A Novel Lightweight Explainable Multilayer Adaptive RNN-Based Intrusion Detection Framework

Nidhi Srivastav[1], Rajiv Singh[2]
Department of Computer Science, Banasthali Vidyapith, Rajasthan, India [1, 2]
Centre for Artificial Intelligence, Banasthali Vidyapith, Rajasthan, India[2]

***Abstract*—A rapid increase in the instances of cyberattacks has been observed with the expanding digitization. This leads to an urgent and critical need for developing robust intrusion detection systems (IDS) which can identify the occurrences of malicious activities within the network traffic. The present work proposes a novel, explainable, multilayer, lightweight adaptive IDS based on a Recurrent Neural Network (RNN). The purpose of this proposed IDS is to improve threat detection capabilities, especially low frequency high severe attack. The performance of the proposed IDS is evaluated using the UNR-IDD dataset. The network traffic is classified into normal and attack categories to assess the effectiveness of the proposed IDS. Two separate IDS models are developed. Model A is used to detect attacks on the basis of the frequency of the attacks, and Model B detects threats based on the severity of the attacks. Through the layered approach, the overall detection accuracy of 95.7% is achieved in Model A, and 97.5% is obtained in Model B. The present work highlights that the proposed IDS shows a remarkable improvement in the detection of less frequent severe attacks in comparison to existing IDS. The comparative result analysis of RNN-based IDS with Machine Learning models such as LR, Naïve Bayes, Cat Boost, Random Forest and Multilayer Perceptron models shows RNN-based IDS has outperformed the Machine Learning models. Explainable AI (XAI), the SHAP method is used for better interpretation of the proposed decisions. XAI helps to identify the network traffic that can influence predictions and detect potential biases. It also helps researchers and practitioners to validate model behaviour and establish trust in the system's outputs.**

***Keywords*—IDS; network security; adaptive techniques; RNN; cybersecurity; explainable AI; UNR-IDD**

I. Introduction

With a rapid increase in the number of networks and internet usage, there has been a substantial rise in cybersecurity issues. Fig. 1 depicts the organization global weekly cyber-attack data. It shows that during the first quarter of 2025, cyberattacks increased noticeably. Each business saw 1,925 attacks on average per week, which is 47% increase over the same time period in 2024. Government, telecommunications, and education have become the most targeted industries as cybercriminals continue to improve their techniques [1]. Ransomware attacks rose by 126%, with North America accounting for 62% of global incidents, and Consumer Goods & Services being the most targeted sector [1]. The complexity and variety of modern cyberattacks pose a considerable challenge to traditional IDS methods. Traditional methods, such as anomaly-based detection, which identifies deviations from typical behavior, and signature-based detection, which depends on predefined attack signatures, frequently have drawbacks like high false-positive rates, a slow rate of adaptation to new attack types, and an inability to identify zero-day attacks [2].

With the increasing complexity of cyberattacks, it becomes difficult to detect low-occurrence and zero-day attacks. Recent years have seen several rare but high-impact cyberattacks that demonstrate the growing sophistication of threat actors. XZ Utils backdoor did widespread Linux server compromise [3]. ICS "blackhole" is an attack that disrupted control traffic without malware [4]. Real-world incidents include the FrostyGoop [5]. Malicious PyPI packages attempt credential theft in the supply-chain attack [6]. Another example of such an attack is TetraSoft's industrial ecosystem attacking the major energy sector [7]. Kyivstar telecom attack [8], which disrupted nationwide communications and alerts. All the above examples show that these events occur rarely but have large-scale operational and national consequences.

Generally, low-occurrence attacks are rare, but their per-incident impact is often experienced as disproportionately high. As in [9], the 2021 Colonial Pipeline ransomware attack affected the entire fuel supply chain, causing billions in economic damage. The North American Electric Reliability Corporation (NERC) identified events of coordinated cyberattacks on critical infrastructure, leaving severe consequences, despite their rarity [10]. According to IBM's Cost of a Data Breach report, low-frequency, high-impact attacks, like advanced persistent threats (APTs), usually target sensitive infrastructure like power grids or financial systems and worsen the damages [11].

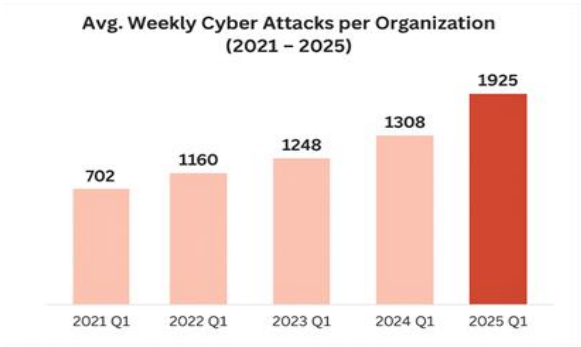

Fig. 1. Global weekly cyber attacks per organisation.

The rest of the study is structured as follows: Section II addresses relevant work about intrusion detection systems, with a particular emphasis on the use of deep learning and machine learning methods. Section III describes the methodology that introduces the proposed layered IDS framework. Section IV throws light on the architecture, steps of data preprocessing and

RNN-based IDS. Section V discusses experimental requirements, configuration of models, datasets, and metrics used for performance assessment and then analyses the results. Section VI summarizes the findings of this research.

II. Literature Review

Since the last decade, there has been significant research on Intrusion Detection Systems (IDS) based on Machine Learning (ML) algorithms. Researchers extensively used ML algorithms such as Decision Trees, Support Vector Machines (SVM), Random Forest, K-Nearest Neighbor (KNN) and Naïve Bayes for developing IDSs for decreasing false alarms [12], [13], [14]. Presently, deep learning techniques like Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), and Gated Recurrent Units (GRU) are used particularly for increasing anomaly detection rates due to their ability to identify sequential patterns [15] [16], [17]. With the increasing complexity, the traditional ML IDS often fail to detect complex attack patterns. Therefore, deep learning models are preferred over traditional ML models, due to Deep Learning classifiers and ensemble classifiers' ability to process an enormous amount of data and capture temporal dependencies [18], [19], [20]. Therefore, deep learning classifiers and ensemble classifiers are used to detect intrusions. However, several challenges remain, such as overfitting, high computational costs, and dataset dependency. Each examined study attempts to enhance the effectiveness and application of IDS by uniquely addressing these issues. Despite advancements in ML classifiers, IDS still come across challenges such as overfitting, high computational costs, and dataset dependency. Researchers have addressed these issues in their work to increase the effectiveness of IDS [21].

Prior literature review over the past few years, Gautam [22] introduced a composite approach combining LSTM and GRU architectures for improving IDS performance. The combination of LSTM and GRU enables the model to retain long-term dependencies and handle short-term events effectively. This hybrid approach helped to extract complex patterns in large-scale traffic data. The model was trained on CICIDS-2017, which comprised current attack types like DDoS, port scans and web attacks. In this model, a very high accuracy of 99.3% was reported. Ibrahim, 2023[23] modelled IDS with Long Short-Term Memory and Simple RNN. Ibrahim's study devised an IDS using a combination of simple RNN and LSTM, aiming to compare the performance of basic RNN with its enhanced LSTM counterpart. The primary emphasis was on evaluating the trade-offs between Simple RNN computational efficiency and LSTM accuracy. Ibrahim [23] used the KDD Cup dataset and achieved a 90.6% accuracy, which was lower than other deep learning approaches. The limitation of the study was to use an older dataset and a simpler RNN architecture. The focus was on evaluating the trade-offs between computational efficiency (Simple RNN) and accuracy (LSTM). Ibrahim uses KDD Cup dataset, and older but still used datasets for IDS research were used. The IDS model achieved a 90.6% accuracy, which is lower than other deep learning approaches. The limitation in accuracy was attributed to the older dataset and simpler RNN architecture. Fog-Cloud-Based IDS Using Bi-Directional LSTM and Simple RNN by Syed, 2023[24].

