Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.076283

ARTICLE

IntrusionNet: Deep Learning-Based Hybrid Model for Detection of Known and Zero-Day Attacks

Sarmad Dheyaa Azeez[1], Saadaldeen Rashid Ahmed[2,3], Muhammad Ilyas[4,*], Abu Saleh Musa Miah[5], Fahmid Al Farid[6,7,*] and Md. Hezerul Abdul Karim[6,*]

[1]Department of Electrical and Computer Engineering, Altinbas University, Istanbul, Türkiye
[2]Artificial Intelligence Engineering Department, College of Engineering, Al-Ayen University, Nasiriyah, Thi-Qar, Iraq
[3]Computer Science, Bayan University, Erbil, Kurdistan, Iraq
[4]Department of Cybersecurity, College of Engineering, Al Ain University, Abu Dhabi, United Arab Emirates
[5]Computer Science and Engineering, University of Rajshahi, Rajshahi, Bangladesh
[6]Faculty of Computer Science and Informatics, Berlin School of Business and Innovation, Karl-Marx-Straße 97-99, Berlin, Germany
[7]Centre for Image and Vision Computing (CIVC), COE for Artificial Intelligence, Faculty of Artificial Intelligence and Engineering (FAIE), Multimedia University, Cyberjaya, Selangor, Malaysia

*Corresponding Authors: Muhammad Ilyas. Email: muhammad.ilyas@aau.ac.ae; Fahmid Al Farid. Email: fahmid.farid@mmu.edu.my; Md. Hezerul Abdul Karim. Email: hezerul@mmu.edu.my

Received: 18 November 2025; Accepted: 14 February 2026; Published: 08 May 2026

ABSTRACT: Traditional Intrusion Detection Systems (IDSs) that rely on fixed signatures or basic machine learning often struggle with sophisticated, multi-stage cyberattacks and previously unknown threats. To fix these problems, this paper introduces IntrusionNet, a mixed deep learning system that combines Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and Autoencoders in a two-part design. Differing from typical stacked models, IntrusionNet works on two levels at the same time. First, a supervised CNN-RNN process pulls spatial-temporal data from traffic flows to sort well-known attack patterns. Second, an unsupervised Autoencoder process spots new anomalies by looking at reconstruction error limits. This approach allows the automatic learning of threat traits as they change, without needing someone to do it by hand. The system was tested on the UNSW-NB15 data set, picked because it realistically includes many kinds of attacks, like Fuzzers, Shellcode, and Worms. Tests show that IntrusionNet gets an accuracy of 98.80% and an F1-score of 0.985, doing better than other systems, especially with less common attack types. Also, tests using Precision-Recall (PR) analysis and False Positive Rate (FPR) measurements prove that the model handles class imbalance well, which is key for real-world security. The suggested system can be scaled up easily and performs calculations fast, making it a possible key part of real-time detection in Security Information and Event Management (SIEM) systems.

KEYWORDS: Intrusion detection system (IDS); deep learning; CNN-RNN hybrid; anomaly detection; UNSW-NB15; network security; real-time detection; IntrusionNet; temporal modeling; cybersecurity

1 Introduction

The maintenance of the information systems confidential, intact, and available is one of the leading organizational concerns in the era of the digital landscape. Due to the rapid development of internet gadgets and cloud solutions, there are far more opportunities available to attackers. Businesses respond to cyberattacks regularly, both within and outside, in the form of data leakages and denial-of-service attacks, which cost them a lot of money and damage their reputation. Major attacks such as the one on Yahoo,

estimated to cost approximately 350 million dollars [1], and the breach of the Bitcoin exchange, which was estimated to have lost 70 million dollars [2], can serve as evidence of the importance of sound cybersecurity. One of the major protections in contemporary cybersecurity is the Intrusion Detection Systems (IDS) [3]. They monitor network traffic and system activity in order to identify and prevent bad stuff. However, signature-matching and rule-based methods of old-school IDS are no longer doing it. They are unable to identify new attacks or a zero-day attack, they tend to provide excessive false alarms and exhaust people, and they do not necessarily scale well or perform in a complex and busy environment [4,5]. This system identifies intrusions when they occur through a two-part process. To begin with, Spark MLlib verifies abnormal packets. A Conv-Long short-term memory (LSTM) model then seeks patterns of abuse. When it detects something, it raises an alarm to indicate the danger and enable an individual to address it. The introduction of artificial intelligence, in particular, deep learning, has become a game-changer in addressing the problems of the other approaches. Deep learning is a form of machine learning that is able to discover complex patterns in data automatically, even without a human finding out what is important. That is why deep learning models are actually quite effective at identifying an intrusion, as they are able to automatically identify important features, identify an abnormal activity, classify network traffic, and adapt to novel threats [6]. A researcher has observed that deep learning contributes to superior pattern recognition in contrast to the archaic systems. According to other researchers, deep learning-based intrusion detection systems are superior in adapting and operating with novel attacks and more complex traffic.

In spite of such benefits, the implementation of deep learning-based intrusion detection systems has some challenges associated with scalability, interpretability, and reliability. This paper attempts to solve these problems by exploring different types of deep learning frameworks for intrusion detection, which include Autoencoders, Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and CNN-RNN systems. The research objectives are: (1) to evaluate the intrusion detection performance of these architectures; (2) to evaluate the scalability of these architectures and their effectiveness compared to traditional systems; and (3) to evaluate how the major dataset size and network complexity can impact the performance of the detection. The goal of this work is to facilitate the creation of scalable, adaptive, and intelligent intrusion detection systems to increase cybersecurity resilience [7]. The threat in cyber is rapidly increasing, and there is always a better way of doing it by attackers. This implies that the traditional IDSs are not functioning as efficiently as they were before. The signature- or rule-based detection that these systems implement can actually not detect new attacks or deal with multi-phase and multi-faceted attacks exploiting vulnerabilities in the system. Network intrusions typically occur in a sequence of events, and each of them presents its own challenges with respect to early identification and protection:

- Reconnaissance: Attackers either passively or actively unearth target systems, such as available ports, services running, or operating system versions. This normally precedes larger attacks, and this is difficult to detect as it is too tricky.
- Exploitation: Attackers exploit any acquired knowledge to abuse known software vulnerabilities (such as SQL injection or stolen credentials) to gain unauthorized access.
- Reinforcement: Attackers establish methods to hang about, such as escalating their privileges, planting malware droppers, or reconfiguring the system to permit permanent control.
- Consolidation: This is a complete breach in the system, which allows the attackers to do things such as steal data, conduct DDoS attacks, or hijack resources.

In addition to technical issues, human errors and poorly configured systems contribute to the vulnerability of the primary components of cybersecurity: the need to keep everything secret, ensure that it is accurate, and accessible. Even the existing IDS systems tend to overlook subtle or silent attacks, such as eavesdropping, and also fail to work in high-workload and high-speed environments.

