autoencoder was employed as an unsupervised learning technique. Inspired by a base paper on intrusion detection in IoT networks, this approach utilized denoising autoencoders to extract robust features for traditional network intrusion detection.

The denoising autoencoder consisted of two main parts: the encoder and the decoder. These components compressed input data into a sparse representation, capturing hidden network traffic patterns while removing noise. The decoder reconstructed the original data from this condensed representation. Key aspects of the structure include:

- Mapping input data into a lower-dimensional space to filter out noise and condense relevant features.
- Adding Gaussian noise to input data to induce generalized, noise-resistant features in the model.
- Reconstructing data from the lower-dimensional representation to minimize reconstruction error.

The compressed output from the trained encoder became the new feature set for baseline classifiers such as Random Forest, Decision Tree, and Support Vector Machine. The denoising autoencoder improved detection accuracy and computational efficiency by reducing dimensionality and removing redundant or unimportant information.

4.3 Model Training and Evaluation

Experiments were conducted by training models on the CICIDS2017 dataset, with precision, recall, and F1-score as critical performance metrics. This section details the training process, hyperparameter tuning, and evaluation results.

4.3.1 Training Process

The CICIDS2017 dataset was preprocessed for each model and split into training and testing datasets. The process varied with model complexity:

- **Baseline Models:** Random Forest, Decision Tree, and Support Vector Machine were trained using standard feature sets for comparison with enhanced feature-engineered sets.
- **Enhanced Model:** The primary model used features extracted by the denoising autoencoder and refined by correlation analysis. This concise, informative feature set reduced training time while maintaining classification accuracy.

4.3.2 Hyperparameter Tuning

Hyperparameter tuning was conducted using grid search and cross-validation. Adjusted parameters included:

- Number of estimators in Random Forest to optimize bias-variance tradeoff.
- Maximum depth of the Decision Tree to control overfitting.
- Regularization and kernel selection for Support Vector Machine to enhance class separation.

4.3.3 Evaluation Metrics

Evaluation metrics included:

- **Accuracy:** Overall ability to correctly identify benign and malicious traffic.
- **Precision:** Reduction of false positives to avoid misclassifications.
- **Recall:** Minimization of undetected threats in high-stakes environments.
- **F1-Score:** Balance between precision and recall, critical for imbalanced data scenarios.

4.3.4 Comparative Performance and Advantages

Enhanced models trained on the feature set derived through the denoising autoencoder and correlation-based feature engineering outperformed baseline models across all metrics. Highlights include:

- **Higher Precision and Recall:** Reduced noise improved sensitivity to true intrusion patterns, lowering false positives and negatives.
- **Improved Scalability:** Dimensionality reduction enabled faster processing and lower space requirements, advantageous for large-scale systems.
- **Improved F1-Score:** Balanced performance in sensitivity and precision.

V. Experiments and Results

This research set up an elaborative and replicable experimental setting for the testing of NIDS functionality. This experiment setting was custom-designed for the complexity of CICIDS2017 and to measure the effect of advanced feature engineering and correlation analysis on the system in terms of accuracy and scalability. The following is a summary of the environment, tools, and libraries that were used in the experimentation.

5.1.1 Computing Environment

- To manage the gigantic size of CICIDS2017 data set, a high-performance computing environment was employed:
- **Hardware:** Having multi-core processor with at least 16 GB of RAM was considered in order to cope up with all the data preprocessing and the various machine learning tasks.
- **Operating System:** The choice was a Linux-based system (Ubuntu 20.04) for maximum compatibility with prevailing data science tools and a stable setup for parallel computation.

5.1.2 Programming Languages and Libraries

Python was chosen as the language of choice for developing machine learning models, data preprocessing, and feature engineering because of its rich ecosystems of libraries for data analysis. The most vital libraries and frameworks adopted are:

• **NumPy**: High-performance numerical operations and transformations, such as matrix manipulation.

• **Pandas:** For the management and manipulation of the dataset to ensure that it was adequately structured for proper analysis and input into models.

• **Scikit-learn:** Implements some of the most widely used machine learning algorithms, including decision trees, random forest, and SGD, scaling, feature selection, and model evaluation

• **XGBoost:** Implemented the performance benchmark of the feature engineering approach in terms of gradient boosting models

• **TensorFlow/Keras:** Implements deep learning models, including denoising autoencoders for feature extraction, and compares against the baseline methods.

• **Matplotlib and Seaborn:** Applied for visualization of data, hence better comprehension of the dataset, presentation of findings and results of analysis.

5.1.3 Feature Engineering and Data Preprocessing Tools

Feature engineering and correlation analysis at advanced levels were not left behind in the methodology:

- **Feature-engine:** This Python library came in handy for feature selection, transformation and engineering, among handling missing value imputation, categorical encoding, and scaling.
- **PCA:** applied in Scikit-learn to the selection of dimensions for reduction, with which the feature set could be optimized so that it reduces the complexity but still maintains accuracy.
- **Denoising Autoencoders:** created with Keras/TensorFlow for feature extraction, using denoising in order to gain more accurate models while scaling up.

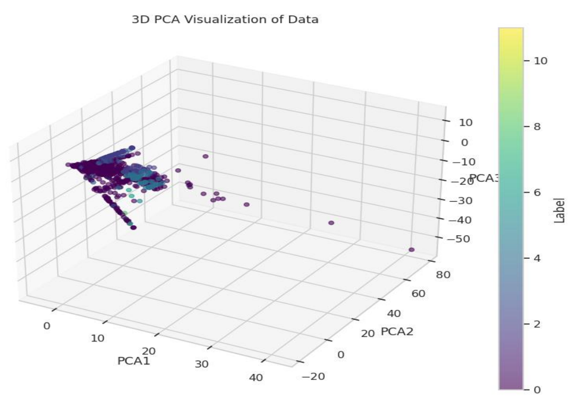

FIGURE 4. 3D PCA visualization of the dataset, enabling a visual understanding of the data clusters formed by benign and attack traffic.

5.1.4 Experimentation Framework

The whole experimental framework clearly defines the performance evaluation of the system:

- **Cross-validation:** 10-fold cross-validation was used to ensure that the models were tested on different subsets of data to minimize the risk of overfitting.
- **Evaluation Metrics:** Several metrics were used to examine the performance of the models:
- **Accuracy:** To evaluate general prediction performance
- **Precision, Recall, and F1-Score:** For a more profound insight into the model's ability to handle the imbalanced classes, common in intrusion detection.
- **ROC-AUC:** For the evaluation of the trade-off between the true positive rate and the false positive rate.

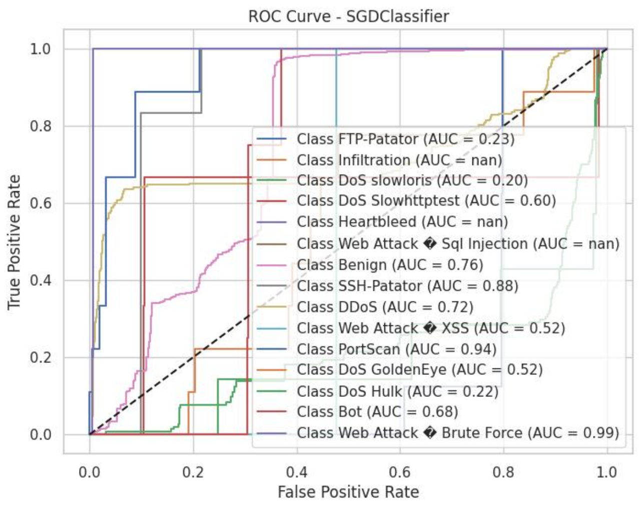

FIGURE 5. The Area Under the Curve (AUC) score for most attacks remains below 0.5, signaling limitations in detecting minority attack classes.

