AI2DS: Advanced Deep Autoencoder-Driven Method for Intelligent Network Intrusion Detection Systems

Abstract

Network security remains a critical concern as digital infrastructure continues to expand and evolve. Traditional security measures often struggle to cope with the sophistication and variety of modern cyber threats. This research introduces an innovative approach called AI2DS (Advanced Deep Autoencoder-Driven Method for Intelligent Network Intrusion Detection Systems) using an advanced deep autoencoder-based architecture for intelligent network intrusion detection, which enhances network security by identifying deviations from normal network behavior. The autoencoder is trained exclusively on normal data, enabling it to reconstruct these patterns accurately and detect network intrusion when deviations occur. Key contributions of this study include the development of an adaptive thresholding technique that leverages reconstruction error from the autoencoder to dynamically classify network activities as normal or anomalous. Additionally, the research simplifies the detection process by aggregating various attack types into a single 'intrusion' class, making it applicable to a broad range of network intrusion scenarios. The proposed AI2DS model demonstrates promising results in detecting network intrusions with high accuracy and offers potential for real-time application. Future directions for this research include enhancing the model's real-time capabilities, exploring deeper and more complex autoencoder architectures, and extending its application to other domains requiring anomaly or intrusion detection. This study sets the groundwork for a more adaptive, robust, and efficient approach to network security, aligning with the evolving needs of modern digital landscapes. The proposed AI2DS model yields an average improvement of 8.31%, 25.08%, 9.81%, and 16.35% based on Precision, Recall, F1 Score, and Accuracy over the state-of-the-art methods.

CCS Concepts

• **Security and privacy → Network intrusion and its mitigation**.

Keywords

Intrusion Detection, Cybersecurity, Autoencoder, Network security, NSL-KDD, Machine Learning

ACM Reference Format:

. 2024. AI2DS: Advanced Deep Autoencoder-Driven Method for Intelligent Network Intrusion Detection Systems. In *Proceedings of ACM Workshop on Artificial Intelligence and Security (AISec '24).* ACM, New York, NY, USA, 10 pages. https://doi.org/XXXXXXX.XXXXXXX

AISec '24, October 18, 2024, Salt Lake City, Utah, USA

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

In today's dynamic digital landscape, the security of network infrastructure remains a critical imperative concern for organizations worldwide. The rapid advancement of technology has created an era where cyber threats are not only increasing in frequency but also growing in complexity [7, 16]. As adversaries continually refine their tactics to counteract modern defense mechanisms, the need for innovative strategies to bolster cyber defenses becomes increasingly urgent. In this context, intrusion detection systems (IDSs) emerge as the frontline defenders, tasked with the crucial mission of vigilantly monitoring and analyzing network traffic to detect and react accordingly to potential malicious activities [3, 28].

IDSs serve as the first line of defense against a multitude of threats, ranging from common malware attacks to much more sophisticated, targeted intrusions. They actively scan network traffic in real-time, employing various detection techniques such as signature-based detection, anomaly-based detection, and behavior-based detection to identify potential security incidents [2, 11]. Signature-based detection relies on predefined patterns associated with known threats, while anomaly-based detection seeks deviations from normal network behavior, and behavior-based detection focuses on the behavior of network entities and users. Through continuous monitoring and analysis, IDS can rapidly detect and respond to security incidents, thereby safeguarding network assets and preserving the integrity of an organization's critical infrastructure.

However, traditional methods of intrusion detection, like signature-based detection, have often found it challenging to keep up with the dynamic and complex nature of cyber threats [27]. While these methods can effectively identify known attack patterns, they tend to struggle with new, sophisticated attacks that bypass traditional detection techniques. The necessity to continuously update signature databases to track evolving threats adds to the complexity, as cybercriminals continually develop new evasion strategies. This situation underscores the urgent need for more adaptable and resilient intrusion detection systems.

In this context, the application of machine learning (ML) technologies in the cybersecurity field is becoming increasingly prevalent [4]. Machine learning offers significant improvements to intrusion detection systems by enabling them to learn and adapt to new threats autonomously. By analyzing extensive network data and applying sophisticated algorithms, ML-driven systems can identify unusual patterns that may indicate malicious activities, without relying on specific signatures or pre-established rules. This proactive stance helps combat the continuously changing threat landscape by utilizing the strength of autonomous pattern recognition across vast and complex network datasets [15].

Machine learning models for intrusion detection employ diverse learning strategies such as supervised, unsupervised, and semi-supervised learning, enabling them to dynamically adapt to the continuously evolving landscape of cyber threats [22]. Supervised learning relies on labeled data to train models to differentiate between normal and malicious activities by recognizing established patterns. On the other hand, unsupervised learning detects anomalies without the need for labeled data, which is particularly suitable for addressing modern cybersecurity challenges where new threats emerge rapidly [22]. Semi-supervised learning combines elements of both approaches, creating a flexible framework that leverages both labeled and unlabeled data. This holistic approach enhances the adaptability and effectiveness of responses to a broad spectrum of cyber threats.

In our approach, we harness the capabilities of an autoencoder architecture to revolutionize intrusion detection utilizing widely acknowledged NSL-KDD dataset [24]. Autoencoders, renowned for their ability to distill complex input data into compact representations and learn intricate patterns in a dataset, offer a transformative advantage in network intrusion detection tasks [30]. By integrating autoencoders into our methodology, we empower our intrusion detection system to discern between normal network behavior and anomalies with high accuracy by use of the reconstruction error. This utilization of autoencoder architectures represents a paradigm shift in the research field of intrusion detection, enabling the system to autonomously learn and adapt to emerging threats without the need for explicit information. Furthermore, by leveraging the power of deep learning techniques incorporated into the autoencoder, we augment our approach with the capability to extract intricate patterns from vast amounts of network data in the fraction of a unit of time compared to traditional methods. This integration of autoencoder architectures and deep learning methodologies not only enhances the agility and effectiveness of our intrusion detection system but also signifies a significant advancement in the field of cybersecurity to further safeguard our utmost important assets. In summary, the major contributions of this paper are as follows:

- Proposes a unique utilization of an advanced autoencoder architecture for intelligent network intrusion detection systems, where the normal data is utilized to train the autoencoder model, enabling the system to reconstruct normal patterns and detect intrusion based on deviations from these patterns.
- Introduces an adaptive thresholding technique based on reconstruction error from the autoencoder model, allowing for the classification of intrusion by comparing the error with dynamically determined thresholds.
- Methodology is generalized to binary classification by aggregating various attack classes into a single 'intrusion' class, making it applicable to a wide range of network intrusion detection scenarios.

This paper comprises 5 main sections and is structured as follows: A thorough introduction is provided in Section 1. Section 2 provides a comprehensive literature review. In Section 3, we present our methodology, including data preprocessing, autoencoder architecture, and experimental setup. Section 4 presents the results of our experiments and comparative analysis with existing methods. Finally, Section 5 concludes the paper and outlines avenues for future research.