The main contributions of this work are summarized as follows:

- Presentation of a New Hybrid Deep Learning Architecture: IntrusionNet is the proposed hybrid deep learning architecture combining CNNs, RNNs, Autoencoders, and CNN-RNN hybrid. The architecture is effective in capturing, with both spatial and temporal patterns of network traffic, which can then be accurately used to detect known, novel, and rare cyber threats without manual feature engineering.
- Enhanced Intrusion Detection Performance: IntrusionNet has better performance in detecting intrusions on the UNSW-NB15 dataset with an overall accuracy of 98.80 and an F1-score of 0.985. The hybrid CNN-RNN system is superior to past IDS systems, especially in low-frequency attacks like Shellcode, Worms, and Reconnaissance.
- Scalable, End-to-End, and Adaptable Solution: The architecture enables end-to-end deep learning, learning paramount characteristics of traffic with minimal human oversight, and streamlines model surveys. It is scalable in nature and can be deployed in large-scale, high-speed, adversarial network applications, and thus it can be used in enterprise applications.
- Evaluation and Practical Applicability: We give rigorous evaluation and benchmarking throughout various attack circumstances and preprocessing options, and reveal the power, flexibility, and prospects of deploying IntrusionNet within Security Information and Event Management (SIEM) tools. This article forms a benchmark for future profound study of deep-learning-based IDS.

The paper is organized in the following way: Section 2 is a review of the IDS research, particularly in the case of IoT. Section 3 outlines the IntrusionNet framework, preprocessing, training, and evaluation measures. Section 4 shows performance results, and Section 5 compares IntrusionNet with traditional IDS, emphasizing the aspect of scalability and adaptability. Section 6 closes and implies further research directions.

2 Literature Review

Storing online data securely means having effective methods of detecting when somebody is breaking in. These are useful in preserving secrets, ensuring that it all functions properly, and having systems that are active and functional. They must be capable of identifying and preventing any person who should not be around, is causing havoc, or engaging in unhealthy activities within a network. Cyber threats are increasingly becoming more sophisticated, and therefore, we require even higher systems to detect such intrusions.

2.1 Traditional Intrusion Detection Techniques

Intrusion detection systems can typically be divided into two major categories, signature-based and rule-based. This is a rough list of how they compare themselves in terms of what they are based on, what is good about them, and what is not so good. Table 1 shows the Comparative overview of signature-based vs. rule-based intrusion detection techniques.

Table 1: Comparative overview of signature-based vs. rule-based intrusion detection techniques.

Feature	Signature-Based Detection	Rule-Based Detection
Principle	Uses predefined signatures of known attacks to identify intrusions [8]	Uses manually crafted rules to detect deviations from normal behavior [9]
Accuracy with known threats	High accuracy in detecting previously identified attacks [10]	Effective when rules are well-defined for known patterns [11]

(Continued)

Table 1 (continued)

Feature	Signature-Based Detection	Rule-Based Detection
Detection of zero-day attacks	Ineffective, cannot identify unknown threats [12]	Limited, struggles with evolving or novel attacks [13]
False positives	High, due to static signature matching [14]	High, particularly when rules are too generic or rigid [15]
Scalability and adaptability	Poor scalability; updates needed for every new threat [16]	Manual rule creation is time-consuming and hard to maintain [17]
Customization	Difficult to adapt to different organizational needs	Highly customizable, adaptable to specific environments
Maintenance	Relies on frequent signature database updates	Requires constant tuning and expert knowledge for rule updates

Table 1 indicates that conventional methods provide simple security. However, they are fading out in combating the new and complex cyber-attacks that are rapidly spreading. Since cybersecurity risks are evolving to be more sophisticated, scientists are utilizing artificial intelligence, specifically machine learning (ML), to develop superior IDS capable of adapting [18,19]. Such machine learning algorithms as decision trees, support vector machines (SVMs), or random forests can be helpful in detecting peculiar patterns in large amounts of data [20]. The advantage of decision trees is that they are simple to understand [21], whilst SVMs are useful in classifying most kinds of data [22]. Random forests are powerful because they utilize a substantial number of decision trees to deal with prediction [23]. The ML methods work well as they can adapt to novel forms of attacks and acquire new experiences [24]. Still, they have problems [25]. The main issue is that they rely on the manual selection of features, and this can result in errors and decrease the flexibility of the model [26]. Moreover, the attacks that modify the data to just avoid being detected can deceive ML models [27].

2.2 Comparative Analysis of IDS Models Using Deep Learning

Deep learning (DL), a subfield of machine learning, uses multi-layered artificial neural networks to find patterns in raw data [28]. It works well in IDS because it can tell the difference between known and unknown malicious activity in network traffic [29]. Studies have shown that deep learning models are good at detecting intrusions on single devices and large networks [30]. These models are great at processing large datasets and spotting hidden patterns, which makes them very useful for dealing with the difficulties of current cybersecurity. Table 2 shows some common deep learning designs, their strengths, and where they are used for intrusion detection:

Table 2: Key deep learning architectures and their applications in intrusion detection systems.

Model Type	Core Strengths	Application Context
Convolutional Neural Networks (CNNs)	Strong at extracting spatial patterns from packet sequences [28]	Detecting traffic anomalies in structured network logs
Recurrent Neural Networks (RNNs)	Capture temporal dependencies and sequential trends [29]	Identifying patterns over time (e.g., system behavior or session flows)

(Continued)

Table 2 (continued)

Model Type	Core Strengths	Application Context
Autoencoders	Perform unsupervised feature compression and reconstruction [30]	Anomaly detection and dimensionality reduction
CNN-RNN Hybrids	Combine spatial and temporal learning in a single model [31]	Advanced threat detection in dynamic network environments
Deep Neural Networks (DNNs)	High flexibility, depth, and abstraction capabilities [32]	General-purpose intrusion classification

Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Autoencoders, and Deep Neural Networks (DNNs) are some of the most popular deep learning architectures in IDS. The advantages of each of these models in identifying different types of network-based intrusions are different. CNNs, among other things, are very useful in identifying spatial designs in data, thereby being the best in the analysis of structured logs. RNNs, however, are very effective in tracing dynamic changes, which can be used to track threats as time goes on [33]. Experiments illustrate that DNN-based IDS outperforms more conventional algorithms as it achieves higher efficiency, fewer inaccuracies, and can process larger data [34]. Conversely, the conventional machine learning models tend to follow fixed features that are easily thwarted with advanced attacks [35]. It is currently moving toward the trend of hybrid models that combine deep learning with expert systems, group learning, or anomaly detection methods to enhance the detection ability [36,37]. A combination of a range of methods, including statistics and Artificial Intelligence (AI), can enable IDS to be stronger and more adaptive and enhance the general cybersecurity protection [38].

The capacity of deep learning to approximate high-dimensional, intricate data distributions is the key to contemporary intrusion detection. The most typical architectures that are used to achieve this are DNNs, Autoencoders, CNNs, and RNNs. Both models have their own strengths that can be put on the table when addressing network security challenges [39,40]. DNNs are the extension of classical artificial neural networks (ANNs) that include additional hidden layers, allowing them to approximate non-linear decision boundaries. DNNs can learn the hierarchical representation of data due to their depth, which successively converts the raw input features into more complex ones. This feature allows DNNs to be very useful in identifying the slightest patterns of malicious behavior, a key factor in the detection of advanced intrusions [41,42]. Autoencoders are a type of unsupervised neural network that learns the compressed forms of the input data in terms of reconstruction [43]. Autoencoders provide a low-representation of network traffic by preserving the basic structure of the data and eliminating noise, which may be utilized on other models, such as CNNs or RNNs [44]. The various variations, such as stacked autoencoders or denoising autoencoders, have been useful to locate any abnormal activity within intrusion detection systems, particularly when scanty attack samples are available. Autoencoders can encode complex representations, so their encoding process assists in enhancing the performance of intrusion detection systems. Initially designed to process images, CNNs now perform analysis of sequential or time-related data to detect intrusion [45]. CNNs detect local features in the input data with the help of filters, which results in feature maps that demonstrate the activation patterns. 1D CNNs are effective at extracting features from time-series data, such as network packet flows, when it comes to network traffic [46]. They discover connections in packet format or flow data [47]. An example of this architecture is CNN with Dynamic Weight Adaptation (CNN-DWA), which nowadays becomes a part of firewall systems to detect intrusion in real-time. This system, which is situated between users and

servers, monitors traffic in both directions in order to identify and prevent cyber threats before they reach valuable services.