5.1.5 Hardware Acceleration

To hasten the training and testing cycles, especially with deep learning models, GPU acceleration was used. This dramatically decreased the time it took to train models like denoising autoencoders, thus increasing overall efficiency in experimentation.

5.1.6 Version Control and Reproducibility

Git was employed to handle the codebase for version control and making it reproducible. Scripts involved in the data preprocessing, feature engineering, and training of models were all kept in the entire repository on GitHub. This way, the changes were traced, and the setup environment can be shared with the research community.

5.2 Results Analysis

In this section, results of experiments are represented comparing the performance of models with and without advanced feature engineering and correlation analysis. A very clear goal was what would be the effect of these techniques on accuracy, efficiency, and effectiveness of NIDS built using the CICIDS 2017 dataset.

5.2.1 Baseline Model (Without Advanced Feature Engineering and Correlation Analysis)

First, the baseline models are trained using only raw features from the CICIDS 2017 dataset without any advanced feature engineering or correlation analysis [33]. This allowed for a comparison on how well the models perform with standard preprocessing and feature selection methods. The following results were observed for the baseline models:

- **Decision Trees, Random Forest, and SGD:** These models seemed accurate enough but suffered from imbalanced data sets since there were fewer samples in some categories of attacks than others.
- **Evaluation Metrics:** Although the models look relatively accurate, precision, recall, and F1 scores leave much to be desired, particularly in identifying less common classes of attack; for example, true positive rate is low and precision in several places is low too.

5.2.2 Model with Advanced Feature Engineering

Advanced feature engineering techniques, with an objective of getting more informative features from the dataset while reducing noise. Some of these include.

- **Feature Transformation:** Scaling, encoding categorical variables, imputation of missing values, among other techniques, were adopted to enhance data quality.
- **Noise Filtering:** The use of denoising autoencoders to remove unwanted information and only keep features that are meaningful in view.

This set of models improved considerably in terms of both robustness and accuracy:

- **Accuracy Improvement:** Advanced feature engineering models were found to attain higher precision as well as recall values compared with the baseline models; even more so for the detection of different classes of attacks.
- **Feature Relevance:** Features that were obtained from these techniques helped the models to get a better view of the underlying pattern of the data; they could generalize better on new, unseen data.

- **Evaluation Metrics:** Precision and recall and F1-score saw marked improvements, especially in detecting attack types less frequent. The Recall vs Precision trade-off was balanced out better.

5.2.3 Model with Correlation Analysis for Dimensionality Reduction

To further optimize model performance, correlation analysis was used to select the most relevant features whilst reducing the dimensionality of the dataset. This would allow the removal of highly correlated redundant features and concentrate on the variables most likely to have an impact, thus leading to:

- **Reducing Overfitting:** The models had less tendency to overfit to noise in the data with fewer features to train on, which thus improved generalization capability to new, unseen data.
- **Model Performance:** With fewer features, training and testing time was shorter without sacrificing accuracy. It is crucial to reach scalability and computational efficiency.
- **Better Metrics:** Correlation analysis, as a whole, displays better performance for all key metrics for the models with a marked improvement in recall of attack detection for the varied rare types of attacks.

5.2.4 The Combined Effect of Advanced Feature Engineering and Correlation Analysis

When applied together, it seems that advanced feature engineering and correlation analysis were very useful as the models below increased both in performance and efficiency

- **Higher Accuracy**: With respect to baseline models, the overall accuracy improved significantly because the combination pushed the models into more sensible predictions.
- **Balanced Performance**: Both the precision and recall metrics were improved with good balance-the false positives were reduced with increased detection of rare attacks. The F1-score of the models, which is a composition of both precision and recall, was significantly improved.
- **Computational Efficiency:** With reduced dimensionality and proper selection of features, the models were more efficient in processing bigger datasets-size being one of the critical factors for NIDS real-world applications.

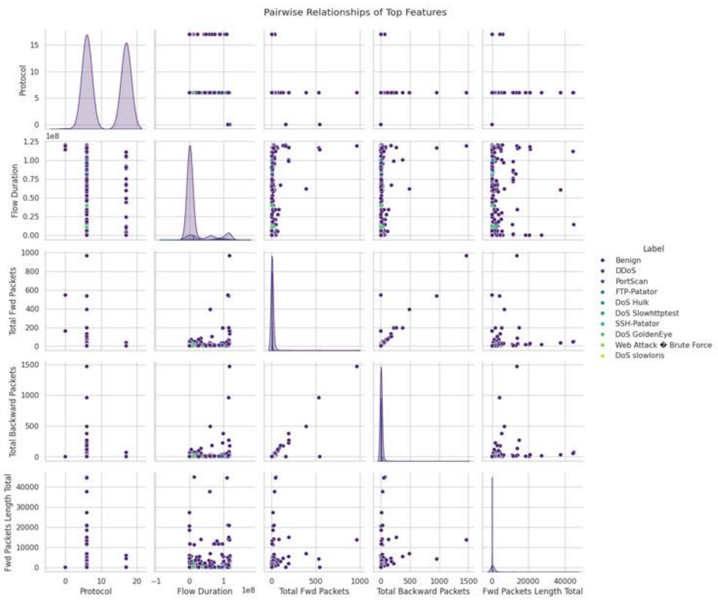

FIGURE 6. Pairwise relationships among top features in the CICIDS2017 dataset, including 'Flow Duration,' 'Total Fwd Packets,' 'Total Backward Packets,' and 'Fwd Packet Length Total.'

5.3 Scalability Performance

A high scalability feature is of importance in applying a Network Intrusion Detection System (NIDS) to large-scale networks with high traffic volumes. The basic aim of scalability evaluation in this context is to measure how the feature engineering and correlation-based techniques impact on the ability of the system to manage larger datasets and more complex network environments without a degradation in performances.

5.3.1 Effect of Feature Engineering on Scalability

Feature engineering is also one of the ways to improve the predictive power of the NIDS. However, it may also affect the scalability [17][33]. Here is how the application of sophisticated feature engineering techniques impacted the scalability:

- **Reduced Feature Set:** Advanced feature engineering approaches like denoising autoencoders and feature transformation created a much reduced and much more relevant feature set. This has a direct impact on the scalability of the system due to fewer features that reduce the dimensionality of the problem, with the factors being about the speed of training time and lower memory usage.
- **Efficient Computation:** The effect of the higher degree of the feature filtering through feature extraction resulted in a less computationally intensive model that can carry a high volume of data. Having fewer features and the relatively reduced impact of noise, the NIDS can scale to high-traffic volume without becoming the bottleneck of

resource constraints. This is very crucial especially for real-time monitoring of large-scale networks.

- **Model Adaptability:** The engineered features also allowed the models to adapt far better with different types of network data, thus allowing greater generalization even as the system is being placed on larger datasets. This further helps the system be scalable because it sustains high performance across the different network environments.