2 Literature Review

This section provides descriptions of the applications of traditional machine learning algorithms, ensemble-based methodologies, and deep learning approaches for network intrusion detection. The use of artificial neural networks (ANNs) along with their powerful use in the field of network intrusion detection will be reviewed more thoroughly.

Traditional machine learning algorithms have been utilized in recent years to detect, classify, and counteract cyberthreats in modern computing systems such as desktops or mobile devices [12, 17]. These models utilize classical ML algorithms such as Support Vector Machine (SVM), Decision Tree, Logistic Regression, and Naïve Bayes [25, 26]. In network intrusion problems, the classification tasks can vary significantly, ranging from binary class classification to multi-class classification. Binary class problems involve distinguishing between normal and intrusion-like network activity, which simplifies the detection process but may potentially miss nuanced or more specific threats [17]. More complex machine learning frameworks involve 4-class or 5-class problems, where the classification extends to different types of intrusions, such as DoS attacks, Probe attacks, R2L (Remote to Local) attacks, and U2R (User to Root) attacks as seen in [18, 26]. These multi-class tasks provide a more intricate detection capability, allowing for more specific countermeasures against distinct types of intrusion threats. The selection of binary class rather than multi-class classification settings depends on the specific requirements and complexity of the network environment under consideration.

Although conventional machine learning methods have demonstrated promising results in combating diverse cyber threats over time, each algorithm has its own vulnerabilities. Ensemble-based algorithms address these individual weaknesses by conjoining multiple weaker classifiers to construct a much more overall resilient model. Popular ensemble techniques in machine learning typically comprise of Stacking, Random Forest, or Xtreme Gradient Boosting (XGBoost) [8, 13, 21]. For example, research outlined in [19] showcased the effectiveness of an XGBoost ensemble method for Internet of Things (IoT) network intrusion detection. XGBoost is an ensemble learning algorithm that combines the predictions of multiple decision trees to improve predictive accuracy and generalization by iteratively training new trees to correct the errors of the previous ones to yield thorough and precise classifications.

Several recent papers [9, 14] highlight the efficiency of deep learning in diverse realms of cyber intelligence compared to both traditional and ensemble-based methodologies, encompassing analyzing of malware samples as well as network intrusions. The adaptability of Deep Neural Networks (DNNs) is noteworthy, as their structural configurations can significantly diverge based on the particular domain of study and the specific application at hand. Notably, within contemporary intrusion detection frameworks, deep learning architectures like Long Short-Term Memory Networks (LSTMs), Convolutional Neural Networks (CNNs), and Autoencoders (AEs) have each played a pivotal role in cybersecurity research, showcasing their versatility and effectiveness [6, 20]. For example, in

paper [6], the authors introduced an intrusion detection framework integrating CNN with BiSRU (Bi-directional Simple Recurrent Unit) to effectively discern and classify intrusions through the analysis of network traffic records. Additionally, in [5], authors introduced a Deep Belief Network (DBN) that provides an accurate and efficient intrusion detection system based on the NSL-KDD dataset.

The proposed AI2DS model is evaluated against several state-of-the-art models. These models focus on binary classification for intrusion detection. Tauscher et. al. [23] utilizes a hierarchical approach to distinguish between normal and intrusive behaviors. In contrast, Kasongo [10] employs a specialized recurrent neural network (RNN) known as XGBoost-LSTM, which identifies abnormal behavior by retaining critical information and dropping unnecessary details for classification. Lastly, Yousefi et. al. [29] presents an innovative unsupervised feature learning method for network-based intrusion detection, leveraging an autoencoder to effectively learn feature representations.

Our proposed AI2DS model extends upon previous research by leveraging the capabilities of deep learning for feature learning and the ability to perform classification based on the generated reconstruction error. We aim to address the challenge of network intrusion detection by utilizing autoencoders for binary classification, distinguishing between normal network activity and abnormal behavior or network intrusion. Autoencoders offer the capability of learning compact representations of input data, making them well-suited for vast and diverse datasets. By use of an autoencoder-based architecture to handle reconstruction calculations and employing multiple thresholds of error, our approach achieves exceptional performance in terms of both computation time and independent test evaluation metrics with a unique approach compared to state-of-the-art methods.

3 Materials and Method

3.1 Dataset Description

This research utilizes the NSL-KDD dataset, which is widely acknowledged as an upgraded version of the KDD-Cup'99 dataset, renowned in the field [24]. This dataset is particularly rich in information, comprising a total of 43 features. These features span different data types, including categorical, integer, binary, and real number values, which collectively contribute to enhancing the dataset's resilience and versatility in modeling various threat scenarios. In the dataset, samples are categorized into four main attack classes: 'DoS' (Denial of Service), 'Probe', 'R2L' (Remote to Local), and 'U2R' (User to Root), alongside a 'normal' class. However, for modeling the problem as a binary classification, we merge the four attack classes into a single 'intrusion' class. For a comprehensive breakdown of the distribution between these two categories, please consult Table 1.

3.2 Data Analysis

The dataset under consideration comprises of four types of features: categorical, binary, real, and integer. It consists of 3 categorical features, 6 binary features, 15 real features, and 17 integer features. Combining all these features together, a total of 41 unique features are present at our disposal for training and testing purposes. Table 2 provides a detailed description of our analysis from the NSL-KDD dataset.

Table 1: Training and Test Sample Distribution of NSL-KDD Dataset

Class	Training Sample	Test Sample
normal	67343	9711
anomaly	58630	12833
Total	125973	22544

3.3 Data Preprocessing

Data preprocessing is a fundamental aspect of constructing a robust machine-learning model. It encompasses several key steps aimed at preparing the dataset for effective analysis and training. Firstly, categorical attributes are encoded into numerical representations to ensure compatibility with machine learning algorithms. This transformation facilitates the algorithm's understanding of categorical data, which is essential for accurate model creation. Additionally, integer attributes undergo min-max normalization, which scales their values to a 0 to 1 range, preventing attributes with larger scales from overshadowing those with smaller scales during model training.

Moreover, a Zero-Statistical-based technique is integrated for data cleansing and feature reduction, identifying, and removing features or instances with zero statistical variance or minimal contribution to the model's predictive power. This process optimizes the dataset by focusing on informative features while reducing noise, ultimately enhancing the efficiency and effectiveness of the machine learning model. Subsequent sections of the paper will delve deeper into each preprocessing step, providing comprehensive insights into their implementation and impact on model performance.

3.3.1 Data Cleaning. During the dataset analysis, it was observed that a binary feature, namely 'su_attempted', contained a value of '2', which was inconsistent with the expected binary values. To maintain consistency and enhance model learning, this feature was adjusted to a value of '1'. Since, the rest of the features showed no irregularities, there was no need for further data cleaning.