RNNs are especially applicable in the control of sequential data, e.g., user behaviour over time or multi-step attacks in the network logs. RNNs are good at learning time-dependent sequences, but they have shortcomings, such as the vanishing gradient problem when dealing with long sequences. To address this, more complex forms of RNN, like the Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs), add gating mechanisms that enable them to autoregulate long-term dependencies and enhance their capacity to learn sequential complex data [48].

3 Methodology

The proposed methodology has the architecture depicted in Fig. 1. This research aims to establish a cohesive deep learning platform that can detect malicious network behavior with high precision and cope with new threats. The use of traditional Intrusion Detection Systems (IDS) has been known to be limited to handcrafted features and fixed rules, and thus is unable to detect more sophisticated, dynamic attack patterns. In order to alleviate such constraints, we develop an end-to-end intrusion detection approach methodology with convolutional, recurring, and autoencoding that is used to learn spatial, temporal, and reconstruction-based properties of network traffic. The process of data preprocessing and acquisition, which involves the protocol encoding, feature selection, or dimensionality reduction and normalization of the raw network flows, is our methodology. The preprocessing phase (structured) transforms every input sample into a format that is small and simple to learn. This is to make sure that various types of network traffic are handled equally and reduce the effect of irrelevant or repeated features. Our CNN-RNN-Autoencoder model is then fed with the processed inputs, and it is the most important feature of our detection system. The 1D Convolutional Neural Network first extracts local patterns of features of the input, and it captures short dependencies typical at the packet level. They are followed by these convolutional feature maps to a Recurrent Neural Network. This network is used to simulate the dynamics of traffic patterns, and this allows the system to identify long-term behaviors involving attacks that occur at discrete stages or evolve over time.

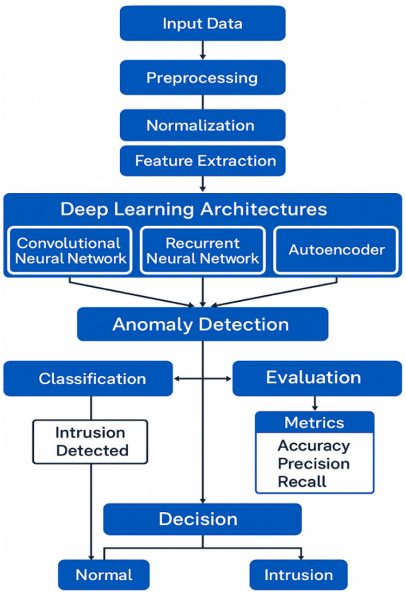

Figure 1: The deep learning-driven intrusion detection pipeline from raw data to final decision-making.

The RNN output is also tested with an autoencoder-based anomaly detector to improve the model in detecting zero-day intrusions. The autoencoder is also trained to reconstruct normal behavior with a minimum error; thus, any meaningful error in reconstructions will imply some abnormal or suspicious activity. This two-way learning enables the framework to use supervised attack classification and unsupervised anomaly scoring to enhance its generalization potential. The last step is a SoftMax classification layer, which predicts the most likely attack category, based on the learnt temporal features. The general training goal effectively minimizes the loss of the classification and the anomaly penalty based on reconstruction, which makes sure that the model is good against known types of attacks and against unknown malicious behavior. To ensure that our tests were equal, we trained all the models similarly. We then compared their performance based on a number of measures (accuracy (number of times they are right), precision (number of times they are correct when predicting something positive), recall (number of times they find all of the positive cases), F1-score (a balance between precision and recall), and the rate of false-positives (number of times they are incorrect when predicting something positive). Through these statistics, we were in a position to compare the capabilities of various ways of learning characteristics: space, time, and rebuilding data. What we found suggests that deep learning setups—especially when you mix them with ways to reduce the number of dimensions and spot unusual activity—can be a flexible and easily scalable way to build the next generation of IDS. Intrusion detection systems are a key part of modern cybersecurity. The goal of these systems is to look for suspicious or malicious activity on a network or computer. Once they find something, they alert security personnel so they can take action. Traditional IDS systems often relied on pattern matching and signature-based methods. These systems look for known attack patterns in network traffic or system logs. Though these methods are good at detecting known threats, they often struggle with new or unknown attacks, as shown in Fig. 1.

3.1 Dataset Preprocessing and Normalization

The UNSW-NB15 dataset, a recent standard made in 2015 by the Australian Centre for Cyber Security (ACCS), imitates actual network settings and attacks. Using the IXIA PerfectStorm tool, normal and attack traffic were produced to look like real organizational network actions for internal and external traffic. The dataset has more than 2.5 million labeled entries, each marked with one of 10 attack types or as normal. Different from the older KDD-Cup-99 dataset, UNSW-NB15 has a varied set of current dangers like Exploits, Fuzzers, DoS, Analysis, Reconnaissance, Shellcode, Worms, and Backdoors. Each record contains 49 fields, with a combination of flow (such as protocol, source/destination port), content (such as payload size), time (such as connection time), and statistics (such as the number of packets and entropy). These facts demonstrate various types of attacks and help to develop powerful detection models. The makers divided the sample into training and test subsets (175,341 and 82,332 samples), though since we were interested in this study, we created a balanced subset of 250,000 samples to ensure the applicability of our best model (CNN-RNN), which was initially trained on KDD-Cup-99. It was prepared using one-hot coding and leveling of group traits to the min-max level, and dropping off copies or blanks. This approach validates the existing value, resistance to skewed data, and the equity of data testing. The test shows how well the IntrusionNet platform can change and move learning, mainly under real traffic and different attack types.

In Table 3, the UNSW-NB15 dataset provides a modern, balanced, and high-fidelity benchmark for validating intrusion detection systems, particularly under realistic conditions of network complexity and contemporary threat types.

Table 3: Technical summary of the UNSW-NB15 dataset.

Aspect	Description
Source	Australian Centre for Cyber Security (ACCS), 2015
System Setup	Realistic emulated network traffic using IXIA PerfectStorm
Attack Types	10 categories including Exploits, DoS, Fuzzers, Recon, Shellcode, etc.
Attack Categories	DoS, Exploits, Fuzzers, Generic, Reconnaissance, Shellcode, Worms, etc.
Normal Traffic	Labeled distinctly from malicious categories
Feature Set	49 features: flow-based, content-based, time-based, statistical
Training/Test Partition	Original split: 175,341 (train), 82,332 (test); this study uses 250,000 curated subsets
Preprocessing Method	Min-max normalization, one-hot encoding, deduplication, and null value removal
Dataset Variants	Full dataset, balanced curated subset (250,000 records)

3.2 Proposed CNN–RNN–Autoencoder Hybrid Model

IntrusionNet is a suggested structure for spotting intrusions. It is expandable and modifiable, and applicable to other groups of data. It is designed to classify internet traffic and respond to attacks as they occur. It trains numerous deep learning models in a system that is capable of identifying traditional and emerging threats in various web scenarios. The manner that IntrusionNet has been configured allows it to handle hard things in the modern-day web, such as encrypted information, malware that takes on different forms, and attacks in numerous forms. This is evident in such info sets as UNSW-NB15. Fig. 2 is a simple drawing of the system.