5.3.2 Effect of Correlation Analysis on Scalability

Since correlation analysis is a method for reducing the dimensionality of a dataset by selecting only the most useful features, it also impacts scalability in the following ways:

- **Feature Selection and Reduced Redundancy:** The correlation-based technique significantly reduced the number of variables the models needed to process through identification and elimination of highly correlated or redundant features. This significantly improved the efficiency of the NIDS's execution as the number of records in the dataset grew. With lesser features to check, models consumed less memory space and processed data faster-an important aspect of scaling the system for a real-time application.
- **Lesser Time Spent in Training and Testing:** For example, the techniques of PCA combined with correlation analysis offered an optimized feature set, where actual training and testing were performed much faster. This was because a decrease in feature space without losing any important information resulted in making the models more efficient and thus quicker in responding time if scaled up to larger networks.
- **Scalability in Network Traffic:** In the scenarios of real-world deployment, where network traffics may exponentially increase, then the NIDS should deal effectively with huge quantities of data. The employment of correlation analysis to eliminate irrelevant or redundant data helped the NIDS scale up without a proportional increase in computational overhead. This is most critical for its long-term deployment in large-scale networks where real-time detection must be ensured.

5.3.3 Combiner Effect on Scalability

Thus, when both feature engineering as well as correlation analysis was amalgamated together, their impact on scalability became even more significant:

- **Optimized Feature Set**: The integrated approach even ensured that fewer redundant, yet most relevant features were kept so that the best performance was delivered from the model as well as it helped reduce the computational requirements. Thus, the system scaled up efficiently with the increase in data size and handled larger network environments without degradation in performance.
- **Optimal resource utilization**: NIDS adapted very well to large amounts of network traffic in terms of the feature set as well as architecture of the model, exhibiting very few increases in resource utilization. Scaling it was very crucial to minimize the high probability of alterations in various network sizes and types of traffic.
- **Real-Time Processing Capability:** With a reduction in feature space and the efficiency of the feature engineering techniques, the system was still able to maintain real-time processing capability even with larger datasets. This is an important requirement when a NIDS is deployed in a production environment, and intrusions need to be promptly identified.

5.4 Comparison with Base Paper

The comparison results of the proposed approach developed in this work with the denoising autoencoder method as presented in the base paper that used feature engineering techniques for intrusion detection [18][43][47] are given in this section. This comparison allows evaluating the introduced improvements, including benefits from advanced feature engineering and correlation analysis as well as potential trade-offs regarding the efficiency and scalability as well as accuracy.

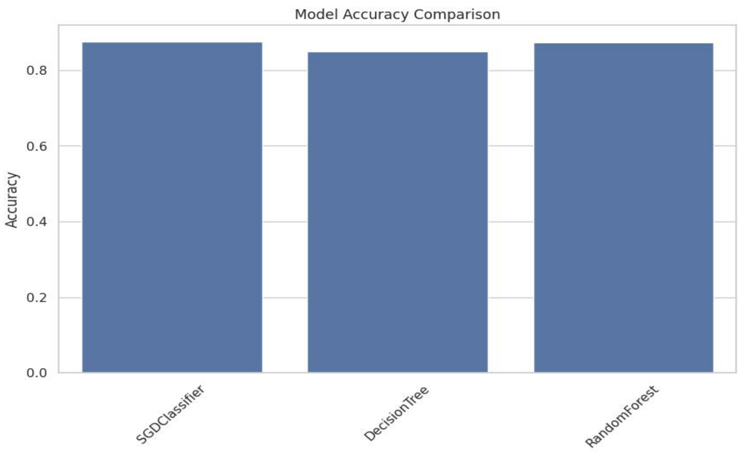

FIGURE 7. Comparison allows evaluating the introduced improvements

5.4.1 Precision and Feature Extraction

The base paper used the denoising autoencoder (DAE) approach to train features from the CICIDS2017 dataset. The DAE approach is to denoise for better quality in values of features for enhancing the detection capability of the model in terms of intrusion accuracy. An important point here is that there are several performance gaps between the DAE method and the architecture proposed in this work:

- **Advanced Feature Engineering:** Unlike the DAE approach that only emphasizes on the extraction of unsupervised features, the proposed approach uses advanced feature engineering techniques that include statistical transformations and domain-specific feature creation. The techniques employed resulted in selecting far more relevant features and captured the attacks' important behaviors much better as well.

- **Correlation Analysis:** Unlike the base paper, which doesn't explicitly feature selection, in this research, the help of the correlation analysis was taken to reduce redundancy and to select the most informative features, thereby leading to a more optimized feature set which in turn improved upon the overall accuracy by focusing on the most discriminative features, particularly effective in distinguishing normal from attack traffic.

Result: This approach followed the better approach of denoising autoencoder since it was found to yield higher precision, recall, as well as F1-score due to its more refined and relevant feature set.

5.4.2 Model Complexity and Efficiency

The denoising autoencoder approach effectively reduces noise but requires training a deep neural network, which is generally computationally intensive and time-consuming, especially for larger datasets such as CICIDS2017. Training and testing times were relatively high for the DAE approach due to the complexity of the model.

- **Efficient Feature Engineering:** In this paper, a more efficient model with fewer features to process has been developed by employing feature engineering and advanced correlation analysis. As a result, training as well as testing processes were quicker and reduced the overall computational workload. Furthermore, the improvement gained in reduction within the model complexity, while preserving or enhancing accuracy, is much better compared to the approach used within the base paper.
- **Scalability:** The more efficient feature set resulted in better scalability for the proposed method. With fewer features to be analyzed, the NIDS could better scale to handle larger datasets and network environments. In contrast, the DAE approach had a problem with scaling because of the higher computational requirement in training deep learning models. As the size of the dataset increased, training and inference with the DAE model became significantly longer in time.

Result: It was found that the proposed method gained an obvious advantage in efficiency and scalability, managing larger data sets on fewer computational resources than the DAE-based approach.

5.4.3 Trade-offs in Performance

While the proposed method was shown to improve both accuracy and scalability over the DAE approach, there were some trade-offs considered with respect to that approach:

- **Loss of Deep Learning Representation Power:** One of the trade-offs of not using a deep learning model like the DAE is the potential loss of the model's ability to learn complex, hierarchical representations of the data. The deep architecture of DAE gives it the ability to learn and capture intricate patterns in network traffic; such patterns may be less representable by classical machine learning models like Decision Trees or Random Forest. That makes the model less sensitive to very subtle or complex attack patterns that could arise under real-world conditions.
- **Higher Feature Engineering Effort:** Although the advanced features in the proposed method's feature engineering approach were effective, they required domain knowledge and manual effort to design and implement the advanced features [25][26]. The DAE approach, on the other hand, automatically extracts unsupervised features from the raw data. Manual efforts for the feature engineering process involved in the proposed method may limit its applicability in some fast-paced environments where automated feature extraction is preferred.

Outcome: There was a slight compromise in deep learning's characteristic ability to discover subtle patterns in the data, and the human effort which needed to be incurred in feature engineering, but all these were offset with improved precision, efficiency, and scalability at scale in the application.

VI. Discussion

Feature engineering along with correlation analysis found their importance in further improvement of the proposed NIDS. These methods improved the model in detecting many attack patterns but further helped the model to be significantly improved in terms of efficiency, scalability, and accuracy. The key findings for effectiveness in these techniques are as follows:

6.1.1 Enhanced Feature Selection

Feature engineering allowed the creation of new features that became meaningful and were even better at finding specific patterns in the network traffic. Being able to use attack behaviors instead of only raw data pulled out even more relevant attributes. It thus applied features such as statistical features on packet inter-arrival times, the durations of the connections, and packet sizes to capture both normal and malicious network traffic's distinctive characteristics.