3.3.2 Label Encoding. Label encoding was employed to convert categorical data into a format that could be easily interpreted by machine learning algorithms. This process involved assigning integer values to each category, starting from '0' and incrementing sequentially by an integer value of 1. Afterwards, this encoding scheme was then applied to the target variable 'class', ensuring consistency in representation across the dataset.

3.3.3 Normalization. To maintain uniformity within the dataset, we implemented min-max normalization on numeric integer attributes, following the formula outlined in Equation 1. This method adjusts each value to lie within the range between 0 and 1. This normalization method supports model training by maintaining consistent scaling across all features. Moreover, it's noteworthy that the real (continuous) features present in the NSL-KDD dataset had already been normalized to fit within the range of [0,1], thus

Table 2: Training and Test Attribute Distribution of NSL-KDD Dataset

Feature Set	Feature Names	Feature Counts
Categorical	protocol_type, service, flag	3
Binary	land, logged_in, root_shell, su_attempted, is_host_login, is_guest_login	6
Real	serror_rate, srv_serror_rate, rerror_rate, srv_rerror_rate, same_srv_rate, diff_srv_rate, srv_diff_host_rate, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_serror_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate	15
Integer	duration, src_bytes, dst_bytes, wrong_fragment, urgent, hot, num_failed_logins, num_compromised, num_root, num_file_creations, num_shells, num_access_files, num_outbound_cmds, count, srv_count, dst_host_count, dst_host_srv_count	17

eliminating the necessity for further adjustments to those specific values.

$$X_{new} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{1}$$

3.3.4 Intrusion Class Distribution. The NSL-KDD dataset contains numerous intrusion types in both the training dataset and the testing dataset. As mentioned in the dataset description, there are four main attack classes formed based on the subclasses presented in Table 3: 'DoS', 'Probe', 'R2L', and 'U2R'. However, for our specific task, every subclass from each dataset will be conjoined into a single 'intrusion' class. The priority will be to focus on if an instance is considered 'normal' or 'intrusion'.

3.4 Feature Selection

Feature selection is pivotal in refining features and improving model efficiency and efficacy. Among several methods, Zero Statistical Test-based feature selection is notable for identifying and removing features with minimal variability, particularly those dominated by zero values. In our study, we utilized this approach throughout our dataset, excluding feature vectors with 100% zero values. Consequently, 'num_outbounds_cmds' emerged as the only feature having no meaningful variation, prompting its removal from further analysis. This underscores our commitment to retaining only features with significant predictive power, thereby streamlining subsequent analysis steps, reducing dimensionality, and enhancing our model's interpretability. Our adoption of Zero Statistical Test-based feature selection significantly enhanced our model's efficiency and efficacy by effectively removing features with minimal informational value. To provide comprehensive insights into each selected feature's significance, Figure 1 offers detailed visualizations and explanations of the selection process. The figure displays a bar chart showing the ratio of '0' values (the number of samples for which a given feature has zero values divided by the total number of samples) across the 41 features in the KDDTrain+ dataset. Each bar represents an individual feature, with its height indicating the ratio of '0' values. The red bar represents the feature ('num_outbound_cmds') with the highest ratio of null values (100%) compared to the other features in the dataset. Thus, the 'num_outbound_cmds' feature was dropped.

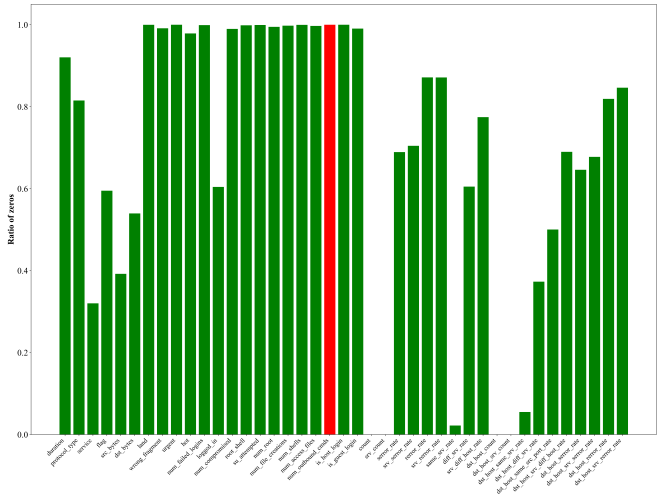

Figure 1: Null Value Distribution Across the 41 Features of the KDDTrain+ Dataset.

3.5 Autoencoder Based Intrusion Detection

In this study, we utilize a deep autoencoder as a powerful tool for unsupervised representation learning in the context of binary classification of network intrusions. The autoencoder is specifically trained to distinguish between normal sample representations and potential intrusions within the network traffic data. Central to this approach is the utilization of reconstruction error, serving as a pivotal indicator of anomaly or intrusion. During the testing phase, samples with reconstruction errors surpassing the predefined threshold of 0.0223 for minimum validation loss are identified as potential intrusions. This choice of threshold is strategically determined to strike a balance between sensitivity and specificity in intrusion detection.

The underlying principle driving this methodology lies in the assumption that intrusions present inherent complexities that hinder their accurate representation through the autoencoder's learned normal sample representations [1, 23]. Consequently, instances with elevated reconstruction errors are deemed likely to deviate significantly from the learned normal patterns, warranting their

Table 3: Intrusion Class Distribution of NSL-KDD Dataset

Dataset	Intrusion Types
Training	Neptune, Smurf, Back, Teardrop, Pod, Land, Satan, Ipsweep, Portsweep, Nmap, Warezclient, Guess_Passwd, Warezmaster, Imap, Ftp_Write, Multihop, Phf, Spy, Buffer_Overflow, Rootkit, Loadmodule, Perl
Testing	Neptune, Smurf, Apache2, Processtable, Back, Mailbomb, Teardrop, Pod, Land, Udpstorm, Mscan, Satan, Saint, Ipsweep, Portsweep, Nmap, Guess_Passwd, Ftp_Write, Warezmaster, Snmpguess, Snmpgetattack, Httptunnel, Multihop, Named, Sendmail, Xlock, Xsnoop, Imap, Phf, Worm, Buffer_Overflow, Ps, Rootkit, Xterm, Loadmodule, Perl, Sqlattack