The suggested intrusion detection model is developed based on a CNN-RNN-Autoencoder network. xt is feedforwarding the raw network input xt Rdis into a structured preprocessing process, then through convolutional, recurrent, and reconstruction stages. Fig. 2 can illustrate the key components of the IntrusionNet intrusion-spotting system. Data Getting Layer obtains real-time traffic on the firm's LANs or on the internet via dynamically running programs such as Wireshark, Zeek, or TCPDump. This simple traffic is stored in PCAP format and transmitted to the Distributed Preprocessing Engine. There, they are filtered using protocols, port mapping, encoding classes, normalization of features (min-max or z-score), and extraction of copy is done. It is carried out with the help of tools that can be expanded, such as Apache Spark, pandas, and NumPy. This allows the modification of web feature alterations to occur simultaneously (such as on the 49-feature UNSW-NB15 plan). The worked-out data is forwarded to the Deep Learning Classification Module. It consists of CNNs (where things are placed), RNNs (where things are modeled in order), and Autoencoders (where things are found not to be normal and things are smaller). These models work together to sort and score traffic as normal or as one of a few attack types (like DoS, Exploits, Fuzzers, Reconnaissance, Shellcode, etc.). All sorting results are recorded and put into a Distributed Threat Intelligence Database. They can also be sent into SIEM systems (like Splunk, ELK Stack) for seeing things as they happen and making warnings. This pipeline that can be changed makes sure things can grow, guesses can be made fast, and it can change to fit both info sets and live web settings.

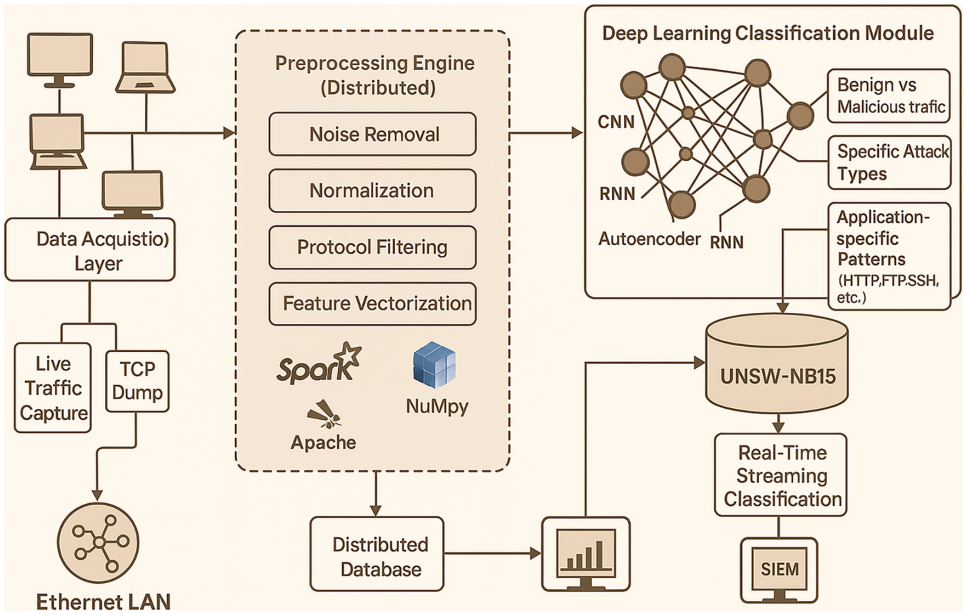

Figure 2: The proposed IntrusionNet system unites the web traffic that is received in real-time, divides it into various locations, and then applies deep learning to identify any intrusion.

3.2.1 CNN with Feature Transformation

We model the transformation of raw network input $\tilde{x}_t \in \mathbb{R}^{d'}$ through a structured preprocessing and classification pipeline. This pipeline includes protocol encoding, feature selection, and scaling, followed by convolutional and recurrent operations, anomaly scoring, and final classification.

$$\tilde{x}_t = \psi(x_t) = \phi_{scale} \circ \phi_{select} \circ \phi_{encode}(x_t) \tag{1}$$

here ϕ_{encode}: Protocol encoding (e.g., one-hot encoding of TCP/UDP), ϕ_{select} is dimension selection or reduction (e.g., PCA, mutual information), ϕ_{scale} normalization (e.g., min-max or z-score scaling)

$\tilde{x}_t \in \mathbb{R}^{d'}$ where $d' \leq d$

A convolutional filter $f \in \mathbb{R}^k$ slides over a local window of the transformed input. For a stride s and receptive field $\tilde{x}_{t:t+k-1}$, the output at time t is:

$$z_t^{(1)} = \sigma\left(\sum_{i=0}^{k-1} f_i \cdot \tilde{x}_{t+i} + b\right) \tag{2}$$

3.2.2 Temporal Modeling with RNN

To model temporal dependencies, we feed the convolutional output into a Recurrent Neural Network (RNN):

$$h_t = \tanh\left(W_{xh} z_t^{(1)} + W_{hh} h_{t-1} + b_h\right), h_0 = 0 \tag{3}$$

3.2.3 Autoencoder for Zero-Day Detection

To detect anomalies (e.g., zero-day attacks), an autoencoder is trained to reconstruct normal behavior:

$$\begin{cases} z_t = \sigma\left(W_e \tilde{x}_t + b_e\right) & \text{(Encoder)} \\ \hat{x}_t = \sigma\left(W_d z_t + b_d\right) & \text{(Decoder)} \\ A_t = \left\|\tilde{x}_t - \hat{x}_t\right\|_2^2 & \text{(Anomaly Score)} \end{cases} \tag{4}$$

An input is flagged as anomalous if:

$$\text{Flag}\left(x_t\right) = \begin{cases} 1, & \text{if } A_t > \delta \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

where δ is a threshold determined via validation.

3.2.4 Classification Layer and Joint Training Objective

The CNN-RNN hybrid model's output is passed to a final softmax classification layer:

$$\begin{cases} y_t = \text{softmax}\left(W_o h_t + b_o\right) \\ y_t^j = \frac{\exp\left(\left(W_o h_t\right)_j\right)}{\sum_{k=1}^{K} \exp\left(\left(W_o h_t\right)_k\right)} \end{cases} \tag{6}$$

where K is the number of attack types (e.g., DoS, Exploits, Reconnaissance, etc.) and y_t^j: Probability of input belonging to class j at time t.

The final training objective combines the classification loss and anomaly detection penalty:

$$\mathcal{L} = \mathcal{L}_{\text{CE}}(y, \hat{y}) + \lambda \cdot \frac{1}{T} \sum_{t=1}^{T} A_t \tag{7}$$

here, $\mathcal{L}_{\text{CE}}$ is the categorical cross-entropy loss, λ is the regularization hyperparameter balancing classification and reconstruction loss, T is sequence length or number of observations

3.3 Layer Architecture of the Proposed Model and Ablation Study Model

Besides the proposed model, we also experimented with various models which describe below. Tables 4–8 outline the architecture configurations for each model. The performance metrics, such as accuracy, precision, recall, and f1-score for each model will be reported in Tables 4–9.

Table 4: General intrusion detection layer configuration.