This study proposes fog-cloud computing-based IDS, comprising Simple RNN and Bi-Directional LSTM. The fog-cloud architecture is used to decentralize IDS by reducing the latency and improving real-time detection. Bi-Directional LSTM is found capable of processing data in both directions. It is used for capturing complex traffic patterns and was tested on the BoT-IoT dataset. A hybrid NID-Recurrent Neural Network (RNN) with Long Short-Term Memory (LSTM) was developed by Amutha et al., 2022 [25] to enhance detection capabilities. The study used the UNSW-NB18 dataset and found 8% increase in accuracy in comparison to the simple RNN-based models. Recently, Azarudeen, 2024 [26], developed an innovative model integrating LSTM-RNN with a hybrid sparse autoencoder and Deep Neural Network (DNN). The performance of IDS was found to be better than earlier methods in terms of accuracy, detection rate, and low false alarm rates. Adamax optimizer was used on the NSL-KDD dataset for multi-attack categorization. Network IDS developed through Attention Mechanisms and RNNs by [27] Djaidja, 2024. IDS developed by combining RNNs, LSTM and Gated Recurrent Units (GRU) for detecting network intrusions at an earlier. stage. During testing on the CICIDS-2017 and 5G-NIDD datasets, the model performed early-stage detection before the flow termination. Parveen, 2024 [28] presented a real-time intrusion detection system (IDS) on combining Convolutional Neural Networks (CNNs) and RNNs. Using the UNR-IDD dataset, the hybrid CNN-RNN model achieved a 96.2% accuracy with a low false alarm rate. Samriya, 2024 [29] presented a novel NIDS which comprises Support Vector Machines (SVM) and Extreme Gradient Boosting (XGBoost). The developed NIDS performance was improved through hyperparameter optimization using the Crow Search Algorithm. Rathee [30] examined a number of deep learning models, such as CNNs, shallow neural networks, and deep neural networks (DNN), with attention mechanisms. NIDS developed through Deep Learning techniques tested on NSL-KDD, Kyoto, and UNSW-NB15 datasets. NIDS achieved robust detection results across varying architectures. Mighan, 2021 [31] developed IDS from a hybrid approach consisting of stacked autoencoders for feature extraction, followed by traditional classifiers like SVM, random forests, and decision trees. The IDS was tested on the UNB ISCX 2012 dataset. The system showed scalability and high performance in real-time scenarios.

The study explored various feature selection approaches to enhance the effectiveness of ML-based IDS. [32] focused on optimizing deep learning models for attack classification through hyperparameter optimization methods such as grid search and random search. The accuracy of the model was 83.33% on the NSL-KDD dataset and 95.79% on the CSECIC-ID2018 dataset. Kasongo [33] proposed a layered framework for IDS, where each layer contributed to the detection of specific types of network intrusions. The novelty of this approach lies in the hierarchical structure of deep learning layers that help improve feature extraction. The framework combined multiple deep learning techniques (e.g., CNN, LSTM) in a pipeline for multi-stage detection. The datasets used are UNSW-NB15, a dataset containing contemporary attack scenarios. [34] presented a novel NIDS using a stacked non-symmetric deep autoencoder combined with an SVM classifier, the system achieved 99.65% by using KDD Cup'99 dataset. [35] proposed

a stack ensemble of tree-based boosting classifiers-XGBoost, LightGBM, and Cat-Boost-for classification.[36] presents a privacy-preserving hybrid machine learning model, AOPRF-XGBoost, designed to enhance intrusion detection in IoT systems. Prominent existing work and their findings are summarized in Table I.

TABLE I. EXISTING IDS TECHNIQUES

Author	Year	Methodology	Findings
Gautam et al. [22]	2022	Used LSTM and GRU with RNN.	Worked on CICIDS-2017, achieved 99.3% accuracy.
Ibrahim and Elhafiz [23]	2023	Use Long Short-Term Memory (LSTM) with Recurrent Neural Network (RNN) architecture.	Worked on KDD Dataset and achieved only 90.6% accuracy on testing dataset.
Amutha et al. [25]	2022	integrating NID-RNN with Long short-term memory (LSTM)	Used UNSW-NB18 data set. Accuracy increases by 8% as compared to simple RNN.
Parveen et al. [28]	2024	This model used CNNs,RNNs, and synthetic Neural Networks (ANNs),	Worked on UNR-IDD dataset achieve an accuracy of 96.2% with very less false alarm rate.
Samriya et al. [29]	2024	Use Support Vector Machine (SVM) and eXtreme Gradient Boosting (XGBoost) techniques.	Worked on NSL-KDD and UNR-IDD dataset.
Qazi et al. [32]	2022	utilized a stacked Non-symmetric deep auto encoder & support vector machine classifier	Accuracy of 99.65% is achieved at terms of increasing complexity
Kasongo et al. [33]	2022	Developed a framework where each layer contributes to data preprocessing, sequential pattern analysis, and a hierarchical model structure.	Used dataset UNSW-NB15, that improved both the detection accuracy and the interpretability of results.
P. R Buvaneswari, et al. [35]	2024	Uses stacked ensemble model	Accuracy is 98% but complexity of model increases

Although extensive research has been carried out in the field of intrusion detection using Artificial Intelligence and Machine Learning techniques, the literature indicates that only a limited number of studies have specifically focused on the detection of low-frequency yet high-impact (severe) attacks. Moreover, many of the existing systems that achieve high overall detection accuracy are often complex in nature, leading to increased computational cost and reduced real-time applicability. The literature reviewed shares several common challenges, including:

- Dataset limitations: The reliance on older or static datasets like NSL-KDD and KDD CUP'99 raises concerns about the ability of these models to generalize to modern network environments.
- Model complexity: Deep learning models are generally accurate, but often require significant computational resources, as in a stacked ensemble model, which is complex in terms of system design and management, and therefore, they are challenging to deploy in real-time, resource-constrained situations like cloud systems or the Internet of Things.
- Attack diversity: Many models are trained on specific datasets with a limited variety of attack types, which affects their ability to detect novel, zero-day, or obscure attacks.
- Detection of less frequent attacks: Although many systems achieve high overall accuracy, often reaching 96% to 97% but, they struggle to detect less frequent attack types despite achieving a good accuracy. This limitation arises because these rare attacks are underrepresented in the training data, leading to insufficient learning and poor detection performance for such cases.
- Detection of highly severe attacks: Many systems achieve good accuracy in detecting attacks, but the time required to detect all attacks is so high that sometimes it reduces the practical usefulness of attack detection.

The objectives of this research are as follows.

- To develop an IDS that uses a simple Recurrent Neural Network (RNN) within a layered architecture to enhance detection accuracy in less time and minimize false positives.
- To introduce a novel multi-layered IDS framework that combines data preprocessing, feature extraction, and RNN-based sequence modeling corresponding to specific attack type which helps to effectively detect low-frequency, zero–day–like behavior patterns by learning generalized representations of anomalous traffic rather than relying solely on predefined attack signatures.
- To highlight the advantages of the layered design, such as improved modularity, enhanced detection capabilities, and reduced computational overhead and detection time.
- To integrate Explainable Artificial Intelligence (XAI) techniques with black-box machine learning models to enhance model transparency by not only generating predictions but also providing interpretable explanations.