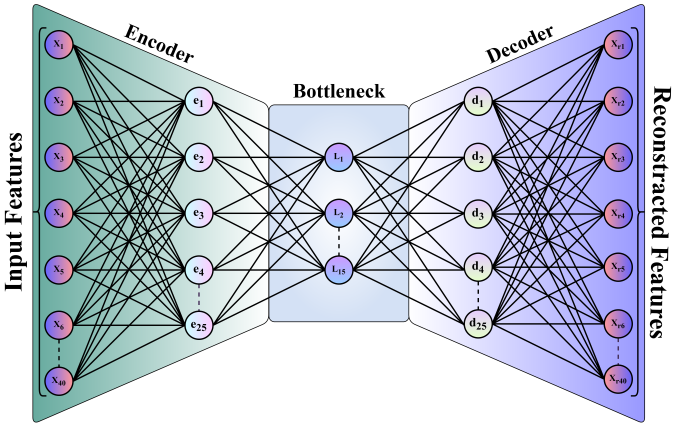

Figure 2: Architecture of the Proposed AI2DS Autoencoder.

classification as anomalies. This foundational understanding of reconstruction error is the core of our effective and robust intrusion detection approach. Figure 2 illustrates an autoencoder architecture used for binary class intrusion detection on the NSL-KDD dataset. It depicts the three primary components of the autoencoder: the Encoder, the Bottleneck, and the Decoder. The Encoder compresses the input features X_1 to X_{40} into a lower-dimensional space represented by nodes e_1 to e_{25}, capturing the most relevant information. The Bottleneck, the narrowest part of the architecture, further distills this information down to the nodes L_1 to L_{15}, which represent the most significant features. The Decoder then reconstructs the input data from these compressed features into the reconstructed features X_{r1} to X_{r40}, aiming to match the original input as closely as possible. This architecture is particularly useful for detecting anomalies or intrusions by learning to reconstruct normal data and identifying deviations in the reconstructed output for anomalous data.

3.6 Proposed Model

Figure 3 presents a comprehensive block diagram of the proposed Autoencoder-based intrusion detection system. The process commences with data collection, drawing from the KDDTrain+, and KDDTest+ datasets, known benchmarks in network intrusion detection. Subsequent data analysis bifurcates the dataset into feature sets—categorized under categorical (nominal) and numerical (binary, real, and integer)—and the target variable, which discerns 'Normal' network behavior from 'Anomaly' indicative of network intrusions. The data preprocessing phase encompasses label encoding for nominal features, extensive data cleaning, and feature elimination to streamline the input feature space, followed by Min-Max normalization to aid in model convergence. A crucial step involves labeling all attack vectors as 'Intrusion' to clearly define the target classes. In the intrusion detection phase, the model, trained exclusively on 'Normal' data samples, employs an autoencoder to determine the reconstruction loss, which serves as a critical metric for intrusion identification. A threshold for this reconstruction loss is established from the validation set, distinguishing normal instances from potential intrusions. During testing, the autoencoder computes the reconstruction loss for new data, classifying it as 'Normal' if the loss falls below the threshold, or 'Intrusion' if it meets or exceeds the threshold, thus achieving the primary objective of detecting anomalous behavior within network traffic. The proposed network intrusion detection method is presented as a pseudo-code in Algorithm 1.

Algorithm 1 Autoencoder-based Network Intrusion Detection Algorithm

Parameters: Normal dataset X, Test dataset q_i, $i = 1, \ldots, N$, threshold α defined by validation loss
Parameters: e_θ: Encoder, g_ϕ: Decoder
Output: reconstruction errors, network intrusion indicator

$g_0, e_0 \leftarrow$ train an Autoencoder with normal dataset X
LOOP Process
for $i = 1$ to N **do**
  reconstruction error $(i) = \frac{1}{N}\|q^{(i)} - g_0(e_0(q^{(i)}))\|^2$
  if reconstruction error $> \alpha$ **then**
    $q^{(i)}$ is an intrusion
  else
    $q^{(i)}$ is a normal (true) query
  end if
end for

3.7 Experimental Setup

The experiments were conducted meticulously on a high-performance computing system equipped with an Intel Core i5 7200U dual-core CPU and 32GB of RAM. Key Python libraries utilized included scikit-learn (v1.2.2) for model training and evaluation, matplotlib (v3.7.4) for insightful visualization of performance metrics, and pandas (v2.0.3) for seamless data manipulation and analysis. For constructing intricate autoencoders to precisely record the reconstruction error, Keras (v2.12.0) with a TensorFlow 2.12.0 in backend was utilized.

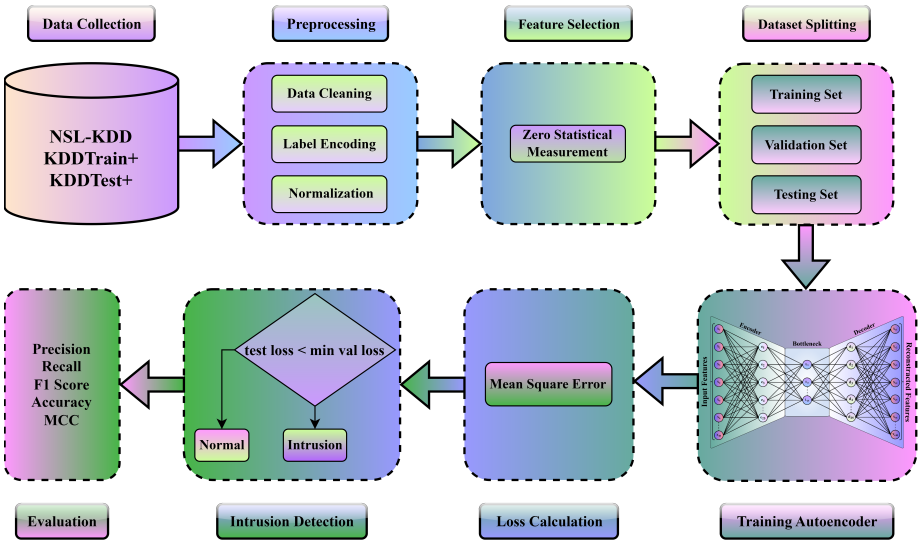

Figure 3: Detailed Block Diagram of Autoencoder-based Intrusion Detection System.

4 Result Analysis

4.1 Performance Evaluation Metrics

Precision assesses how accurately positive predictions are made by measuring the proportion of correctly predicted positive instances among all instances predicted as positive. Conversely, recall evaluates the model's capability to identify actual positive instances by calculating the percentage of correctly identified positive instances among all actual positives. The F1 score, which is the harmonic mean of precision and recall, offers a well-rounded evaluation of the model's effectiveness in binary classification tasks. Accuracy reflects the overall correctness of predictions, while the Matthews Correlation Coefficient (MCC) combines true positives, true negatives, false positives, and false negatives to assess classification performance, especially in datasets, where the number of samples in given classes may not be equally balanced. Initially, precision and recall are computed using equations 2 and 3. Finally, the F1 score, accuracy, and MCC are calculated using equations 4, 5, and 6, respectively. These metrics collectively provide a comprehensive assessment of the model's performance across various aspects of classification accuracy, F1 score, and MCC.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{4}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{5}$$

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{6}$$

Where, TP = number of samples correctly predicted as an anomaly; TN = number of samples correctly predicted as normal; FP = number of samples falsely predicted as an anomaly; FN = number of samples falsely predicted as normal.