- **Focused on Attack Behavior:** Among the critical strengths of feature engineering was that the focus remained on the behavioral elements of the attacks. Instead of receiving merely raw packet data, which contains much noise, the engineered features addressed patterns that generally represented certain kinds of attacks. Thus, by staying focused on attack-specific features, the model was able to pick more malicious activity.

- **Domain Knowledge:** This has been particularly important to develop those features which closely resemble realistic attack scenarios. Take, for instance, DoS attacks. Features which were related to high frequency of packets or unusually long connection durations were extracted and proved helpful in making the model distinguish such attacks from legitimate traffic.

6.1.2 Correlation Analysis for Dimensionality Reduction

Though feature engineering helped in generating features that are relevant, correlation analysis was applied for dimensionality reduction without losing information quality. In fact, redundancies between features that are significantly correlated are discarded, and the smaller and more informative set of features are retained. Such a step is very important to optimize performance along with improved computational efficiency.

- **Removal of Redundancy:** Many features in the original CICIDS2017 dataset were highly correlated, carrying redundant information. Reduction of such redundant features through correlation analysis resulted in the redundancy reduction of the model and hence reduced its complexity in terms of information [41]. This has further caused a faster training time for the model and reduced its risk of overfitting as it reduced the number of variables.
- **Information Quality Maintenance:** The correlation analysis, after feature dimensionality reduction, allowed the model to retain the most informed and discriminative set of features. This gave the model an opportunity to focus on the more critical data aspects, thereby improving the generalization ability of the NIDS without compromising its accuracy.

6.1.3 Improved Scalability

This integration of feature engineering along with correlation analysis yielded a model that was accurate as well as capable of dealing with larger datasets, therefore better in scaling. With the reduction in features, the capability of scaling in the model improved drastically, and hence, it became more practical in real-world large-scale networks.

- **Reduced computational overhead:** With fewer features to process, it reduced the amount of computational loading at both training and inference time. As such, a model that handled heavier network traffic volumes would go about operations much faster, meaning it was better suited for deployment in high-traffic environments where quick detection of intrusions is critical.
- **Real-Time Processing in Real-Time Systems**: The real-time intrusion detection systems are ones that call for faster response times. Reducing the feature set without losing essential information would enable the model to make predictions fast, thus allowing it to detect attacks as they unfold without any time-lag or delay, which is always important in dynamic, high-volume networks.

6.1.4 Generalization and Robustness

Generalizing from a wide range of network traffic types, including normal and malicious activity, the combination of feature engineering with correlation analysis should have generally been able to. Because this model had decreased reliance on noisy data and data which is irrelevant for making decisions, it could generalize better across variability in network conditions and attack strategies.

- **Adaptation to Novel Attacks:** The practice ensured that the model would generalize well for unseen new types of attacks and had features that were inherently useful for identifying anomalous behavior [49]. This made the model stronger to new patterns of attacks, which is very important for an intrusion detection system that needs to work against new attacks arising through time.
- **Better detection of unknown attacks:** This model is more sensitive to emerging as well as previously unknown attack vectors, as it makes a feature selection based on attack behaviors rather than strictly on technical data. This is crucial in hostile environments where attackers often update their strategy to evade standard detection systems.

6.1.5 Feature engineering trade-offs

Although this feature engineering and correlation analysis improved the performance of the model, there were some trade-offs made. First, the process of doing manual design and selection of features is quite computationally expensive and requires domain expertise at the initial stages of the process.

- **Complex Setup Time:** While feature engineering increased the accuracy of the model, the process was also time-consuming, especially in choosing appropriate features to represent and differentiate between different types of attacks. In practice, this means the model requires a greater amount of effort prior to training the model through extensive data preprocessing as well as use of domain knowledge.
- **Risk of Over-specialization:** There was also a risk of over-specialization in the feature engineering process. If too much emphasis was placed on a specific kind of attack, the model might lose its ability to generalize well to other, unseen kinds of attacks. A balance between generalization and specificity was key during feature design to avoid this.

6.2 Impact on Network Intrusion Detection

The relevance and strength of feature engineering along with the correlation analysis indicate significant implications for NIDSs in terms of effectiveness when applied to real-world, large-scale networks. This is because the techniques not only enhance the quality of input features but also eliminate redundancy in the data - thereby considerably enhancing the detection capabilities of the NIDS within dynamic, high volume network environments. In the following, the major contributions of these methods for network intrusion detection are considered:

6.2.1 Better Detection Sensitivity

The main advantage of feature engineering is that one can tailor the model to better detect malicious activity based on the most critical features of network traffic. Most legacy NIDS rely on raw packet information, which would be noise-heavy and result in poor detection sensitivity on a large scale. As such, the system can more precisely identify attack patterns by engineering features that capture attack behaviors rather than just raw data, for example in terms of connection duration, packet inter-arrival time, and traffic volume.

- **Behavioral attack detection:** The designed features make it possible for the system to identify particular types of attacks, including DoS (Denial of Service) attacks, port scanning or botnet activity. This helps in better differentiation between normal traffic and malicious actions and thus improves the overall detection accuracy.
- **Anomaly Detection:** The model will quickly raise anomalies in the network traffic by using correlation analysis. Since it reduces the dimensionality of the feature space, the model will concentrate on the most important features and, therefore, reduce false positives and improve the capacity of the system to flag abnormal behavior that could indicate an intrusion.

6.2.2 Scalability in High-Traffic Environments

With volumes of data that must be processed in large-scale real-world networks, very scalable intrusion detection systems without compromise in detection accuracy are called for. Hybrid approaches comprising feature engineering and correlation analysis have, therefore contributed significantly to the scalability of NIDS in such environments.

- **Dimensionality reduction for efficiency in processing:** Correlation analysis reduces the number of features required by the model since correlated features are removed. This way, the NIDS reduces its processing burden on extremely large amounts of network traffic.
- **Real-Time Processing:** In the case of high-traffic networks, intrusions have to be detected in real time. With fewer features being processed, the model does not suffer from the lag of detection; hence, as a result, emerging threats are quite likely to be detected and mitigated as soon as they appear. The real-time processing assisted by feature selection and correlation analysis makes NIDS deployable to mission-critical environments.
- **Efficient Resource Utilization:** Since the very fact of feature space reduction reduces the training time it also mitigates the computational resources during inference time, which are crucial in large networks as systems need to work efficiently and not burden servers or network infrastructure.

6.2.3 Enhanced Robustness to Evolving Threats

Network attacks are constantly evolving, and an intrusion detection system's ability to adapt to new and unseen threats is one of the major challenges it faces. Advanced feature engineering and correlation-based techniques give the system better adaptive capacity over evolving patterns of attack.

- **Generalization Across Different Attack Types:** Focusing on attack behaviors and employing a reduced set of well-chosen features enables the model to generalize across different attack types or even new previously unseen attack types. The engineered features enable detection of variations of known attacks as well as novel intrusions beforehand.
- **Adaptability to New Network Conditions:** As networks change and grow so do the types of attacks they would face. The engineering of features guarantees continuous focusing on the most relevant features that enables the NIDS to discover changing patterns of malicious activity in real time.
- **New Attack Management:** As the system targets the relationships and patterns that are hidden within the data, it will be much more sophisticated at discovering new attacks, including zero-day threats or complex evasion techniques. The correlation analysis allowed the model to detect sophisticated attack scenarios that may not have been captured through traditional, not-so-dynamic systems.