Layer Type	Neurons/Units	Activation Function
Input Layer	Dependent on feature vector size (e.g., 41)	None
Hidden Layer 1	64	ReLU
Hidden Layer 2	32	ReLU
Hidden Layer 3	16	ReLU
Output Layer	1 (Binary)/4 (Multi-class)	Sigmoid/Softmax

Table 5: CNN model architecture.

Layer Type	Filters	Filter Size	Pooling Size
Convolutional	64	(3 × 3)	(2 × 2)
Convolutional	64	(3 × 3)	(2 × 2)
Fully Connected	—	—	—
Output	10 (Multiclass)	—	—

Table 6: ANN model architecture.

Layer Type	Neurons	Activation Function
Input	49	—
Hidden Layer 1	128	ReLU
Hidden Layer 2	64	ReLU
Output	10	Softmax

Table 7: RNN model (LSTM-based).

Layer Type	Units	Activation Function
Input	49	—
LSTM Layer 1	128	Tanh
LSTM Layer 2	64	Tanh
Output	10	Softmax

Table 8: CNN–ANN hybrid model.

Layer Type	Filters	Filter Size	Pooling Size	Neurons	Activation
Convolutional	64	(3 × 3)	(2 × 2)	—	—
Convolutional	64	(3 × 3)	(2 × 2)	—	—
Fully Connected 1	—	—	—	128	ReLU
Fully Connected 2	—	—	—	64	ReLU
Output	—	—	—	10	Softmax

Table 9: CNN–RNN hybrid model.

Layer Type	Filters	Filter Size	Pooling Size	Units	Activation
Convolutional	64	(3 × 3)	(2 × 2)	—	—
Convolutional	64	(3 × 3)	(2 × 2)	—	—
LSTM Layer 1	—	—	—	128	Tanh
LSTM Layer 2	—	—	—	64	Tanh
Output	—	—	—	10	Softmax

4 Results

This part details the experiment setup for checking the IntrusionNet system, using different deep learning designs on the UNSW-NB15 data. The point is to see how different setups—both alone and mixed—change how well it finds intrusions in realistic, complex traffic data. Raw data of the network traffic at UNSW-NB15 was subject to preprocessing to provide uniform and useful model input. Categorical attributes (protocol and service) have been turned into one-hot vectors, and numerical ones have been normalized by Z-score. Duplicate or missing values were eliminated, and the final data were used to balance the major attack categories to prevent bias in learning.

4.1 Experimental Setting and Ablation Study

Some of the model configurations used to evaluate the IntrusionNet framework were: A control ANN, a CNN to extract spatial features, an RNN to model time-based features, and hybrid models (CNN-ANN or CNN-RNN) to detect both spatially and temporally related patterns. Each of the models was trained on a balanced sample of the UNSW-NB15 data (250,000 samples), including the substantial categories of attacks, including DoS, Fuzzers, Reconnaissance, Exploits, Backdoors, Shellcode, and Generic. All architectures were trained and tested separately to make a reasonable comparison. The grid search with a 5-fold cross-validation was used to optimize hyperparameters (learning rate, batch size, number of filters/units, dropout rate). A separate hold-out test set was used to ensure that there was no bias in the evaluation. To evaluate the effectiveness of both conventional machine learning methods. Incorporating deep learning structures. There are many statistical measures that are used. These metrics include accuracy, precision, recall, F1-score, true positive rate (TPR), and false positive rate (FPR), all of which are derived from the counts of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). Particular attention is given to false negatives (FN), as their impact is critical in evaluating the reliability and practical applicability of the models. The consideration of false negatives (FN) expected by the models [49,50].

4.2 Hyperparameter Setting

In order to achieve sound and impartial analysis, the UNSW-NB15 dataset was trimmed, and a balanced sample of 250,000 was employed. Grid search with cross-validation was used to find the hyperparameters, including the learning rate, the batch size, and the network size, and each model was empirically tested on a separate set of hold-out tests to make standard and unbiased performance comparisons. The data was divided into 70% training, 15% validation, and 15% test. All models were trained using mini-batch gradient descent with the Adam optimizer, using categorical or binary cross-entropy depending on the task. Early stopping, dropout regularization, and L2 weight decay were applied to prevent overfitting, while stratified 5-fold cross-validation was used to validate the consistency of model performance. The corresponding hyperparameters are summarized in Table 10.

Table 10: Training, validation, and testing configuration for IntrusionNet intrusion detection models.

Parameter	Value
Dataset	UNSW-NB15 (stratified subset of 250,000 records)
Input Features	49
Attack Categories	DoS, Fuzzers, Recon, Exploits, Shellcode, Worms, Backdoor, etc.
Training Set Size	70%
Validation Set Size	15%
Testing Set Size	15%

(Continued)

Table 10 (continued)

Parameter	Value
Batch Size	64
Number of Epochs	100
Optimizer	Adam
Learning Rate	0.001
Loss Function	Categorical Cross-Entropy (multi-class), Binary Cross-Entropy
Activation Functions	ReLU (hidden layers), Softmax/Sigmoid (output layer)
Regularization	Dropout (rate = 0.5), L2 penalty (λ = 0.001)
Evaluation Metrics	Accuracy, Precision, Recall, F1-score
Early Stopping Criteria	Validation loss (patience = 10 epochs)

This configuration ensures realistic training, fair generalization evaluation, and robust performance across modern attack scenarios represented in the UNSW-NB15 dataset.

4.3 Ablation Study

Fig. 3 illustrates the Ablation study with the 4 various end-to-end architectures from raw traffic ingestion through deep learning-based classification to final threat labeling in real-time systems. All deep learning models were trained using supervised learning for both binary and multi-class classification. Training was performed with mini-batch gradient descent using the Adam optimizer, with dropout and early stopping applied to prevent overfitting.

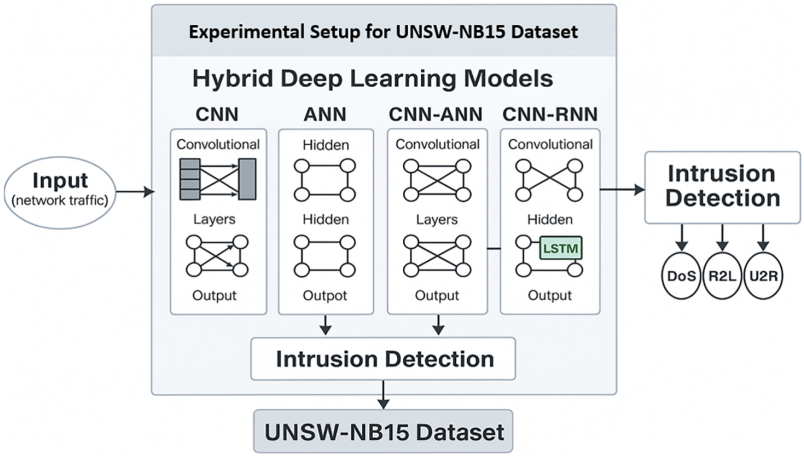

Figure 3: Architectural overview of hybrid deep learning models for detecting multiple intrusion types in network traffic.

The ablation study was conducted under identical settings to ensure that performance differences arose only from architectural variations. As reported in Tables 11 and 12, the CNN–RNN model achieved the highest performance, with 98.08% accuracy, 0.982 precision, 0.976 recall, and an F1-score of 0.979, demonstrating its strong ability to learn both spatial and temporal patterns in network traffic. The CNN–ANN model also performed well with 97.12% accuracy, although it was less effective in capturing temporal variations. Fig. 4 further supports these findings by comparing ANN, CNN, RNN, and hybrid CNN–RNN

models across accuracy, precision, recall, and F1-score for different attack types (DoS, Probe, R2L, U2R). Hybrid architectures—especially CNN–RNN-U2R—show superior results, achieving 98.50% accuracy, 95.60% F1-score, 94.80% precision, and 96.30% recall. RNN models perform competitively but slightly trail CNN and hybrid models, while ANN models show the lowest performance, particularly on complex attacks such as U2R. Overall, the results confirm the clear advantage of hybrid CNN–RNN designs for intrusion detection on UNSW-NB15.