4.2 Performance Evaluation

This section is organized into three subsections, each conveying the key phases of the study. First, the process of training the autoencoder is elaborated, encompassing the optimization of model parameters to efficiently reconstruct input data while also minimizing reconstruction error. Details such as hyperparameters, neural network optimizers, and training/validation strategies are elucidated to provide a comprehensive understanding of the model's training procedure. Subsequently, the evaluation of the proposed AI2DS model's performance on an independent test dataset is conducted to assess its generalization ability. Metrics like reconstruction loss and loss functions are employed to precisely gauge the model's performance, complemented by visual aids such as ROC curves and precision-recall curves. Finally, a comparative analysis with state-of-the-art studies in the field is undertaken to contextualize the findings and assess the novelty of the proposed approach. This involves identifying relevant literature, comparing methodologies, and the reported results from other studies to highlight the proposed model's advantages and potential contributions.

4.3 Training Autoencoder

The autoencoder architecture comprises a total of five layers, specifically structured to produce effective representation learning. This includes an input layer, followed by three hidden layers, and a final output layer. Among these layers, each plays a distinct role in the encoding and decoding process, comprising an encoder layer responsible for extracting essential features, a bottleneck layer for compressing the representation, and a decoder layer for reconstructing the original input space.

To ensure robust model performance, additional measures are implemented to promote regularization and counteract overfitting tendencies. Notably, the encoder includes 0.15 Gaussian Noise and a dropout rate of 0.05. These regularization techniques serve to introduce variability and prevent the model from becoming overly reliant on specific features or noise during training. Additionally, the activation function of choice for both encoder and decoder layers is the Scaled Exponential Linear Unit (SeLU), renowned for its ability to preserve the mean and variance of input data distributions, thereby facilitating stable learning dynamics.

The optimization of the autoencoder model is primarily based through the adoption of Root Mean Square Propagation (RMSprop) as the optimizer. With a learning rate set at 0.0035, this optimization strategy aims to navigate the model towards convergence while effectively balancing the trade-off between rapid learning and stability. Furthermore, a batch size of 32 is selected, alongside a 25% validation split, to ensure model performance is rigorously evaluated across diverse subsets of the data. Moreover, early stopping mechanisms are employed, with a patience level set at 6 steps, strategically designed to prevent overfitting and foster optimal convergence during the training process.

During the training phase, we tracked key performance indicators to assess the potential effectiveness of our model. This included continuous monitoring of both training loss and validation loss across multiple epochs. These metrics offer invaluable insights into the autoencoder's performance potential prior to its application on unseen test data. Refer to Figure 4 for a detailed visualization of the training dynamics of an autoencoder model over 10 epochs. This figure effectively captures the progression of both training and validation losses. Initially, the training loss starts higher but rapidly declines, showing significant improvement within the first few epochs. Similarly, the validation loss demonstrates initial volatility, peaking around the third epoch before decreasing and stabilizing. Both losses exhibit minor fluctuations but ultimately converge to remain consistently below 0.05, indicating that the model achieves a stable and low-error state as training progresses. This convergence suggests good generalization without substantial overfitting, as evidenced by the close alignment between the training and validation loss trends towards the end of the training process. Notably, the model achieved its best performance at the 9th epoch, as indicated by the black circle. At this point, the model attained the minimum validation loss of 0.0223, signifying the optimal set of weights stored during the training process. This indicates that the model was most effective in generalizing to unseen data at this epoch.

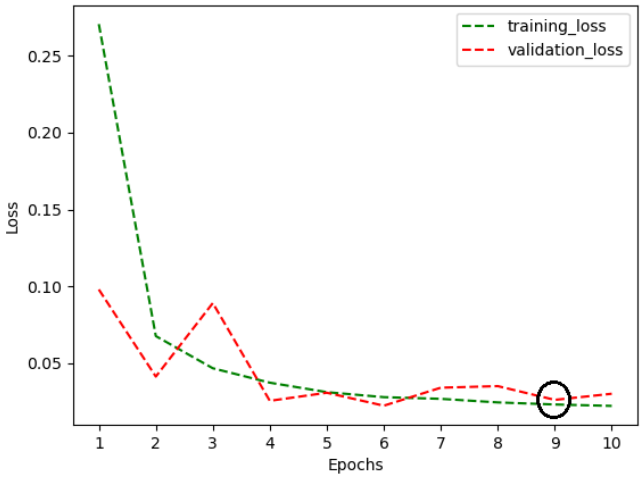

Figure 4: Training and Validation Loss During the Model Building.

4.4 Performance Analysis on Independent Test Dataset

The independent test dataset was utilized to assess the effectiveness of the proposed AI2DS model. The result of independent testing is presented in the form of a confusion matrix, depicted in Figure 5, enabling a comprehensive evaluation of the model's performance. In this representation, rows represent true classes, while columns depict predicted classes from the model. Each cell in the heatmap contains numerical values, with darker shades indicating a higher percentage of correct classification for a specific class. For instance, out of 12,833 samples, the model accurately classified 12,273 as class 'intrusion', and among 9,711 samples, 8,296 were correctly identified as class 'normal'. To gain a clearer understanding of the performance, we also included a normalized confusion matrix heatmap to display percentages rather than raw values from each class, as shown in Figure 6. Furthermore, the model's performance was assessed through Receiver Operating Characteristic (ROC) curves for each class, illustrated in Figure 7. As shown in Figure 7, the ROC curve achieves an outstanding value of 95%. The figure displays the ROC curve (green dashed line) for the autoencoder-based binary classifier that consistently began to rise until the 0.95 mark, indicating excellent performance in distinguishing between the two classes on an independent test set. However, relying solely on an ROC curve alone for comparing classifier performance is insufficient. The Precision-Recall curve, as depicted in Figure 8, illustrates the balance between precision and recall across various thresholds. A substantial area under the curve indicates both high precision and high recall. Here, high precision correlates with a minimal false positive rate, while high recall corresponds to a minimal false negative rate. The figure shows the precision-recall curve (green dashed line) for the classifier on the independent test dataset. The curve plots the trade-off between precision (y-axis) and recall (x-axis) at different classification thresholds, with an Area Under the Curve (AUC) of 0.96 or 96%, indicating strong performance in balancing these metrics.