The proposed model focuses on creating more adaptable models that can handle large-scale and complex data using some simple, less complex ML/DL models, which are able to detect severe low-occurrence attacks with less computational overhead and are fast in terms of inference time, in spite of having few samples in the dataset.

III. PROPOSED METHODOLOGY

The suggested approach uses machine learning and deep learning classifiers in a multi-layer classification architecture to identify and stop network attacks. The approach is designed to optimize detection accuracy by categorizing threats based on their frequency of occurrence. The architecture is structured into two distinct layers, each tailored to a specific category of attacks: Layer 1 focuses on the detection of frequent

attacks/severe attacks, whereas Layer 2 is dedicated to identifying less frequent/ less severe attack types. This approach not only allows for a more systematic detection process but also helps in addressing challenges like high false-positive rates, real-time detection, and resource management.

The proposed model employs the UNR-IDD dataset for the detection of attacks. Authors in [37] suggested that UNR-IDD captures modern network behaviour and includes realistic network traffic, which is more representative of contemporary network environments. It is clearly seen from Table II that various attack categories (such as TCP-SYN, PortScan, Diversion and Blackhole) represented with enough samples, but instances of Overflow were very few in UNR-IDD, contributing only 2% of total instances and thus considered as a less frequent attack. It is very difficult to detect such attacks, as the models were not perfectly trained.

Data gathered from various sources [38], [39], [40], [41] has been summarized in Table III, which shows that Overflow and Blackhole attacks are among the most severe, although these attacks occur less frequently. As a result, there are fewer samples of these attacks in datasets, which makes it challenging to effectively train models to detect them. The limited number of samples hampers the model's ability to learn and generalize types of attacks. One potential solution is to synthetically generate samples to increase their representation in the dataset. However, this approach often leads to a decrease in the overall accuracy of the model.

To address this challenge, the proposed methodology employs a simple, low-complexity layered adaptive framework specifically designed to detect less frequent, severe attacks, as shown in Fig. 4. This method enhances the detection accuracy for Overflow attacks without compromising the overall performance of the model.

TABLE II. ATTACK TYPES AND POPULATION SIZE IN UNR-IDD

Attack Types	Population Size	% of attack
Blackhole	8420	22.5%
Diversion	5615	15.00%
Normal	3773	10.08%
Overflow	1022	2.7%
PortScan	9500	25.39%
TCPSYN	9081	24.27%

TABLE III. ATTACK SEVERITY AND FREQUENCY OF ATTACKS AVAILABLE IN UNR-IDD DATASET [38], [39], [40], [41]

Attack Types	Description	Severity	Frequency	Impact
TCP-SYN Flood	DoS attack that exploits the TCP handshake to flood a server	Moderate to High	high	Causes server overload, resource exhaustion, and denial of service
PortScan	Reconnaissance attack identifies open ports and services on a target system.	Moderate	Very High	provides information for further exploitation
Overflow	sends more data than a buffer can manage	Critical	low	System crash, remote code execution system compromise.
Blackhole	drop packets or reroute traffic, disrupting data flow.	Critical	moderate	Complete data loss disruption of comm. in networks.
Diversion	Traffic is misdirected leading to data theft or service disruption.	Moderate to High	high	Sensitive data exposure, potential service interruption,

IV. ARCHITECTURE OF ADAPTIVE PROPOSED FRAMEWORK

The proposed system is trained in a hierarchical manner similar to its detection workflow. In the offline training phase, as shown in Fig. 2, the dataset is first preprocessed and then divided into frequent/ severe and less-frequent/less-severe attack subsets. Separate feature extractors and classifiers are trained for each category.

During the detection process, incoming traffic is first evaluated using the frequent-attack classifier. After a frequent attack is detected, the traffic is further analyzed using the less-frequent attack classifier, as shown in the pseudocode of Fig. 3 and the architecture of the proposed layered framework in Fig. 4. This hierarchical training and detection strategy improves detection accuracy for rare and severe attacks while maintaining efficiency for common attack patterns.

The proposed adaptive framework works for two models:

Model A: Responsible for frequency-based attack detection.

Model B: Responsible for severity-based attack detection.

Both Models are based on a layered/hierarchical approach, where Layer 1 is responsible for detecting frequent/severe attacks, and Layer 2 is responsible for detecting less frequent / less severe attacks. Layer is designed specifically to detect less frequent and severe attacks. Despite the limited number of samples, it operates efficiently by focusing solely on these less common but critical threats. Records which remain unidentified after Layer 2 are considered as Normal.

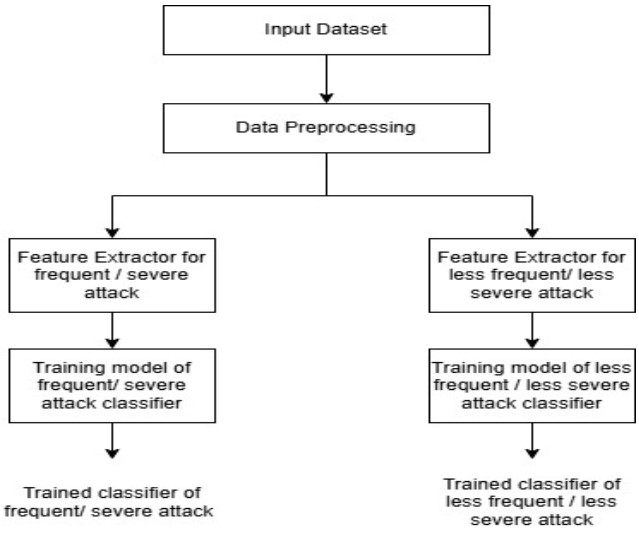

Fig. 2. Offline training process.

```
Algorithm Detect_Attack
    Input: dataset ← load_dataset("all_features")
    Output: attack_label ∈ {"Frequent Attack", "Less Frequent Attack", "Normal Traffic"}
    Begin
    // ---------------------------- Layer 1 ----------------------------
    // Step 1: Preprocess dataset for frequent attacks
    preprocessed_frequent ← data_preprocessor(dataset)
    // Step 2: Extract features relevant to frequent attacks
    features_frequent ← extract_features(preprocessed_frequent, type = "frequent")
    // Step 3: Classify using frequent attack model
    result_frequent ← classify (features_frequent, model = "frequent_attack_classifier")
    IF result_frequent == "Frequent Attack" THEN
    block traffic ()
    RETURN "Frequent Attack"
    ELSE
    Proceed to Layer 2
    END IF
    // ---------------------------- Layer 2 ----------------------------
    // Step 4: Preprocess dataset for less frequent attacks
    preprocessed_less_frequent ← data_preprocessor(dataset)
    // Step 5: Extract features for detecting rare or stealthy attacks
    features_less_frequent ← extract_features(preprocessed_less_frequent, type = "less_frequent")
    // Step 6: Classify using less frequent attack model
    result_less_frequent ← classify(features_less_frequent, model = "less_frequent_attack_classifier")
    IF result_less_frequent == "Less Frequent Attack" THEN
    block_traffic()
    RETURN "Less Frequent Attack"
    ELSE
    // ---------------------- Normal Traffic ----------------------
    // If not classified as either frequent or rare attack
    mark_as_normal()
    RETURN "Normal Traffic"
    END IF
    End
```

Fig. 3. Proposed methodology for detection of attacks.