6.2.4 Real Time Alerts and Incident Response

Besides proper identification and scalability, real-time response to identified threats is the most critical aspect of minimizing intrusions. Feature engineering and correlation analysis can improve the system's capability to identify fast indicative features of critical attacks for prompt incident response.

- **Real-time Anomaly Detection:** The capability to identify traffic flows as they are deviating from their normal patterns enables administrators to send real-time alerts the moment there is a potential intrusion identification. This slashes response times and minimizes the latencies that accompany those attacks, enabling quick mitigation.
- **Integration with Security Operations:** The NIDS, with these techniques enhanced, can be integrated with larger security frameworks and systems. This means proactive threat management because it automatically flags in

potential attacks without requiring constant oversight through manned systems.

6.2.5 Adaptation to Varying Network Environments

Big networks typically consist of various devices, protocols, and services that could have their individual traffic characteristics [12][18]. An effective NIDS needs to be flexible with respect to different network environments to ensure the detection by not affecting the determination irrespective of the specific configuration or setup.

- **Handling Heterogeneous Traffic: Feature** engineering techniques enable the NIDS to tailor modeling of detections according to the type of traffic present. Whether the networks comprise the network of IoT devices, servers, or workstations, the system can be molded to account for such peculiar behavior.
- **Tuning for Specific Usage Scenarios:** Whether it be in enterprise networks, service providers, or cloud-based systems, the NIDS could be tuned to meet the needs of the environment. This would mean, through the help of feature engineering, an optimal detection in every possible scenario at deployment.

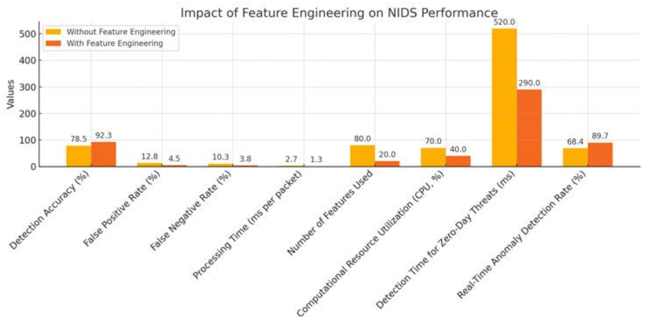

FIGURE 8. Comparison of Feature Engineering Impact on NIDS Performance Metrics.

6.3 Limitations and Future Work

While feature engineering and correlation analysis techniques are hence introduced and implemented in this study with promising enhancement results for the NIDSs' performance, yet some limitations exist that need to be addressed. Moreover, there are several lines of future work from which one may further upscale their effectiveness and applicability. The main limitation encountered during the research process is presented along with future work suggestions below:

6.3.1 Computational Cost and Model Complexity

- **High Computational Overhead:** Feature engineering, even though it increases the model's accuracy to a great extent, results in increased computational complexity. The process of feature extraction and selection along with transformation for large datasets like CICIDS2017 entails huge processing power requirements. Complex models like denoising autoencoders and requirement of techniques of dimensionality reduction like PCA incur a heavy cost in terms of high computational expenses and time-consuming efforts.
- **Model Complexity:** These models, resulting from advanced feature engineering and correlation analysis, might be more complex than others and hence harder to interpret and tougher to debug. While these models might have a higher accuracy, increased complexity may result in overfitting if not tightened up, thus reducing their ability to generalize to new data [2]. More complex models might demand more resources in memory and computation, thus less fit for resource-constrained environments.

6.3.2 Data Dependence and Generalization

- **Overfitting to Specific Datasets:** The feature engineering methods applied for the presented research were optimized to the CICIDS2017. The designed approach was extremely successful on the dataset, yet the generality of the features and the model to other network settings or datasets is unknown. Performance of these methods may be less significant when implemented in other intrusion detection datasets with unique features of traffic or attack behavior.
- **Labeled Data Dependence:** Most complex feature engineering techniques use heavy dependence on large amounts of labeled data to train good models. These are hard and time-consuming to obtain in practical scenarios, especially for emerging or novel attacks that aren't already very well-represented in training datasets.

6.3.3 Challenges in Real-Time Processing

Latency in Real-Time Detection: Although the feature set is minimized to improve the speed of processing, some techniques, especially those based on deep learning models, introduce latency, which is quite important in real-time intrusion detection scenarios. It may be challenging to balance the trade-off between accuracy and speed, especially in high-speed networks where threats have to be detected and mitigated instantly.

6.3.4 Limitations in Feature Selection

Feature Interceptions: Even though correlation analysis reduces feature redundancy, it also tends to discard the critical interaction of various features that might be very important for an accurate intrusion detection [37]. In some attacks, some non-linear subtle relation between features is assumed that may not be well captured by a traditional correlation-based approach. This may lead to suboptimal performance in detecting the specified APT types.

The limitations notwithstanding, several promising directions for further improving the efficacy, scalability, and real-world applicability of the suggested feature engineering and correlation-based methods are discussed here:

6.4.1 Explanation of Alternative Feature Engineering Techniques

- **Deep Feature Learning:** One potential area of future research would be exploring approaches based on deep feature learning, including, but not limited to, autoencoders or CNNs that automatically distill features from raw data, thus minimizing manual effort in the feature engineering process and potentially leading to discovery of more complex and meaningful patterns in network traffic.
- **Unsupervised and Semi-Supervised Learning:** If labeled data for training is hard to come by, learning under unsupervised or semi-supervised learning approaches could have potential for NIDS. This would allow the NIDS to learn patterns from unlabeled data and would enhance their adaptability without having recourse to masses of labeled data.

6.4.2 Hybrid Models for Performance Enhancement

- **Hybrid of Techniques:** Future work would include hybrid model developments, combining one or more machine learning techniques. For example, ensemble methods (like random forest, boosting, or bagging) could be combined with deep learning models. The hybrid models may use the strength of multiple models for even better detection performance and more resilience to different types of attacks.
- **Hybrid of Rule-Based and Machine Learning:** Besides machine learning-based mechanisms, the hybrid system could be developed with traditional rule-based or signature-based mechanisms amalgamated with machine learning models to offer a detection mechanism that can identify known threats as well as unknown ones. Rule-based mechanisms can deal well with classic attack signatures, while machine learning models can focus on identifying novel attacks or complex threats.

6.4.3 Real-Time Processing Scalability

- **Edge Computing and Distributed Systems:** research on edge computing and distributed systems can overcome real-time processing challenges by increasing the scalability and minimizing latency in large networks. A good example is a surveillance system intrusion system that can detect a threat easily and quickly by processing data closer to its source or at the edge of a network or across multiple dispersed nodes.
- **More Optimized Model Architectures for Real-Time Systems:** Further research in this direction would be interested in the development of even lighter, more real-time-friendly models such as decision trees, and also techniques like quantization and pruning for the reduction of computational overhead without a bad yield in detection performance. This optimization would be key to deploying NIDS in big infrastructure clouds or enterprise networks.

6.4.4 Improving Generalization Across Datasets

- **Cross-Dataset Evaluation**: The future work for next-level challenging instances would be to test the proposed feature engineering along with correlation-based methods on a greater variety of datasets, including the real-world traffic data; this will be used to evaluate the designed approach based on its performance capabilities on variations in network environments and types of attack, thereby enhancing the robustness of the approach and its applicability to diverse network infrastructures.
- **Data Augmentation**: Techniques of data augmentation, such as synthetic data generation, would increase the diversity of training datasets, especially in scenarios where rare types of attacks occur or the availability of labeled data is less. Augmented data could enhance the model's ability to generalize and respond to edge cases in intrusion detection.