4.5 Comparative Analysis

The proposed AI2DS model undergoes evaluation in comparison to several cutting-edge models with autoencoder and deep learning [10, 23, 29]. These models all tackle binary classification of intrusion detection using the well-established NSL-KDD dataset. Tauscher et. al. [23] employs a hierarchical strategy for binary classification, distinguishing between intrusion and normal behavior. Moreover, Kasongo [10] utilizes a specialized recurrent neural network (RNN) called XGBoost-LSTM capable of determining whether or not a sample is considered an intrusion based on its ability to

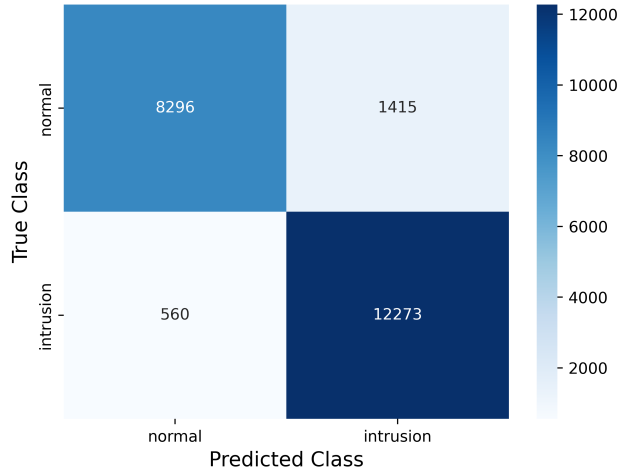

Figure 5: Confusion Matrix of Autoencoder-based Binary Classifier on the Independent Test Dataset.

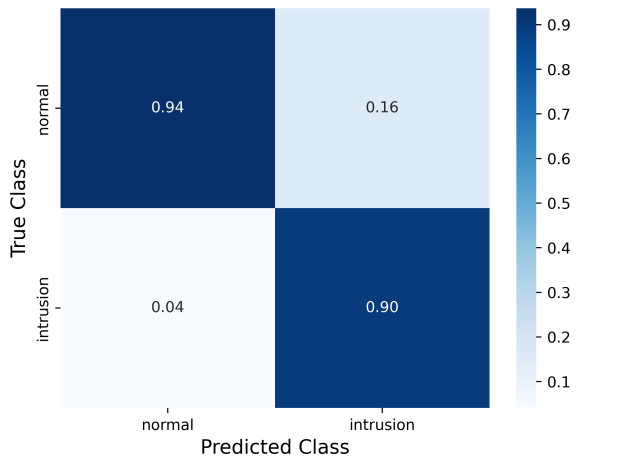

Figure 6: Normalized Confusion Matrix of Autoencoder-based Binary Classifier on the Independent Test Dataset.

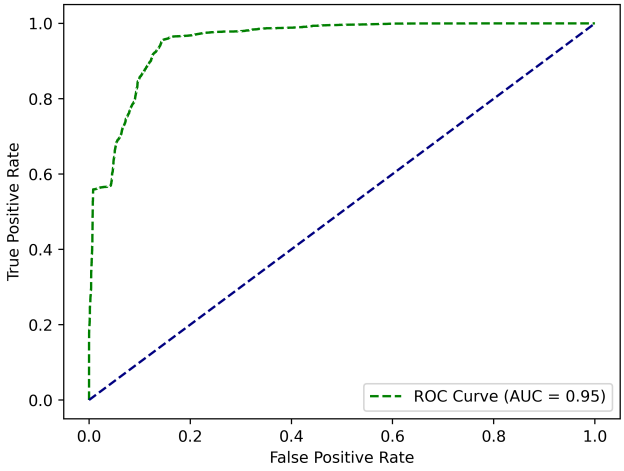

Figure 7: ROC Curve of Autoencoder-based Binary Classifier on the Independent Test Dataset.

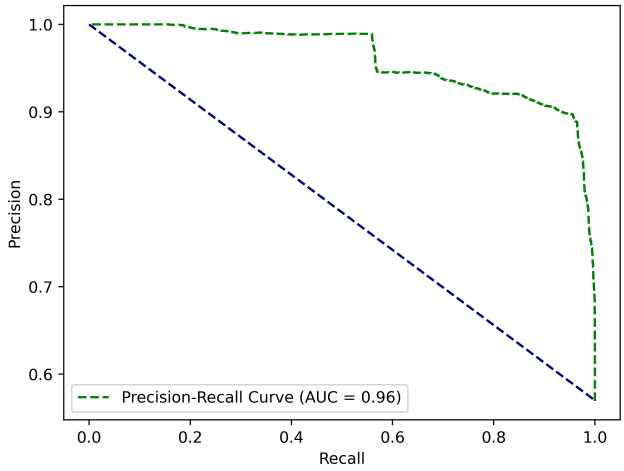

Figure 8: Precision-Recall Curve of Autoencoder-based Binary Classifier on the Independent Test Dataset.

retain important information regarding classification. Additionally, Yousefi et. al. [29] introduces an innovative unsupervised feature learning approach for network-based intrusion detection, employing an autoencoder that effectively learns feature representations. Our approach demonstrated significant improvements over these models, with average increases of 8.31% in precision, 25.08% in recall, 9.81% in F1 score, and 16.35% in accuracy. After examining Table 4, it's clear that our suggested model surpasses all three cutting-edge models based on the numbers demonstrated along with the provided context. This enhancement can be attributed to incorporating a deep autoencoder with a distinctive ability to differentiate between normal and intrusion based on the threshold value of reconstruction loss.

5 Conclusion

This study has explored an in-depth investigation into leveraging the capabilities of a traditional autoencoder architecture to fortify network security by enhancing the effectiveness of intrusion detection mechanisms. The principal contributions of this research revolve around the creation of a specialized deep autoencoder that is trained solely on normal network behavior. This well-crafted model demonstrates proficiency in reconstructing these behaviors with

Table 4: Comparative Performance Analysis of Proposed AI2DS Model on Different ML Classifiers.

Reference Work	Classifier	Precision	Recall	F1 Score	Accuracy	MCC
Tauscher et. al. [23]	Decision Tree (imp. %)	68.16 (34.08%)	83.09 (9.81%)	74.89 (21.75%)	68.28 (33.63%)	-
	Random Forest (imp. %)	87.34 (4.64%)	67.65 (34.87%)	76.24 (19.60%)	76.00 (20.05%)	-
	Naive Bayes (imp. %)	96.21 (-5.01%)	59.95 (52.19%)	73.87 (23.43%)	76.86 (18.71%)	-
	SVM (imp. %)	97.56 (-6.32%)	67.38 (35.41%)	79.71 (14.39%)	80.47 (13.38%)	-
	AdaBoost (imp. %)	86.90 (5.17%)	75.14 (21.43%)	80.59 (13.14%)	79.40 (14.91%)	-
	Gradient Boosting (imp. %)	65.04 (40.51%)	95.13 (-4.09%)	77.26 (18.02%)	68.12 (33.94%)	-
	MLP (imp. %)	95.82 (-4.62%)	63.96 (42.65%)	76.71 (18.86%)	77.90 (17.12%)	-
	Autoencoder (imp. %)	93.20 (-1.94%)	84.22 (8.34%)	88.48 (3.05%)	87.52 (4.25%)	-
Kasongo [10]	SimpleRNN (imp. %)	-	-	98.72 (-7.64%)	823.72 (8.98%)	-
	LSTM (imp. %)	-	-	99.58 (-8.44%)	88.13 (3.53%)	-
	GRU (imp. %)	-	-	99.38 (-8.25%)	85.7 (6.46%)	-
Yousefi-Azar et. al. [29]	Gaussian Naive Bayes (imp. %)	-	-	-	83.30 (9.53%)	-
	KNN (imp. %)	-	-	-	75.00 (21.65%)	-
	SVM (imp. %)	-	-	-	74.73 (22.09%)	-
	XGBoost (imp. %)	-	-	-	78.00 (16.97%)	-
AI2DS	**Autoencoder (avg. imp. %)**	**91.39 (8.31%)**	**91.24 (25.08%)**	**91.18 (9.81%)**	**91.24 (16.35%)**	**82.19(-)**