The proposed framework consists of the following components, as shown in Fig. 4:

1) Data preprocessor: It is used to preprocess the data and can be divided into two phases:

a) Data cleaning: This involves handling missing, inconsistent, or inaccurate data entries. Techniques such as filling missing values (imputation), removing duplicates, and correcting errors are commonly used. In the proposed model, missing values were replaced with mean or median values, or specific rows or columns can be removed if they are unreliable.

b) Data transformation: Data representation and normalization processes are used to preprocess data on the adopted dataset. Data is represented using the one-hot encoding method. Then, the data normalization process can be done using Min-Max transformation, which transforms data into a format that suggested models can use more effectively. It normalizes data to a range (e.g., 0-1) and rescales data to have a mean of 0 and standard deviation of 1, which is frequently necessary for algorithms that are sensitive to feature magnitudes, such as neural networks and k-nearest neighbors.

2) Feature extractor: The purpose of the feature extractor component is to extract features from the dataset that are unique to the attack of a defined layer. Its main aim is to reduce the dataset for effective and smooth functioning in the era of high volume of traffic on the internet. Random Forest is a powerful technique for feature selection in machine learning. Random Forests inherently rank features based on their importance, allowing for the selection of the most relevant features for a given predictive task [42]. Overfitting can be minimized because the random forest construction process is random, and each decision tree is unrelated to the others. Random forest classifier assigns an importance score to each feature, based on which important features are selected as shown in Fig. 5(a) for Model A, high frequent attack and Fig. 5(b) for Model B, for high severe attack.

To enhance interpretability, SHapley Additive exPlanations (SHAP) were employed to quantify each feature's influence on the classifier's output. The SHAP framework distributes the model prediction among all input features according to their marginal contribution, providing a consistent and additive measure of importance. The SHAP bar graph, as shown in Fig. 6(a) and Fig. 6(b), shows each feature's average impact on the prediction. Longer bars indicate stronger influence, and the color indicates whether higher or lower feature values push the model toward normal or attack classes.

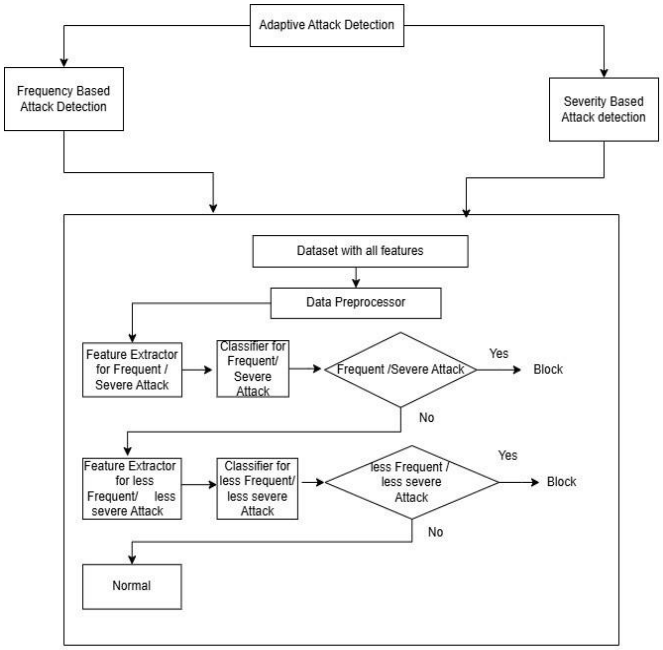

Fig. 4. Architecture of proposed layered framework for attack detection.

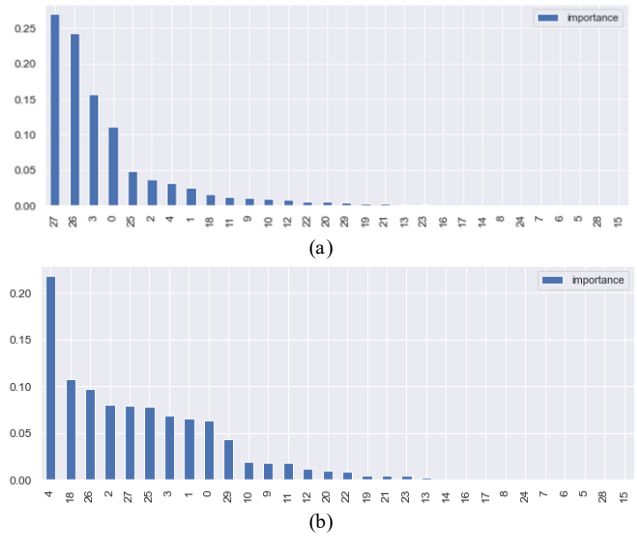

Fig. 5. (a). Features are sorted according to their importance score for high-frequency attack (Model A). (b). Features are sorted according to their importance score for high severe attack (Model B).

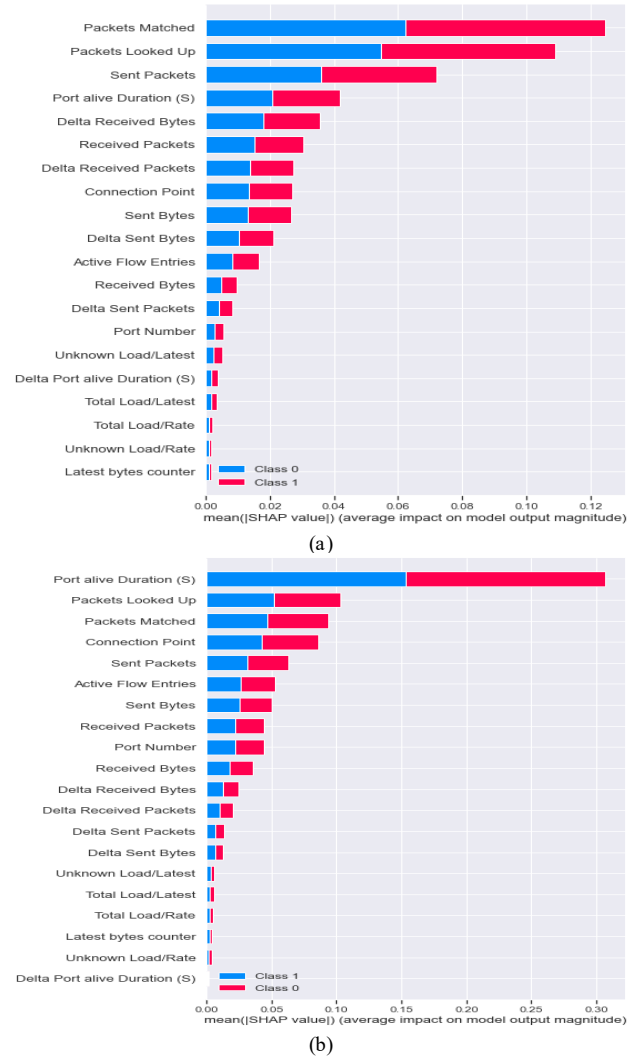

Fig. 6. (a). SHAP Graph corresponding to high Frequent Attack (Model A), (b). SHAP Graph corresponding to high Severe Attack (Model B).

According to the importance score and SHAP explanation following features are selected for Layer1 (Model A): Packets Matched, Packets Looked Up, Sent Packets, Received Packets, Active Flow Entries, Sent Bytes, Port Alive Duration (S) and Received Bytes. The features Active Flow Entries and Received Bytes were retained in the final model due to their strong summarizing capability and significant contribution to computational efficiency. Both attributes act as high-level aggregates that encode essential aspects of network behavior. Active Flow Entries and Received Bytes serve as compact, low-redundancy indicators that preserve predictive performance while substantially reducing model size, training epochs, and real-time inference latency.