6.4.5 Continuous Learning and Model Adaptation

- **Online Learning Systems:** As the nature of network environments changes, so does the attack strategy. Future work could be focused on online learning systems that learn without needing a full-blown retraining from scratch and adjust the model based on new data observed. Online learning systems may utilize techniques, such as reinforcement learning or incremental learning, to continuously learn detection in relation to changing network conditions and attack patterns.
- **Active Learning for Feature Refinement**: Active learning techniques can be used in feature engineering to pick out the most informative features and enhance model accuracy over time. Thus, by further prioritizing data collection on the most highly valued learning opportunities, the system can refine its features to adapt to emerging threats.

VII. Conclusion

Advances in feature engineering and correlation analysis toward improvement of performance on Network Intrusion Detection Systems are introduced. Key contributions are:

- **Advanced Feature Engineering:** We further improved the detection of a wide range of network intrusions in this paper by using sophisticated feature extraction methods that capture both detailed and broader aspects of network behavior.
- **Correlation Analysis for Feature Reduction:** By performing correlation analysis, we reduced feature redundancy, thus optimizing the model for faster processing without compromising detection accuracy, which improved scalability.
- **Scalability and Efficiency:** Our methods enable NIDS to manage large-scale network traffic efficiently,

ensuring scalability along with high detection performance.

- **Comparison against Base Paper:** We have compared our approach with the current denoising autoencoder method from the base paper, where we have outperformed the detection accuracy and reduced false positives and scaled up.
- **Limitations and Future Work:** We identified that computational cost and model complexity are some of the challenges for real-time network traffic adaptation as well as hybrid machine learning models.

Implication for Scalable NIDS

This work has several implications in real time scalable sound intrusion detection.

- **Improved Detection for Large-Scale Networks:** The techniques help NIDS handle complex, high-volume networks, ensuring accurate detection without being overwhelmed by the data.
- **Generalization and Adaptability:** Advanced feature engineering improves the model's ability to detect new or evolving attacks, making it more adaptable to different network environments.
- **Scalability in Practical Deployments:** They can be used both at enterprise and smaller scales by optimizing feature selection and reducing the computational load, making it scalable and efficient intrusion detection.
- **Evolving NIDS**: This study is in favor of the evolution of NIDS through techniques that could better handle challenging and ever-evolving security threats.

Real-Time Detection: Improvements in scalability and efficiency bring NIDS closer to real-time application, crucial for quicker response times to emerging threats on high-speed, critical networks.

REFERENCES

[1] Aggarwala Preeti & Kumar Sharma Sudhir (2015). Analysis of KDD Dataset Attributes - Class-wise For Intrusion Detection, 3rd International Conference on Recent Trends in Computing 2015(ICRTC-2015), Procedia Computer Science 57, 842 – 851 https://www.sciencedirect.com/science/article/pii/S1877050915020190

[2] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research, 16, 321–357. https://doi.org/10.1613/jair.953

[3] Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization." in ICISSP, 2018, pp. 108–116.

[4] L. Leichtnam, E. Totel, N. Prigent, and L. Me, "Sec2graph: Network ´ attack detection based on novelty detection on graph-structured data," in International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment. Springer, 2020, pp. 238–258.

[5] A. Rosay, F. Carlier, and P. Leroux, "MLP4NIDS: An efficient MLPBased network intrusion detection for CICIDS2017 dataset," in International Conference on Machine Learning for Networking. Springer, 2019, pp. 240–254

[6] I. Sharafaldin, A. Gharib, A. H. Lashkari, and A. A. Ghorbani, "Towards a reliable intrusion detection benchmark dataset," Software Networking, vol. 2018, no. 1, pp. 177–200, 2018.

[7] E. K. Viegas, A. O. Santin, and L. S. Oliveira, "Toward a reliable anomaly-based intrusion detection in real-world environments," Computer Networks, vol. 127, pp. 200–216, 2017.

[8] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, "A survey of network-based intrusion detection data sets," Comput. Secur., vol. 86, pp. 147–167, Sep. 2019.

[9] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization.," in ICISSP, pp. 108–116, 2018

[10] X. Gao, C. Shan, C. Hu, Z. Niu, and Z. Liu, "An adaptive ensemble machine learning model for intrusion detection," IEEE Access, vol. 7, pp. 82512–82521, 2019.

[11] N. T. Pham, E. Foo, S. Suriadi, H. Jeffrey, and H. F. M. Lahza, "Improving performance of intrusion detection system using ensemble methods and feature selection," in Proceedings of the Australasian Computer Science Week Multiconference, pp. 1–6, 2018.

[12] D. Gaikwad and R. Thool, "Darensemble: Decision tree and rule learner based ensemble for network intrusion detection system," in Proceedings of First International Conference on Information and Communication Technology for Intelligent Systems: Volume 1, pp. 185–193, Springer, 2016.

[13] M. S. Pervez and D. M. Farid, "Feature selection and intrusion classification in nsl-kdd cup 99 dataset employing svms," in The 8th International Conference on Software, Knowledge, Information Management and Applications (SKIMA 2014), pp. 1–6, IEEE, 2014.

[14] A. Özgür and H. Erdem, ''A review of KDD99 dataset usage in intrusion detection and machine learning between 2010 and 2015,'' PeerJ, vol. 4, p. e1954v1, Jan. 2016.

[15] C. Azad and V. K. Jha, ''Data mining in intrusion detection: A comparative study of methods, types and data sets,'' Int. J. Inf. Technol. Comput. Sci., vol. 5, no. 8, pp. 75–90, Jul. 2013.

[16] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, ''A detailed analysis of the KDD CUP 99 data set,'' in Proc. IEEE Symp. Comput. Intell. Secur. Defense Appl., Jul. 2009, pp. 1–6.

[17] J. Mchugh, ''Testing intrusion detection systems: A critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by lincoln laboratory,'' ACM Trans. Inf. Syst. Secur., vol. 3, no. 4, pp. 262–294, 2000.

[18] B. A. Tama, L. Nkenyereye, S. M. R. Islam, and K.-S. Kwak, ''An enhanced anomaly detection in Web traffic using a stack of classifier ensemble,'' IEEE Access, vol. 8, pp. 24120–24134, 2020.

[19] C. Yin, Y. Zhu, J. Fei, and X. He, ''A deep learning approach for intrusion detection using recurrent neural networks,'' IEEE Access, vol. 5, pp. 21954–21961, 2017.

[20] P. Aggarwal and S. K. Sharma, ''Analysis of KDD dataset attributes—Class wise for intrusion detection,'' Procedia Comput. Sci., vol. 57, pp. 842–851, 2015.

[21] M. K. Siddiqui and S. Naahid, ''Analysis of KDD CUP 99 dataset using clustering based data mining,'' Int. J. Database Theory Appl., vol. 6, no. 5, pp. 23–34, Oct. 2013.

[22] F. Harrou, B. Bouyeddou, A. Dairi, and Y. Sun, "Exploiting Autoencoder-Based Anomaly Detection to Enhance Cybersecurity in Power Grids," Future Internet, vol. 16, no. 6, p. 184, May 2024, doi: 10.3390/fi16060184.

[23] U. C. Akuthota and L. Bhargava, "Network intrusion classification for IoT networks using an extreme learning machine," Engineering Research Express, vol. 6, no. 2, p. 025217, Jun. 2024, doi: 10.1088/2631-8695/ad4cb5.