Note: Here, 'imp. %' represents the percentage enhancement by the proposed AI2DS method over the respective method, while 'avg. imp. %' denotes the average enhancement across listed methods. **(-)** represents the missing value.

precision and in discerning deviations that may indicate potential threats or anomalies lurking within the network environment.

Moreover, to further enhance the robustness of the intrusion detection system, an innovative adaptive thresholding technique has been devised and implemented. This pioneering approach capitalizes on the reconstruction error generated by the autoencoder, dynamically adjusting thresholds to accurately classify incoming network data as either normal or intrusion. This adaptive strategy ensures a responsive and adaptive intrusion detection mechanism capable of effectively safeguarding network integrity against emerging threats. Furthermore, the research streamlined the classification process by aggregating various attack types into a singular 'intrusion' category, simplifying the detection process and enhancing the system's applicability across diverse network environments. More specifically, the proposed AI2DS model results in an average improvement of 8.31%, 25.08%, 9.81%, and 16.35% based on precision, recall, F1 score, and accuracy respectively compared to the state-of-the-art methodologies. However, it is important to note that future improvements can be implemented such as reducing the percentage of false positives found in the normalized confusion matrix heatmap with more intricate deep learning architectures.

Looking forward, several avenues appear promising for extending the capabilities and applications of the autoencoder-driven approach to network security:

(1) **Integration with Real-Time Systems:** Future work could focus on integrating this model with real-time data streaming technologies to provide instantaneous intrusion detection and response. This would involve optimizing the model for lower latency and higher throughput to handle live network data efficiently.
(2) **Expansion to Multi-Class Classification:** While the current model simplifies the classification process by using a binary scheme, exploring multi-class classification could provide deeper insights into the nature of network threats, thereby facilitating more tailored and effective responses to different types of attacks.
(3) **Application of Deep Learning Techniques:** Further research could also explore the use of deeper and more complex autoencoder architectures, such as variational autoencoders (VAEs) or convolutional autoencoders, which might capture more intricate patterns and anomalies in network data.
(4) **Cross-Domain Adaptability:** Testing and adapting the model across different domains beyond network security, such as fraud detection in finance or fault detection in industrial systems, could significantly broaden the impact and utility of the research.
(5) **Enhancing Model Interpretability:** Another vital area of future research could focus on enhancing the interpretability of the autoencoder models. This involves developing techniques that can help explain why certain activities are flagged as intrusion, thus increasing the trust and usability of the models in security operations centers.

By addressing these areas, future research can not only refine the effectiveness of intrusion detection models but also expand their applicability and reliability, paving the way for more secure and resilient network infrastructures.

References

[1] Jinwon An and Sungzoon Cho. 2015. Variational autoencoder based anomaly detection using reconstruction probability. *Special lecture on IE* 2, 1 (2015), 1–18.

[2] A M Viswa Bharathy, N Umapathi, and S Prabaharan. 2019. An elaborate comprehensive survey on recent developments in behaviour based intrusion detection systems. In *Proceedings of the 2019 International Conference on Computational Intelligence in Data Science (ICCIDS)*. IEEE, Chennai, India, 1–5. https://doi.org/10.1109/ICCIDS.2019.8862119

[3] Vaishali Bhatia, Shabnam Choudhary, and KR Ramkumar. 2020. A comparative study on various intrusion detection techniques using machine learning and neural network. In *Proceedings of the 2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO)*. IEEE, Kolkata, India, 232–236. https://doi.org/10.1109/ICRITO48877.2020.9198008

[4] Amol Borkar, Akshay Donode, and Anjali Kumari. 2017. A survey on Intrusion Detection System (IDS) and Internal Intrusion Detection and protection system (IIDPS). In *Proceedings of the 2017 International conference on inventive computing and informatics (ICICI)*. IEEE, Coimbatore, India, 949–953. https://doi.org/10.1109/ICICI.2017.8365277

[5] Monika S Deshmukh and Pavan Ravikesh Bhaladhare. 2021. Intrusion Detection System (DBN-IDS) for IoT using Optimization Enabled Deep Belief Neural Network. In *Proceedings of the 2021 5th International Conference on Information Systems and Computer Networks (ISCON)*. IEEE, Mathura, India, 1–4. https://doi.org/10.1109/ISCON52037.2021.9702505

[6] Shanshuo Ding, Yingxin Wang, and Liang Kou. 2021. Network intrusion detection based on BiSRU and CNN. In *Proceedings of the 2021 IEEE 18th International Conference on Mobile Ad Hoc and Smart Systems (MASS)*. IEEE, Denver, CO, USA, 145–147. https://doi.org/10.1109/MASS52906.2021.00026

[7] Ouafae El Aeraj and Cherkaoui Leghris. 2023. Study of the SNORT intrusion detection system based on machine learning. In *Proceedings of the 2023 7th IEEE Congress on Information Science and Technology (CiSt)*. IEEE, Essaouira, Morocco, 33–37. https://doi.org/10.1109/CiSt56084.2023.10409876

[8] Suraj Gattani, Avdesh Mishra, and Md Tamjidul Hoque. 2019. StackCBPred: A stacking based prediction of protein-carbohydrate binding sites from sequence. *Carbohydrate research* 486 (2019), 107857. https://doi.org/10.1016/j.carres.2019.107857

[9] Gozde Karatas, Onder Demir, and Ozgur Koray Sahingoz. 2018. Deep learning in intrusion detection systems. In *Proceedings of the 2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT)*. IEEE, Ankara, Turkey, 113–116. https://doi.org/10.1109/IBIGDELFT.2018.8625278

[10] Sydney Mambwe Kasongo. 2023. A deep learning technique for intrusion detection system using a Recurrent Neural Networks based framework. *Computer Communications* 199 (2023), 113–125. https://doi.org/10.1016/j.comcom.2022.12.010