Layer1(Model B): Port alive Duration (S), Sent Bytes, Connection Point, Packets Looked Up, Active Flow Entries, Packets Matched, Sent Packets, Received Bytes.

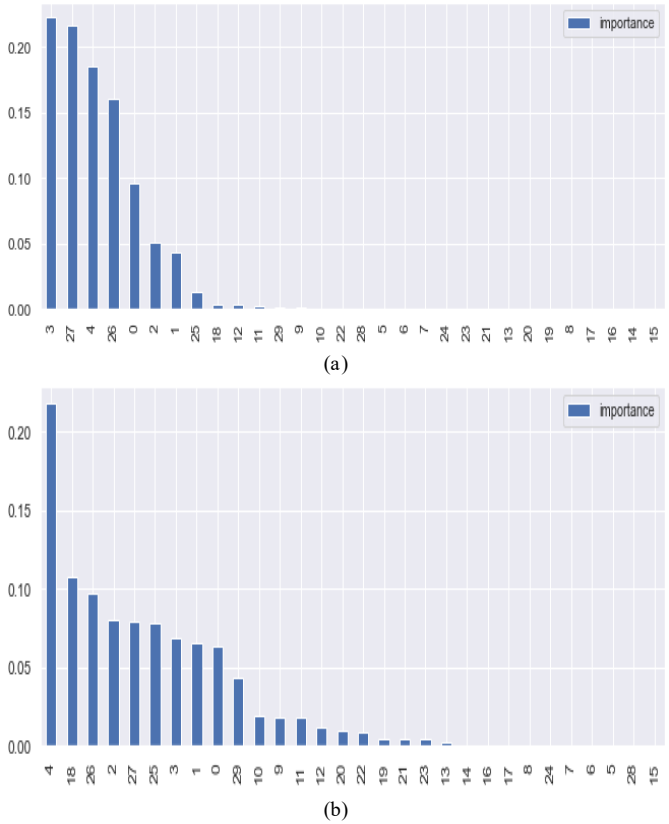

Fig. 7. (a). Features are sorted according to their importance score for low-frequency attack (Model A), (b). Features are sorted according to their importance score for a less severe attack (Model B)

According to importance score and SHAP graph as shown in Fig. 7(a), 7(b) and Fig. 8(a), 8(b), following features are selected for Layer 2 (Model A): Sent Packets, Packets Matched, Port alive Duration (S), Packets Looked Up, Received Packets, Sent Bytes, Received Bytes, Active Flow Entries.

Layer2 (Model B): Packets Matched, Packets Looked Up, Sent Packets, Port alive Duration (S), Received Packets, Sent Bytes, Received Bytes, Active Flow Entries.

A feature extractor is used at each layer so that important features corresponding to a specific attack can be extracted.

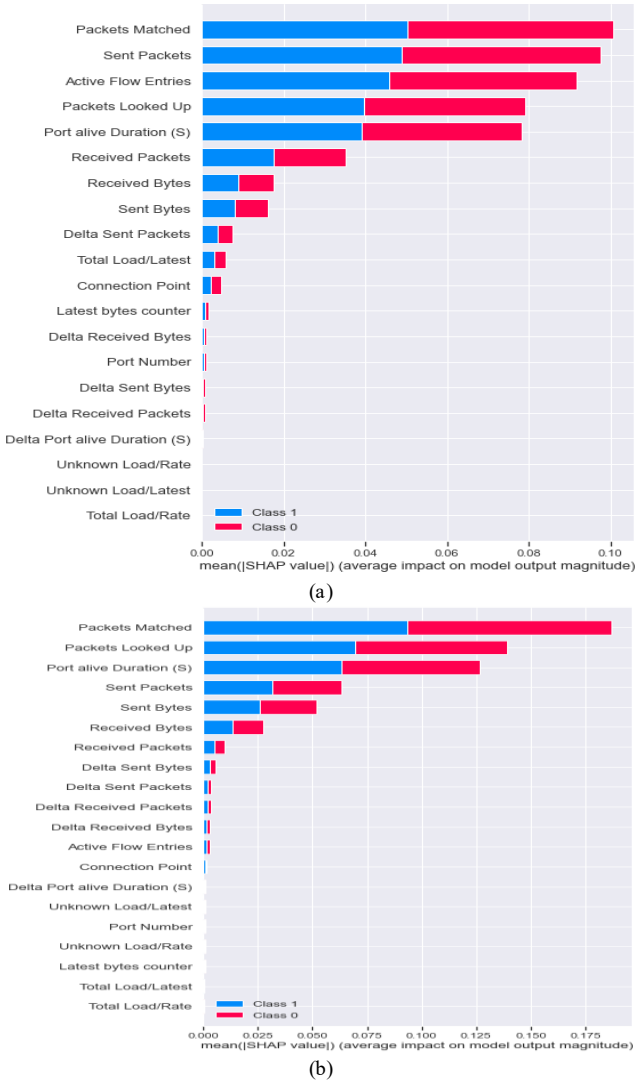

Fig. 8. (a). SHAP graph corresponding to less frequent attack (Model A), (b). SHAP graph corresponding to less severe attack (Model B)

3) Classifier: This component is used to classify records available in the dataset to detect attacks by analysing the training records in the dataset used. Here, the records are categorized as normal or assault using a Simple RNN. For applications like network traffic intrusion detection, Recurrent Neural Networks (RNNs), a kind of deep learning model built to handle sequential input, are ideal. RNNs are perfect for capturing the temporal relationships found in network traffic because, in contrast to standard feedforward neural networks, they have "memory" that enables them to remember information about prior inputs. Natural language processing and time series analysis have made extensive use of RNNs, and their capacity to represent sequential dependencies makes them a desirable option for intrusion detection systems. RNNs, especially simple RNNs, can use fewer parameters than some other networks, like CNNs with multiple layers, which often require more parameters for effective feature extraction. This can result in a lighter, more efficient model. Simple RNNs typically have a lower computational cost and memory footprint than deeper or more complex networks like CNNs or Transformers, which require more layers and heavier processing power, particularly on large datasets. Due to the simpler architecture, RNNs train faster on smaller datasets, where complex networks like CNNs could risk overfitting or require extensive regularization techniques.

V. Experimental Setup

Several tests are carried out in this section to evaluate the performance of the suggested model against the current model. A personal laptop with an Intel Core i7 processor and 16 GB of RAM serves as the development tool for the network intrusion detection technique suggested in this research. Python 3.9 is used to implement each classifier. The dataset used in this experimental setup is UNR-IDD. It is a prominent benchmark dataset containing 32 features. The proposed framework does an 80-20% split, which means 80% data is used for training and 20% is used for testing purposes. The suggested model employs Sigmoid activation (for binary classification) in the final layer, 64 units in the subsequent layer, and 128 units with ReLU activation in the first layer. Training uses a batch size of 32 and 20 epochs. The performance of the suggested model is compared using four evaluation metrics—accuracy, precision, recall, and F1 score—based on the confusion matrix.

Accuracy measures the proportion of correctly classified instances out of the total instances. This measure works well for datasets that are balanced, but not for unbalanced datasets

Precision is the ratio of accurately predicted positive cases to all anticipated positive instances.

Recall is the ratio of accurately anticipated positive instances to all actual positives and is known as recall (sensitivity). Although the model may contain some false positives, high recall indicates that it successfully detects positive events.

The F1 Score is calculated as the precision and recall harmonic means. Because it incorporates both accuracy and recall values, it is primarily utilized with datasets that are unbalanced.