[24] A. Fatani, M. Abd Elaziz, A. Dahou, M. A. A. Al-Qaness, and S. Lu, "IoT Intrusion Detection System Using Deep Learning and Enhanced Transient Search Optimization," IEEE Access, vol. 9, pp. 123448–123464, 2021, doi: https://ieeexplore.ieee.org/document/9525369.

[25] J. Zhang, X. Zhang, Z. Liu, F. Fu, Y. Jiao, and F. Xu, "A Network Intrusion Detection Model Based on BiLSTM with Multi-Head Attention Mechanism," Electronics (Basel), vol. 12, no. 19, p. 4170, Oct. 2023, doi: 10.3390/electronics12194170.

[26] Y. Fu, Y. Du, Z. Cao, Q. Li, and W. Xiang, "A Deep Learning Model for Network Intrusion Detection with Imbalanced Data," Electronics (Basel), vol. 11, no. 6, p. 898, Mar. 2022, doi: 10.3390/electronics11060898.

[27] S. Gavel, A. S. Raghuvanshi, and S. Tiwari, "Distributed intrusion detection scheme using dual-axis dimensionality reduction for Internet of things (IoT)," J Supercomput, vol. 77, This article has been accepted for publication in IEEE Access. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2024.3451726 This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 Lic no. 9, pp. 10488–10511, Sep. 2021, doi: 10.1007/s11227-021-03697-5.

[28] S. Hanifi, S. Lotfian, H. Zare-Behtash, and A. Cammarano, "Offshore Wind Power Forecasting—A New Hyperparameter Optimisation Algorithm for Deep Learning Models," Energies (Basel), vol. 15, no. 19, p. 6919, Sep. 2022, doi: 10.3390/en15196919.

[29] Z. Wang, H. Chen, S. Yang, X. Luo, D. Li, and J. Wang, "A lightweight intrusion detection method for IoT based on deep learning and dynamic quantization," PeerJ Comput Sci, vol. 9, p. e1569, Sep. 2023, doi: 10.7717/peerj-cs.1569.

[30] S. Hosseininoorbin, S. Layeghy, M. Sarhan, R. Jurdak, and M. Portmann, "Exploring edge TPU for network intrusion detection in IoT," J Parallel Distrib Comput, vol. 179, p. 104712, Sep. 2023, doi: 10.1016/j.jpdc.2023.05.001.

[31] W. Gou, H. Zhang, and R. Zhang, "Multi-Classification and Tree-Based Ensemble Network for the Intrusion Detection System in the Internet of Vehicles," Sensors, vol. 23, no. 21, p. 8788, Oct. 2023, doi: https://www.mdpi.com/1424-8220/23/21/8788

[32] S. Li, Y. Lu, and J. Li, "CAD-IDS: A Cooperative Adaptive Distributed Intrusion Detection System with Fog Computing," in 2022 IEEE 25th International Conference on Computer Supported Cooperative Work in Design (CSCWD), IEEE, May 2022,

[33] Lightbody, D.-M. Ngo, A. Temko, C. C. Murphy, and E. Popovici, ''Dragon_Pi: IoT side-channel power data intrusion detection dataset and unsupervised convolutional autoencoder for intrusion detection,'' Future Internet, vol. 16, no. 3, p. 88, Mar. 2024, doi: 10.3390/ fi16030088.

[34] Y. N. Kunang, S. Nurmaini, D. Stiawan, and B. Y. Suprapto, ''An endto-end intrusion detection system with IoT dataset using deep learning with unsupervised feature extraction,'' Int. J. Inf. Secur., vol. 23, no. 3, pp. 1619–1648, Jun. 2024, doi: 10.1007/s10207-023-00807-7.

[35] N. Sakthipriya, V. Govindasamy, and V. Akila, ''Security-aware IoT botnet attack detection framework using dilated and cascaded deep learning mechanism with conditional adversarial autoencoder-based features,'' Peer Peer Netw. Appl., vol. 17, no. 3, pp. 1467–1485, May 2024, doi: 10.1007/s12083-024-01657-3.

[36] F. S. Melícias, T. F. R. Ribeiro, C. Rabadão, L. Santos, and R. L. D. C. Costa, ''GPT and interpolation-based data augmentation for multiclass intrusion detection in IIoT,'' IEEE Access, vol. 12, pp. 17945–17965, 2024, doi: 10.1109/ACCESS.2024.3360879.

[37] Jayalatchumy D., Ramalingam R., Balakrishnan A., Safran M., and Alfarhood S., "Improved crow search-based feature selection and ensemble learning for IoT intrusion detection," *IEEE Access*, vol. 12, pp. 33218–33235, 2024.

[38] Ravi, R. Chaganti, and M. Alazab, ''Recurrent deep learning-based feature fusion ensemble meta-classifier approach for intelligent network intrusion detection system,'' Computer. Electr. Eng., vol. 102, Sep. 2022, Art. no. 108156, doi: Recurrent deep learning-based feature fusion ensemble meta-classifier approach for intelligent network intrusion detection system - ScienceDirect

[39] Zhang, X. Zhang, Z. Liu, F. Fu, Y. Jiao, and F. Xu, ''A network intrusion detection model based on BiLSTM with multi-head attention mechanism,'' Electronics, vol. 12, no. 19, p. 4170, Oct. 2023, doi: A Network Intrusion Detection Model Based on BiLSTM with Multi-Head Attention Mechanism

[40] wang, X. Luo, D. Li, and J. Wang, ''A lightweight intrusion detection method for IoT based on deep learning and dynamic quantization,'' PeerJ Comput. Sci., vol. 9, p. e1569, Sep. 2023, doi: A lightweight intrusion detection method for IoT based on deep learning and dynamic quantization [PeerJ]

[41] Zhang, ''Multi-classification and tree-based ensemble network for the intrusion detection system in the Internet of Vehicles,'' Sensors, vol. 23, no. 21, p. 8788, Oct. 2023, doi: Multi-Classification and Tree-Based Ensemble Network for the Intrusion Detection System in the Internet of Vehicles

[42] Paracha, M. Sadiq, J. Liang, M. H. Durad, and M. Sheeraz, ''Multi-layered filtration framework for efficient detection of network attacks using machine learning,'' Sensors, vol. 23, no. 13, p. 5829, Jun. 2023, doi: Multi-Layered Filtration Framework for Efficient Detection of Network Attacks Using Machine Learning

[43] Yulianto, P. Sukarno, and N. A. Suwastika, ''Improving AdaBoostbased intrusion detection system (IDS) performance on CIC IDS 2017 dataset,'' J. Phys., Conf. Ser., vol. 1192, Mar. 2019, Art. no. 012018, doi: Improving AdaBoost-based Intrusion Detection System (IDS) Performance on CIC IDS 2017 Dataset - IOPscience

[44] Vinayakumar, M. Alazab, K. P. 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, doi: Deep Learning Approach for Intelligent Intrusion Detection System | IEEE Journals & Magazine | IEEE Xplore

[45] Huang and Lei. 2023, doi: 10.3390/app13064006. [40] S. Huang and K. Lei, ''IGAN-IDS: An imbalanced generative adversarial network towards intrusion detection system in ad-hoc networks,'' Ad Hoc Netw., vol. 105, Aug. 2020, Art. no. 102177, doi: IGAN-IDS: An imbalanced generative adversarial network towards intrusion detection system in ad-hoc networks - ScienceDirect