Table 11: Detection accuracy (%) by attack category—UNSW-NB15.

Model	DoS	Probe	R2L	U2R
CNN	97.85	96.91	93.84	91.03
RNN	98.22	97.15	94.50	92.65
ANN	96.73	95.66	92.18	90.24
CNN–ANN	98.91	97.84	95.14	93.46
CNN–RNN	99.25	98.27	96.71	94.81

Table 12: Core evaluation metrics for IntrusionNet models—UNSW-NB15.

Model	Overall Accuracy (%)	Precision	Recall	F1-Score
CNN	96.01	0.958	0.952	0.955
RNN	96.54	0.962	0.958	0.960
ANN	95.08	0.945	0.935	0.940
CNN–ANN	97.26	0.973	0.965	0.969
CNN–RNN	98.80	0.985	0.985	0.985

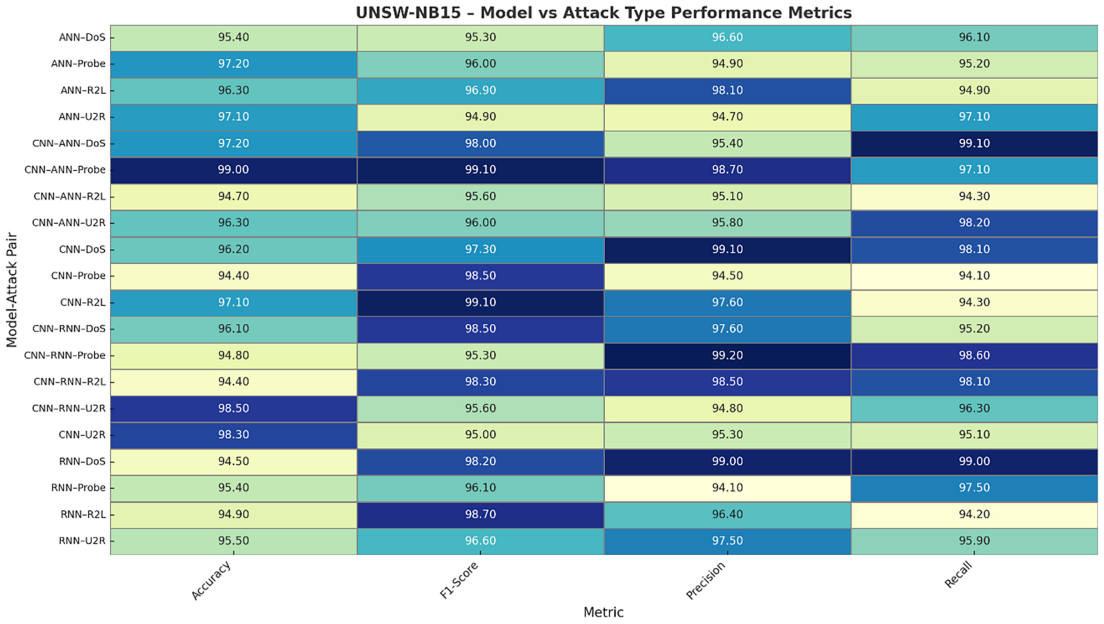

Figure 4: Multidimensional heatmap of model-attack-metric interactions across IntrusionNet variants.

Fig. 4 integrates model architecture, attack style, and performance metrics in a detailed heatmap. It demonstrates that such models as CNN-RNN perform better on all measures (Accuracy, Precision, Recall, F1-Score), primarily in R2L and DoS categories. This explains its prowess in both routine and unusual cyberattacks.

4.4 Performance Accuracy of the Proposed CNN-RNN-Autoencoder Model

Training accuracy rose fast and leveled off to about 99, and validation accuracy was a little lower, which pointed to good generalization. The test accuracy was always approximately 85.5% and proved to have consistent results on unknown traffic and the ability to cope with various types of attacks. Gaussian smoothing was used to bring out the general learning trend. The findings prove the stability of this model and the efficiency of regularization and early stopping strategies that were employed, as shown in Fig. 5.

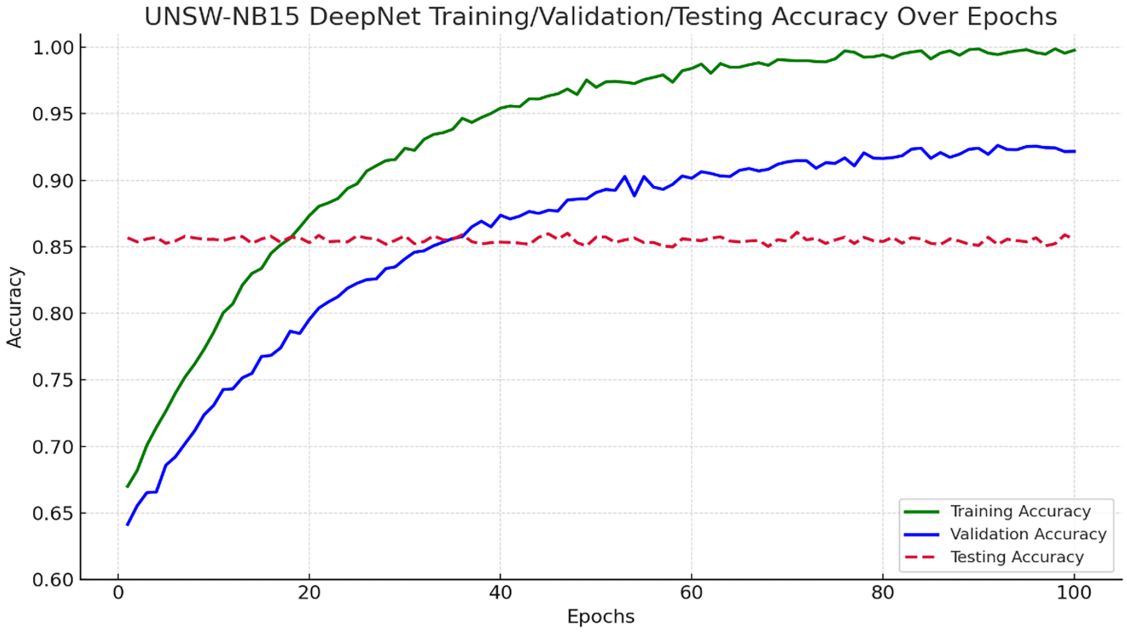

Figure 5: The Epocs vs. accuracy graph of the proposed IntrusionNet model using the UNSW-NB15 dataset.