Tables IV and V present the performance matrix of the proposed model in detecting attacks based on key metrics: Precision, Recall and F1 Score.

TABLE IV. Performance Matrix of Proposed Model A

Attack	Proposed Model A		
	Precision	***Recall***	***F1 Score***
Frequent Attack	0.976	1	0.988
Less Frequent Attack	1	0.235	0.38

TABLE V. Performance Matrix of Proposed Model B

Attack	Proposed Model A		
	Precision	***Recall***	***F1 Score***
Severe Attack	0.766	0.892	0.824
Less Severe Attack	0.965	0.916	0.940

Proposed Model A shows very high precision, recall and F1 score for the most frequent attack, that indicate good detection

performance in detecting frequent attacks. The proposed Model B achieves high recall for severe attacks, which ensures that most critical threats are detected in an efficient way, and the model also shows very high precision and F1-score for less severe traffic, which represents good overall classification performance with fewer false alarms.

Fig. 9 & Fig. 10 represents summarized rule sheet for identifying frequent and severe attacks, respectively. This rule sheet functions as a verification mechanism for evaluating the model's output. By systematically comparing model predictions with predefined rules, it becomes possible to identify inconsistencies, incorrect classification patterns, and potential sources of error such as misclassification, missing feature contributions, or signs of overfitting. This comparison process further helps in understanding how the model distinguishes between attack and benign classes, thereby offering clearer insight into the decision logic behind each prediction.

In [43] focuses on evaluating various machine learning models for intrusion detection using the UNR-IDD Intrusion Detection Dataset. Table VI denotes the Performance matrix of existing classifiers [43].

Table VII denotes the Performance Matrix of Existing Classifiers for detecting severe and less severe attacks.

Following Table VIII presents a comparative analysis of existing classifiers [43] against the proposed classifier based on F1-Score performance for both frequent and less frequent attacks.

```
Deeper Rule Tree (True Labels):

|--- Packets Matched <= 1.63
|   |--- Packets Looked Up <= -0.40
|   |   |--- Sent Bytes <= -0.71
|   |   |   |--- Received Packets <= -0.33
|   |   |   |   |--- Port alive Duration (S) <= -0.71
|   |   |   |   |   |--- Sent Bytes <= -0.71
|   |   |   |   |   |   |--- Packets Looked Up <= -0.43
|   |   |   |   |   |   |   |--- class: 0.0
|   |   |   |   |   |   |--- Packets Looked Up >  -0.43
|   |   |   |   |   |   |   |--- Packets Matched <= -0.42
|   |   |   |   |   |   |   |   |--- class: 0.0
|   |   |   |   |   |   |   |--- Packets Matched >  -0.42
|   |   |   |   |   |   |   |   |--- class: 0.0
|   |   |   |   |   |--- Sent Bytes >  -0.71
|   |   |   |   |   |   |--- class: 1.0
```

Fig. 9. Summarized Rule sheet to identify frequent attacks and normal traffic.

```
Deeper Rule Tree (True Labels):

|--- Port alive Duration (S) <= -0.63
|   |--- Packets Matched <= -0.41
|   |   |--- Sent Packets <= -0.38
|   |   |   |--- Packets Matched <= -0.42
|   |   |   |   |--- Packets Looked Up <= -0.43
|   |   |   |   |   |--- class: 1.0
|   |   |   |   |--- Packets Looked Up >  -0.43
|   |   |   |   |   |--- Packets Looked Up <= -0.43
|   |   |   |   |   |   |--- class: 0.0
|   |   |   |   |   |--- Packets Looked Up >  -0.43
|   |   |   |   |   |   |--- Received Packets <= -0.33
|   |   |   |   |   |   |   |--- Port alive Duration (S) <= -0.80
|   |   |   |   |   |   |   |   |--- class: 1.0
```

Fig. 10. Summarized Rule sheet to identify severe attack and normal traffic.

TABLE VI. PERFORMANCE MATRIX OF EXISTING CLASSIFIER[43]

	Logistic Regression			Naive Bayes			Multilayer Perceptron		
	Precision	***Recall***	***F1 score***	***Precision***	***Recall***	***F1 score***	***Precision***	***Recall***	***F1 score***
Frequent	0.43	0.45	0.39	0.66	0.59	0.59	0.53	0.52	0.46
Less Frequent	0	0	0	0.29	0.3	0.3	0.66	0.22	0.33

TABLE VII. PERFORMANCE MATRIX OF EXISTING CLASSIFIER FOR DETECTING SEVERE AND LESS SEVERE ATTACK

	Logistic Regression			Naive Bayes			Multilayer Perceptron		
	Precision	***Recall***	***F1 score***	***Precision***	***Recall***	***F1 score***	***Precision***	***Recall***	***F1 score***
Severe	0.31	0.19	0.23	0.54	0.44	0.48	0.41	0.07	0.24
Less Severe	0.44	0.51	0.47	0.62	0.66	0.64	0.52	0.58	0.69

TABLE VIII. COMPARATIVE ANALYSIS OF EXISTING CLASSIFIER & PROPOSED MODEL A

	Frequent Attacks			Less Frequent Attacks		
	F1 score	***F1 Score (Model A)***	***% increase***	***F1 score***	***F1 Score (Model A)***	***% increase***
Logistic Regression	0.397	0.988	148.6	0.1	0.380	38.03
Naive Bayes	0.592		66.78	0.3		26.76
MultiLayer Perceptron	0.467		111.38	0.33		15.24

TABLE IX. COMPARATIVE ANALYSIS OF EXISTING CLASSIFIER & PROPOSED MODEL B

	Severe Attacks			Less Severe Attacks		
	F1 score	***F1 Score (Model B)***	***% increase***	***F1 score***	***F1 Score (Model B)***	***% increase***
Logistic Regression	0.23	0.824	258.26	0.47	0.940	100
Naive Bayes	0.48		71.66	0.64		46.875
MultiLayer Perceptron	0.24		243.33	0.69		36.231

As it is evident from Table VIII and Table IX that the F1 Score for frequent, severe, less frequent and less severe are increased manifoldly, and it is graphically shown in Fig. 11.

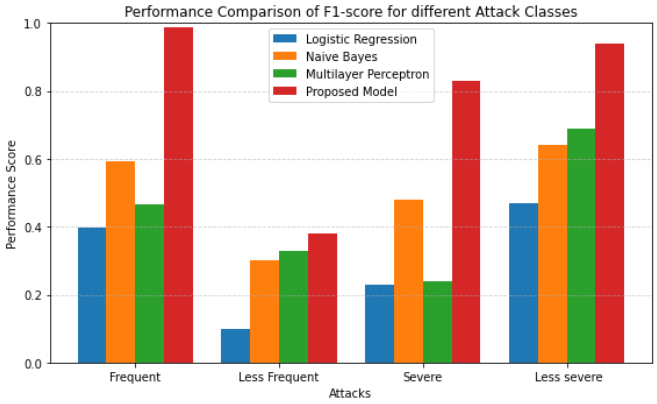

Fig. 11. Comparison of F1 score for different attack types.

In any Intrusion Detection System (IDS), the percentage of attacks successfully blocked is a critical performance metric. In the proposed model A, attacks are categorized into two types: frequent and less frequent. The performance of each detection layer is evaluated separately and compared against the existing model. As presented in Table X proposed Model A demonstrates significantly improved performance, blocking 99.84% of frequent attacks 85.7% of less frequent attacks.