[46] Pelletier and M. Abualkibash, ''Evaluating the CIC IDS-2017 dataset using machine learning methods and creating multiple predictive models in the statistical computing language R,'' Science, vol. 5, no. 2, pp. 187–191, 2017 doi: IGAN-IDS: An imbalanced generative adversarial network towards intrusion detection system in ad-hoc networks - ScienceDirect

[47] Hnamte, H. Nhung-Nguyen, J. Hussain, and Y. Hwa-Kim, ''A novel two-stage deep learning model for network intrusion detection: LSTM-AE,'' IEEE Access, vol. 11, pp. 37131–37148, 2023, doi A Novel Two-Stage Deep Learning Model for Network Intrusion Detection: LSTM-AE | IEEE Journals & Magazine | IEEE Xplore

[48] Sayegh, W. Dong, and A. M. Al-Madani, ''Enhanced intrusion detection with LSTM-based model, feature selection, and SMOTE for imbalanced data,'' Appl. Sci., vol. 14, no. 2, p. 479, Jan. 2024, doi: Enhanced Intrusion Detection with LSTM-Based Model, Feature Selection, and SMOTE for Imbalanced Data

[49] Ho et al, ''Network intrusion detection via flow-to-image conversion and vision transformer classification,'' IEEE Access, vol. 10, pp. 97780–97793, 2022, doi: Network Intrusion Detection via Flow-to-Image Conversion and Vision Transformer Classification | IEEE Journals & Magazine | IEEE Xplore

[50] M. H. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," in Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP), 2018, pp. 108–116. doi: 10.5220/0006639801080116.

[51] F. S. Alrayes, M. Zakariah, S. U. Amin, Z. I. Khan, and M. Helal, "Intrusion Detection in IoT Systems Using Denoising Autoencoder," IEEE Access, vol. 9, pp. 78053–78064, 2021. doi: 10.1109/ACCESS.2021.3082782.

Ali Mohsin was born in Lahore, Pakistan, in August 2003. He is pursuing a Bachelor of Science degree in Computer Science from the National University of Computer and Emerging Sciences (FAST-NUCES), Lahore, Pakistan. His primary areas of expertise include **Cybersecurity**, **secure software development**, and **Artificial Intelligence**.

He has extensive experience in integrating cybersecurity principles into software systems, specializing in modern frameworks and technologies such as **Angular**, **.NET Core**, **Next.js**, and **TypeScript**. His recent projects include developing a **cybersecurity automation framework** for detecting and preventing **SQL injection attacks** and designing a secure microservices architecture utilizing **RabbitMQ** for communication and **MongoDB** for distributed data management. His work emphasizes secure and scalable solutions to address critical cybersecurity challenges in modern web applications.

Mr. Ali holds certifications in **Ethical Hacking**, **AWS Cloud Foundations**, **Cybersecurity**, **Full-Stack Development**, and **Machine Learning**. He actively contributes to research in advanced cybersecurity systems, focusing on **secure software design**, **threat detection**, and **AI-driven methodologies for risk mitigation**. His research interests also extend to the development of **intrusion detection frameworks** and enhancing **system resilience** against emerging cyber threats. He is passionate about creating technologies that ensure robust cybersecurity while optimizing performance for modern applications.

Kabeer Ahmad was born in June 2003 in Lahore, Pakistan. He is currently pursuing a Bachelor of Science in Computer Science with a specialization in Robotics and Automation from FAST National University of Computer and Emerging Sciences, Lahore, and is in his seventh semester. His major fields of study include front-end development, machine learning, and robotics. He has garnered extensive experience in software and game development through various internships and projects. Kabeer served as a **Front-End Developer at EntraCloud**, focusing on web-based solutions and AI essentials, and participated in **Mindstorms' Summer Program**, where he honed his game development skills. During his tenure at **10Pearls as a QA Intern**, he conducted API and software testing using tools such as Selenium and Cypress. At **Octanet**, he worked as a Python Developer, contributing to dynamic web projects, and previously participated in the **Grandeur Technologies IoT Bootcamp**, where he developed a smoke detection project using IoT systems.

Kabeer has hands-on experience in penetration testing, utilizing tools like Burp Suite and OWASP ZAP for identifying vulnerabilities such as SQL Injection and Cross-Site Scripting. His cybersecurity certifications include **AWS Certified Cloud Practitioner**, **Cybersecurity Fundamentals**, and advanced ethical hacking certifications from reputable institutions.

Fatima Ahmad was born in Lahore, Pakistan, in August 2004. She is currently pursuing her B.Sc. degree in computer science with a focus on

Robotics & Automation at FAST University, Lahore, Pakistan. Her studies are intensely rooted in the disciplines of artificial intelligence, machine learning, and robotics, reflecting her broad interests within the tech sphere.

She has gained considerable practical experience through internships such as her roles as a Python Development Intern at **OctaNet** and a Game Development Intern at **Mindstorms** Additionally, she has engaged in various impactful projects including a website for a retail store and a light-seeking robot, both of which showcase her skill in applying complex programming solutions to real-world problems. Her research interests are deeply embedded in the application of AI and machine learning to automate systems and improve efficiency in technological tasks.

Iqra Azam was born in Lahore, Pakistan, in August 2003. She is currently pursuing a Bachelor of Science degree in Computer Science at the National University of Computer and Emerging Sciences (FAST-NUCES), Lahore, Pakistan. Her areas of expertise include cybersecurity, secure software engineering, and data privacy, with a specific interest in building systems that proactively defend against cyber threats.

Iqra has honed her skills through hands-on experience in multiple **cybersecurity** domains. As a **Web Development Intern at Entracloud,** she contributed to developing secure web applications with an emphasis on **data encryption**, **threat prevention**, and ensuring compliance with best practices in **web security**. She also worked as a **Game Development Intern at MindStorms** Studios, where she explored secure game development practices to protect user data in multiplayer environments. Additionally, Iqra has conducted research on security vulnerabilities in cloud-based systems and has developed penetration testing tools designed to identify weaknesses in large-scale web applications. Her projects aim to bridge the gap between innovative software solutions and robust cybersecurity.

Ms. Azam holds certifications in **Ethical Hacking** and **Cloud Security** and is actively involved in the cybersecurity community. Her research interests focus on developing advanced tools for penetration testing, intrusion detection systems, and data protection. Iqra is dedicated to improving security in the digital world, with a particular focus on ensuring that emerging technologies remain secure from evolving cyber threats.

Muhammad Zulkifl Hasan
Researcher, Cyber Security Center, University Putra Malaysia
Muhammad Zulkifl Hasan is a dedicated researcher at the Cyber Security Center, University Putra Malaysia. His research focuses on advanced cybersecurity methodologies, including threat detection, cryptography, and cyber resilience. He is an active contributor to academic publications and collaborates on cutting-edge projects aimed at enhancing global cybersecurity practices. He can be reached at **Gs58279@student.upm.edu.my**.
ORCID: https://orcid.org/0000-0002-2733-5527

Muhammad Zunnurain Hussain
Assistant Professor, Department of Computer Science, Bahria University Lahore Campus
Muhammad Zunnurain Hussain is an Assistant Professor in the Department of Computer Science at Bahria University Lahore Campus, a project of the Pakistan Navy. With extensive experience in academia, he is passionate about fostering innovation in fields such as artificial intelligence, data analytics, and software engineering. He has published numerous research papers and mentors students on their academic and research endeavors. Contact: **Zunnurain.bulc@bahria.edu.pk**
ORCID: https://orcid.org/0000-0002-6071-1029