[11] Zakiyabanu S Malek, Bhushan Trivedi, and Axita Shah. 2020. User behavior pattern-signature based intrusion detection. In *Proceedings of the 2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*. IEEE, London, UK, 549–552. https://doi.org/10.1109/WorldS450073.2020.9210368

[12] Weizhi Meng. 2018. Intrusion detection in the era of IoT: Building trust via traffic filtering and sampling. *Computer* 51, 7 (2018), 36–43. https://doi.org/10.1109/MC.2018.3011034

[13] Avdesh Mishra, Pujan Pokhrel, and Md Tamjidul Hoque. 2019. StackDPPred: a stacking based prediction of DNA-binding protein from sequence. *Bioinformatics* 35, 3 (2019), 433–441. https://doi.org/10.1093/bioinformatics/bty653

[14] Safa Otoum, Burak Kantarci, and Hussein T Mouftah. 2019. On the feasibility of deep learning in sensor network intrusion detection. *IEEE networking letters* 1, 2 (2019), 68–71. https://doi.org/10.1109/LNET.2019.2901792

[15] Yazan Otoum, Vinay Chamola, and Amiya Nayak. 2022. Federated and transfer learning-empowered intrusion detection for IoT applications. *IEEE Internet of Things Magazine* 5, 3 (2022), 50–54. https://doi.org/10.1109/IOTM.001.2200048

[16] Devi Sri Prasad Puvvala, Gopichand Madala, Maheendra Kada, and U Hariharan. 2023. Improved Network Intrusion Detection System Using Deep Learning. In *Proceedings of the 2023 7th International Conference on Electronics, Materials Engineering & Nano-Technology (IEMENTech)*. IEEE, Kolkata, India, 1–6. https://doi.org/10.1109/IEMENTech60402.2023.10423459

[17] Panagiotis I Radoglou-Grammatikis and Panagiotis G Sarigiannidis. 2017. Flow anomaly based intrusion detection system for Android mobile devices. In *Proceedings of the 2017 6th International Conference on Modern Circuits and Systems Technologies (MOCAST)*. IEEE, Thessaloniki, Greece, 1–4. https://doi.org/10.1109/MOCAST.2017.7937625

[18] Md Habibur Rahman, Leo Martinez III, Avdesh Mishra, Mais Nijim, Ayush Goyal, and David Hicks. 2024. Enhancing Intrusion Detection through Deep Learning and Generative Adversarial Network. In *To appear at the 4. Interdisciplinary Conference on Electrics and Computer (INTCEC 2024)*. IEEE, Illinois, USA.

[19] Pedro Miguel Sánchez Sánchez, Alberto Huertas Celdrán, Gérôme Bovet, Gregorio Martínez Pérez, and Burkhard Stiller. 2022. Specforce: A framework to secure iot spectrum sensors in the internet of battlefield things. *IEEE Communications Magazine* (2022). https://doi.org/10.1109/MCOM.001.2200349

[20] Murtaza Ahmed Siddiqi and Wooguil Pak. 2022. Tier-based optimization for synthesized network intrusion detection system. *IEEE Access* 10 (2022), 108530–108544. https://doi.org/10.1109/ACCESS.2022.3213937

[21] V Sidharth and CR Kavitha. 2021. Network Intrusion Detection System Using Stacking and Boosting Ensemble Methods. In *Proceedings of the 2021 Third International Conference on Inventive Research in Computing Applications (ICIRCA)*. IEEE, Coimbatore, India, 357–363. https://doi.org/10.1109/ICIRCA51532.2021.9545022

[22] S Sridevi, R Prabha, K Narasimha Reddy, KM Monica, GA Senthil, and M Razmah. 2022. Network intrusion detection system using supervised learning based voting classifier. In *Proceedings of the 2022 International Conference on Communication, Computing and Internet of Things (IC3IoT)*. IEEE, Chennai, India, 01–06. https://doi.org/10.1109/IC3IOT53935.2022.9767903

[23] Zachary Tauscher, Yushan Jiang, Kai Zhang, Jian Wang, and Houbing Song. 2021. Learning to detect: A data-driven approach for network intrusion detection. In *Proceedings of the 2021 IEEE International Performance, Computing, and Communications Conference (IPCCC)*. IEEE, Austin, TX, USA, 1–6. https://doi.org/10.1109/IPCCC51483.2021.9679415

[24] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A Ghorbani. 2009. A detailed analysis of the KDD CUP 99 data set. In *Proceedings of the 2009 IEEE symposium on computational intelligence for security and defense applications*. IEEE, Ottawa, ON, Canada, 1–6. https://doi.org/110.1109/CISDA.2009.5356528

[25] Monika Vishwakarma and Nishtha Kesswani. 2023. A new two-phase intrusion detection system with Naïve Bayes machine learning for data classification and elliptic envelop method for anomaly detection. *Decision Analytics Journal* 7 (2023), 100233. https://doi.org/10.1016/j.dajour.2023.100233

[26] Treepop Wisanwanichthan and Mason Thammawichai. 2021. A double-layered hybrid approach for network intrusion detection system using combined naive bayes and SVM. *IEEE Access* 9 (2021), 138432–138450. https://doi.org/10.1109/ACCESS.2021.3118573

[27] Haipeng Yao, Pengcheng Gao, Peiying Zhang, Jingjing Wang, Chunxiao Jiang, and Lijun Lu. 2019. Hybrid intrusion detection system for edge-based IIoT relying on machine-learning-aided detection. *IEEE Network* 33, 5 (2019), 75–81. https://doi.org/10.1109/MNET.001.1800479

[28] Clinton Young, Joseph Zambreno, Habeeb Olufowobi, and Gedare Bloom. 2019. Survey of automotive controller area network intrusion detection systems. *IEEE Design & Test* 36, 6 (2019), 48–55. https://doi.org/10.1109/MDAT.2019.2899062

[29] Mahmood Yousefi-Azar, Vijay Varadharajan, Len Hamey, and Uday Tupakula. 2017. Autoencoder-based feature learning for cyber security applications. In *Proceedings of the 2017 International joint conference on neural networks (IJCNN)*. IEEE, Anchorage, AK, USA, 3854–3861. https://doi.org/10.1109/IJCNN.2017.7966342

[30] Guoling Zhang, Xiaodan Wang, Rui Li, Jie Lai, Qian Xiang, and Jiaxing He. 2020. Network Intrusion Detection Method Based on Stacked Denoising Sparse Autoencoder and Extreme Learning Machine. In *Proceedings of the 2020 2nd International Conference on Information Technology and Computer Application (ITCA)*. IEEE, Guangzhou, China, 194–199. https://doi.org/10.1109/ITCA52113.2020.00048