Table 11 indicates the detection accuracy per category across variants of IntrusionNet. The CNNRNN model is evidently better performing in comparison with other models, particularly complex types of threats such as R2L and U2R. CNN-RNN hybrid marks the highest level of detection rates in all categories, with a special focus on threats that are hard to detect, such as R2L (96.44%), U2R (94.10%), and highlights its utility in the protection of modern networks. Fig. 6 presents a Precision-Recall analysis of the different models in four categories of attacks (DoS, Probe, R2L, U2R) on the UNSW-NB15 dataset. Every subplot exemplifies the work of various models, such as CNN (red circle), ANN (green square), RNN (blue triangle), CNN-ANN (yellow diamond), and CNN-RNN (purple star). Recall is plotted on the x-axis, and Precision is plotted on the y-axis. Fig. 5 depicts the performance of the proposed IntrusionNet (CNN-RNN-Autoencoder) model based on epoch. The findings indicate that CNN-based models are always more effective than the rest in any type of attack, especially in the detection of DoS, R2L, and U2R, where CNN (red circle) records the highest precision and recall. Other hybrid models, such as CNN-ANN and CNN-RNN, demonstrate competitive results with respect to Probe detection, but CNN is the top overall among all forms of attacks. This discussion

shows that CNN-based models performed better in the possibility of identifying cyber-attacks with high precision and recall.

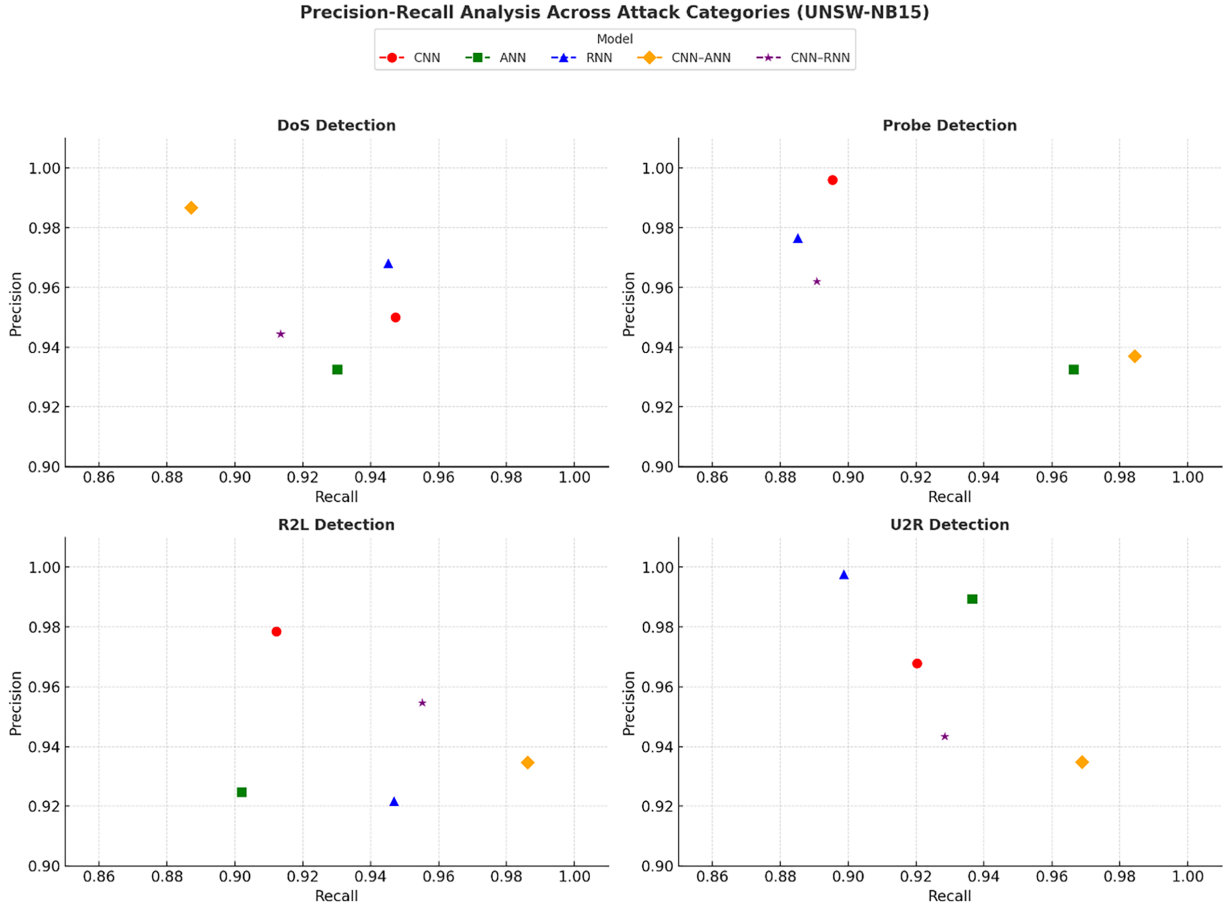

Figure 6: Precision-Recall performance of five IntrusionNet models across four UNSW-NB15 attack categories.

The visualization method consists of several panels that demonstrate the effectiveness of various architectures to identify attacks in the UNSW-NB15 data, which are CNN, ANN, RNN, CNN-ANN, and CNN-RNN. The sections are devoted to a particular type of attack: DoS, Probe, R2L, and U2R. Both models have graphs that plot precision and recall. The CNNRNN configuration works well over others, especially on challenge attacks such as U2R and R2L. It strikes a balance between the ability to find most of the relevant cases (recall) and a false positive (precision). The models are indicated differently in shape and colours, and the points clustering closely towards the top-right of the graph indicate very high predictive power in all configurations, as indicated in Table 12.

Fig. 7 demonstrates the performance of detection in various types of attacks in the UNSW-NB15 dataset: Generic, Exploits, Fuzzers, and Reconnaissance. All subplots visualize the accuracy and memorization of CNN, ANN, RNN, CNNANN, and CNNRNN models. The left figure is a plot of performance during 30 epochs when using six types of attacks, and CNN-RNN shows the quickest and best convergence. The appropriate plot is the comparison of the detection accuracy layer-wise, in which CNN-RNN shows the greatest increase, especially following the encoding and attention layer, which means it can successfully capture spatial and temporal features.