TABLE X. COMPARISON OF EXISTING CLASSIFIER WITH PROPOSED MODEL A ON THE BASIS OF BLOCKING OF ATTACK

Classifiers	% of Frequent Attack Blocked	% of less Frequent Attack Blocked
KNN	82%	46%
Naïve Bayes	59.50%	30%
Decision Tree	93.25%	84%
Random Forest	94.25%	77%
CatBoost	96.25%	85%
XGBoost	97%	93%
Multi Layer Perceptron	52.75%	22%
Proposed Model	**99.84%**	**85.91%**

Similarly, Table XI represents proposed Model B blocks 99.8% of severe attacks and 96.82% of less severe attacks. This clearly highlights the effectiveness and robustness of the proposed model over existing approaches.

Even when considering the Accuracy metric, the proposed model outperforms existing machine learning, deep learning, and ensemble classifiers. Table XII indicate comparative analysis of Ensemble ML classifiers like CatBoost, RF and ET [43] and deep learning classifiers like CNN, RNN and ANN [44] with the proposed classifier, and it is clearly evident that the accuracy of the proposed classifier is higher than classifier in [43] [44].

Table XII indicates that Tree-based ensemble models (RF, ET) perform strongly, indicating the dataset benefits from decision boundaries learned via tree structures. Neural models like RNN also perform well due to sequential data handling. The proposed models, slightly better than the rest, likely incorporate advanced techniques or model fusion to maximize accuracy. It may be possible that the overall Accuracy of ML/DL models may be high, but the system is not able to handle less frequent attacks, which are severe too. To effectively detect less frequent and severe attacks and enhance overall accuracy, a Layered Framework is employed. Layered Framework help to achieve a high level of accuracy even with a simple RNN kind of model, as shown in Table XII. Ensemble classifier XGBoost performs better than Proposed Model A, but this performance comes at the cost of adding complexity in terms of memory used to store the model and the latency of predictions, as shown in Table XIII.

TABLE XI. COMPARISON OF EXISTING CLASSIFIER WITH PROPOSED MODEL B ON THE BASIS OF BLOCKING OF ATTACK

Classifiers	% of Severe Attack Blocked	% of less Severe Attack Blocked
KNN	80.35%	81.05%
Naïve Bayes	43.50%	66.2%
Decision Tree	96.7%	90.8%
Random Forest	96%	92.4%
CatBoost	97.6%	94.6%
XGBoost	98.4%	95.4%
Multi Layer Perceptron	66%	68.2%
Proposed Model	**99.8%**	**96.82%**

TABLE XII. COMPARATIVE ANALYSIS OF ENSEMBLE ML AND DL MODELS WITH THE PROPOSED MODEL

	Model	Accuracy
Ensemble ML Classifier	CatBoost [43]	91.18
	Random Forest [43]	95.12
	Extra Tree [43]	95.23
	XGBoost[43]	96.30
Deep Learning Models	CNN [44]	92.8
	ANN [44]	91.7
	RNN [44]	94.1
Proposed Model	**RNN with layers for frequency based attack detection**	**95.7**
	RNN with layer for severity based attack detection	**97.65**

TABLE XIII. COMPARATIVE ANALYSIS OF THE COMPLEXITY OF EXISTING MODELS WITH THE PROPOSED MODEL

Model Classifier	Model Size	Latency ms per prediction
XGBoost	1.54 MB	0.828
CatBoost	2.99 MB	0.440
RNN (Proposed Model A)	0.064 MB	0.8712
RNN (Proposed Model B)	0.0602 MB	0.535

Table XIII shows that the RNN achieves good performance on the UNR IDD dataset in terms of model size and latency. This advantage primarily stems from its lightweight architecture.

Overall, while CatBoost demonstrates superior speed in terms of latency, it comes at the cost of a much larger model size. The proposed RNN offers an optimal trade-off between ultra-compact model size and acceptable real-time latency, making it highly suitable for deployment in real-time, resource-constrained intrusion detection systems.

As highlighted in the literature review, numerous existing classifiers demonstrate high accuracy; however, they often depend on complex architectures, such as deep convolutional neural networks or hybrid models. On the other hand, lightweight classifiers usually have lower accuracy and perform poorly in detecting less frequent but critical attacks. The proposed classifier, conversely, achieves distinguished performance as shown by the results obtained by maintaining a simple architecture and employing a minimum set of features. While many hybrid models achieve similar or even higher accuracy, their complexity restricts real-time applications, especially in IoT and fog cloud environments. In the proposed model, employing even just eight features, a simple RNN provides efficient results even for low-frequency attacks. Thus, a layered framework facilitates a simple RNN to perform efficiently with fewer features, providing high efficiency and promising results. The contribution of this work lies in the unified integration of layered lightweight detection, adaptive feature selection, and explainable analysis specifically designed for low-frequency high-severity attack scenarios.

VI. Conclusion

The proposed model signifies a significant advancement in Intrusion Detection Systems (IDS) by implementing a layered architectural framework along with an RNN to improve the identification of low-frequency and rare cyberattacks. A major challenge in intrusion detection systems (IDS) is developing models that can be generalized across varied datasets. As model performance mainly depends on the nature of the input data, it becomes essential to evaluate the proposed model on multiple benchmark datasets. The proposed model achieves an overall accuracy of 95.7% (Model A) and 97.5% (Model B) on the UNR-IDD dataset. However, there is still scope for further improvement in the performance of the proposed IDS. Future work should be focused on minimizing cost, losses, and errors to further improve accuracy, precision, recall, and F1-score through trade-offs between model complexity, effectiveness, and computational efficiency. In addition to this, future research work should have an emphasis on improving generalization capabilities and developing efficient real-time applications for detecting zero-day and novel attacks. Future work will also focus on extending the proposed intrusion detection framework toward real-time deployment and stress-testing scenarios to evaluate its practical applicability in dynamic network environments.

References

[1] Check Point Software Technologies Ltd. “Security Blog.” Check Point, 4 May 2025, https://blog.checkpoint.com/security.

[2] Talukder, M. A., et al. “Machine Learning-Based Network Intrusion Detection for Big and Imbalanced Data Using Oversampling, Stacking Feature Embedding and Feature Extraction.” *Journal of Big Data*, vol. 11, no. 1, 2024, p. 33.

[3] Invicti. (2024). XZ Utils backdoor: Invicti Web Security Blog. https://www.invicti.com/blog/web-security/xz-utils-backdoor-supply-chain-rce-that-got-caught.

[4] G. Mitra, P. Dash, Y. E. Yao, A. Mehta, and K. Pattabiraman,“ICS-Sniper: A Targeted Blackhole Attack on Encrypted ICS Traffic,”arXiv preprint arXiv:2312.06140, Dec. 2023.

[5] MITRE ATT&CK, “FrostyGoop,” Software ID S1165. [Online].Available: https://attack.mitre.org/software/S1165/

[6] The Hacker News. (2025). https://thehackernews.com/2025/ 08/ researchers-spot-xz-utils-backdoor-in.html

[7] Reuters, “Cyberattack disrupts heating services in Ukraine,” 2024.

[8] Wikipedia contributors, “2023 Kyivstar cyberattack,” Wikipedia, The Free Encyclopedia, 2024. [Online]. Available: https://en.wikipedia.org/wiki/2023_Kyivstar_cyberattack

[9] *Varonis.* “Cybersecurity Statistics.” *Varonis Blog*, 4 May2025, https://www.varonis.com/blog/cybersecurity-statistics.

[10] North American Electric Reliability Corporation (NERC), and U.S. Department of Energy (DOE). *High-Impact, Low-Frequency Event Risk to the North American Bulk Power System*. Summary Report, June 2010.

[11] *IBM.* “Threat Hunting.” *IBM*, 4 May 2025, https://www.ibm.com/topics/threat-hunting.

[12] Hosseini, S., and M. Azizi. “The Hybrid Technique for DDoS Detection with Supervised Learning Algorithms.” *Computer Networks*, vol. 158, 2019, pp. 35–45.

[13] Abrar, I., et al. “A Machine Learning Approach for Intrusion Detection System on NSL-KDD Dataset.” , International Conference on Smart Electronics and Communication (ICOSEC), IEEE, 2020.

[14] Khraisat, A., et al. “Hybrid Intrusion Detection System Based on the Stacking Ensemble of C5 Decision Tree Classifier and One Class Support Vector Machine.” *Electronics*, vol. 9, no. 1, 2020, p. 173.

[15] El-Shafeiy, E., et al. “Deep Complex Gated Recurrent Networks-Based IoT Network Intrusion Detection Systems.” *Sensors*, vol. 24, no. 18, 2024, p. 5933.

[16] Seniaray, Sumedha, and Rajni Jindal. "Performance Analysis of Anomaly-Based Network Intrusion Detection Using Feature Selection and Machine Learning Techniques." *Wireless Personal Communications* 138.4 (2024): 2321-2351.

[17] Kukartsev, Vladislav, et al. "Using machine learning techniques to simulate network intrusion detection." 2024 International Conference on Intelligent Systems for Cybersecurity (ISCS). IEEE, 2024.

[18] Malathy, M., et al. “Enhancing Security Through the Implementation of Deep Learning Techniques GRU and LSTM for Intelligent Intrusion Detection.” *2024* International Conference on Computing and Intelligent Reality Technologies (ICCIRT), IEEE, 2024.

[19] Vadisetty, Rahul, and Anand Polamarasetti. "Enhancing Intrusion Detection Systems with Deep Learning and Machine Learning Algorithms for Real-Time Threat Classification." 2024 Asian Conference on Intelligent Technologies (ACOIT*)*. IEEE, 2024.

[20] Udurume, Miracle, Vladimir Shakhov, and Insoo Koo. "Comparative Analysis of Deep Convolutional Neural Network—Bidirectional Long Short-Term Memory and Machine Learning Methods in Intrusion Detection Systems." Applied Sciences 14.16 (2024): 6967.

[21] Kasongo, S. M. “A Deep Learning Technique for Intrusion Detection System Using a Recurrent Neural Networks Based Framework.” *Computer Communications*, vol. 199, 2023, pp. 113–125.

[22] Gautam, S., et al. “A Composite Approach of Intrusion Detection Systems: Hybrid RNN and Correlation-Based Feature Optimization.” *Electronics*, vol. 11, no. 21, 2022, p. 3529.

[23] Ibrahim, M., and R. Elhafiz. “Modeling an Intrusion Detection Using Recurrent Neural Networks.” *Journal of Engineering Research*, vol. 11, no. 1, 2023, p. 100013.

[24] Syed, N. F., M. Ge, and Z. Baig. “Fog-Cloud Based Intrusion Detection System Using Recurrent Neural Networks and Feature Selection for IoT Networks.” *Computer Networks*, vol. 225, 2023, p. 109662.

[25] Amutha, S., et al. “Secure Network Intrusion Detection System Using NID-RNN Based Deep Learning.” *2022 International Conference on*

Advances in Computing, Communication and Applied Informatics (ACCAI), IEEE, 2022.

[26] Azarudeen, K., et al. "Intrusion Detection System Using Machine Learning by RNN Method." *E3S Web of Conferences*, vol. 491, 2024, p. 04012.

[27] Djaidja, T. E. T., et al. "Early Network Intrusion Detection Enabled by Attention Mechanisms and RNNs." *IEEE Transactions on Information Forensics and Security*, 2024.

[28] Parveen, F., et al. "Real-Time Intrusion Detection with Deep Learning: Analyzing the UNR Intrusion Detection Dataset." *Journal of Computing and Biomedical Informatics*, vol. 7, no. 2, 2024.

[29] Samriya, J. K., et al. "Machine Learning Based Network Intrusion Detection Optimization for Cloud Computing Environments.", IEEE Transactions on Consumer Electronics (2024).

[30] Rathee, Ashish, Parveen Malik, and Manoj Kumar Parida. "Network Intrusion Detection System Using Deep Learning Techniques." 2023 International Conference on Communication, Circuits, and Systems (IC3S), IEEE, 2023.

[31] Mighan, Soosan Naderi, and Mohsen Kahani. "A Novel Scalable Intrusion Detection System Based on Deep Learning." *International Journal of Information Security*, vol. 20, no. 3, 2021, pp. 387–403.

[32] Qazi, Emad Ul Haq, Muhammad Hamza Faheem, and Tanveer Zia. "HDLNIDS: Hybrid Deep-Learning-Based Network Intrusion Detection System." *Applied Sciences*, vol. 13, no. 8, 2023, p. 4921.

[33] Kasongo, Sydney Mambwe. "A deep learning technique for intrusion detection system using a Recurrent Neural Networks based framework." *Computer Communications* 199 (2023): 113-125.

[34] A. Araújo, D. Rodrigues, P. Leite and J. Gonçalves, "Network Intrusion Detection System Based on Multiple Datasets: Machine Learning Approaches," 13th International Symposium on Digital Forensics and Security (ISDFS), Boston, MA, USA, 2025, pp. 1-5,

[35] P. R Buvaneswari, M. G K, H. Alasady, K. Alagarraja and M. Soni, "Anomaly-Based Intrusion Detection Systems by using Machine Learning based Stacking Ensemble Model," 2025 3rd International Conference on Integrated Circuits and Communication Systems (ICICACS), Raichur, India, 2025, pp. 1-5.

[36] Manderna, Ankit, et al. "Intrusion detection in internet of things using differential privacy: A hybrid machine learning approach." *Ad Hoc Networks* 174 (2025): 103818.

[37] Das, Tapadhir, et al. "UNR-IDD: Intrusion Detection Dataset Using Network Port Statistics." IEEE 20th Consumer Communications & Networking Conference (CCNC), IEEE, 2023.

[38] Skoudis, E., and T. Liston. Counter Hack Reloaded: A Step-by-Step Guide to Computer Attacks and Effective Defenses. Prentice Hall, 2006.

[39] Shirey, R. Internet Security Glossary, Version 2 *(RFC 4949)*. IETF, 2007.

[40] Seqrite Labs. (2025). India Cyber Threat Report 2026. Seqrite (Quick Heal Technologies Ltd.). Ret https://www.seqrite.com/india-cyber-threat-report-2026/

[41] Stallings, W. Network Security Essentials: Applications and Standards. Pearson, 2013.

[42] Ma, Huixin, et al. "Developing an Evolutionary Deep Learning Framework with Random Forest Feature Selection and Improved Flow Direction Algorithm for NOx Concentration Prediction." Engineering Applications of Artificial Intelligence, vol. 123, 2023, p. 106367.

[43] Sunil. "Intrusion Detection, Model Comparison." *Kaggle*,2023, https://www.kaggle.com/code/sunil199/intrusion-detection-model-comparison.

[44] Biswas, Saroj Kr, et al. "Performance of Ensemble Learning Techniques for Network-Based Intrusion Detection System (NIDS): A Comparative Study." *International Conference on Computing and Machine Learning*, Springer Nature Singapore, 2024.