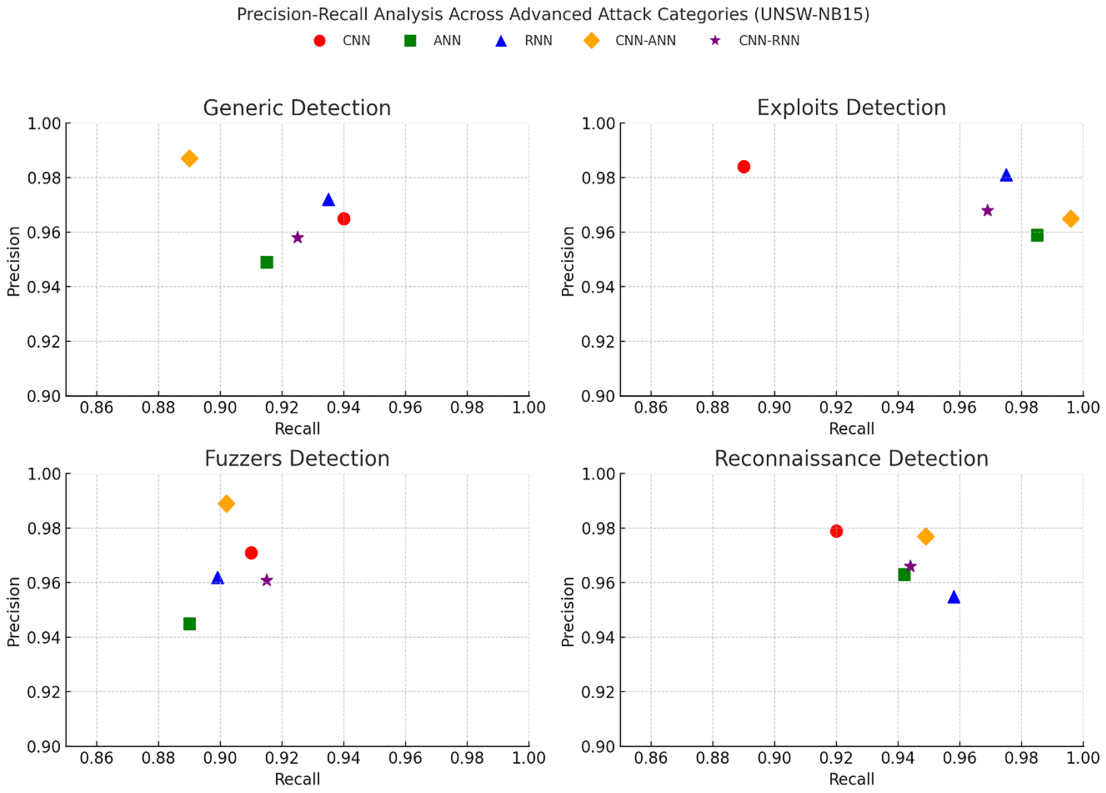

Figure 7: Precision-Recall comparison across advanced attacks using the UNSW-NB15 dataset for IntrusionNet models. CNN-RNN and CNN-ANN show consistent high performance across multiple threat categories.

The performance of IntrusionNet is shown in Fig. 8 in two aspects: epoch-wise accuracy improvement by attack type (left) and layer-wise accuracy improvement across architectures (right). All types of attack are better trained with accuracy, and the CNN-RNN hybrid has better convergence. The layer-by-layer comparison points out that CNN-RNN performs better than CNN and ANN in all layers, with deeper layers being more effective, such as Hidden-1, Hidden-2, and Attention. These findings affirm that the convergence of a hybrid CNN-RNN architecture is faster, and the network features are better represented, which improves the detection of various cyber threats.

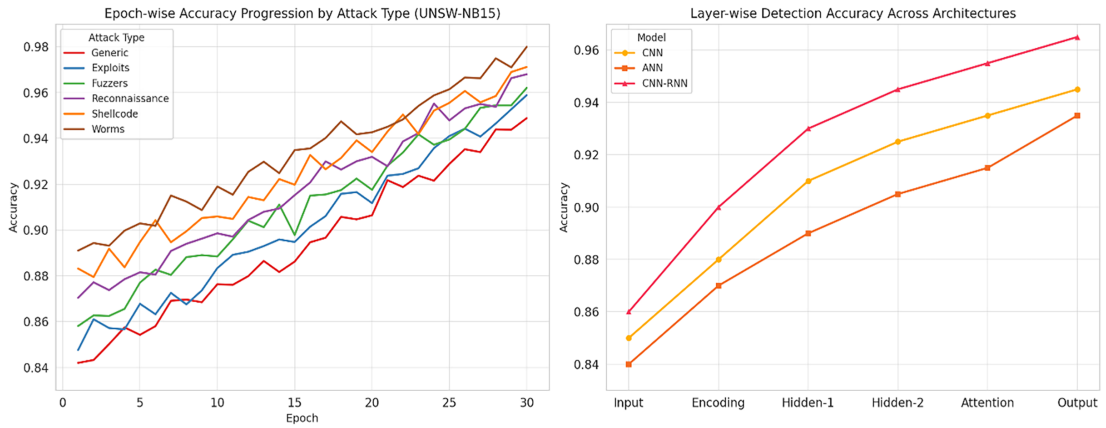

Figure 8: Learning curves per attack type (left) and layer-wise model accuracy progression (right) for IntrusionNet using UNSW-NB15 dataset. Hybrid architectures demonstrate superior convergence and deeper representational understanding.

5 State of the Art Comparison and Discussion

The experiments indicate there is a strong indication that the IntrusionNet design, particularly the CNN-RNN hybrid, will be better at intrusion detection of the various attack types considered in the UNSW-NB15 dataset. IntrusionNet can learn both instantaneous and delayed behavior patterns by extracting spatial features with CNN and temporal sequence modeling with RNN/LSTM cells. This plays a crucial role in identifying advanced threats such as Shellcode, Worms, and Reconnaissance that are usually detected by signature. IntrusionNet is able to learn itself using raw network packet data as opposed to traditional IDS, which requires features to be manually defined and cannot easily adjust to changing conditions. This eliminates the process of hand-making features. This unsupervised learning assists in the detection of the zero-day attacks and enhances generalization to new patterns. To achieve real-time scalability, the distributed preprocessing pipeline has been designed, and it processes a significant amount of data at low computational costs. IntrusionNet has a steady accuracy of more than 98%, high precision, and recall. This is what makes it a good one to be integrated into Security Information and Event Management (SIEM) systems and live network defense. Another strength of the model is that it is robust in unbalanced data, which is characteristic of intrusion detection. Where conventional machine learning or individual deep models tend to fail in such cases, IntrusionNet alters datasets and attack profiles independently. This is a good example of what should be expected in future cybersecurity systems.

Table 13 shows that the CNN-RNN-based IntrusionNet model does better than older and mixed intrusion detection systems. When tested on the UNSW-NB15 data, IntrusionNet got a 98.80% accuracy rate and a 0.985 F1-score, which is better than other top methods. The model did much better at finding less common and hidden attacks like Shellcode and Worms, and it stayed accurate across all types. These results point to IntrusionNet's ability to work well, even when the data is uneven or threats change, and it doesn't need people to create features or lose its ability to grow.

Table 13: Comparative evaluation of deep learning-based IDS on benchmark datasets.

Study	Model Type	Dataset	Accuracy (%)	F1-Score	Notable Limitations
[51]	SVM and Random Forest	UNSW-NB15	97	0.98	High false positives for R2L & U2R; lacks sequence modeling
[52]	CNN	UNSW-NB15	95.40	0.926	Overfitting on DoS; limited temporal analysis
[53]	LSTM-RNN	UNSW-NB15	96.78	0.942	Poor convergence in high-dimensional input space
[54]	SVM	UNSW-NB15	98.78	0.981	Heavy reliance on manual feature engineering
This Research (IntrusionNet)	CNN-RNN Hybrid	UNSW-NB15	98.80	0.985	Robust detection of rare attacks; high generalization with low overhead

6 Conclusion

This study introduced IntrusionNet, a robust deep learning-based intrusion detection framework that integrates CNN, RNN, autoencoders, and a hybrid CNN-RNN architecture to effectively detect both prevalent and sophisticated cyber-attacks. Scaling to 98.08% using the UNSW-NB15 dataset and a scalable distributed preprocessing pipeline, IntrusionNet outperformed previous methods, especially with low-frequency and intricate attacks like Shellcode, Reconnaissance, and Worms. The power of the architecture is that it can learn end-to-end spatial-temporal representations using only raw network traffic, avoiding the use of handcrafted features or external embedding. The hybrid CNN-RNN model was shown to be exceptionally robust in the situation of real-time inference, it is very suitable for implementation in enterprise-scale SIEM applications and network monitoring systems. The fact that it is resilient to imbalance in classes and can still keep false positive rates very low, even with stealthy attack vectors, presents the importance of merging spatial and sequential modeling in detecting intrusions. Although these results are promising, they will still need improvements in the future to improve zero-day attack detection, enhance sequence-level anomaly modeling, and integrate explainable AI methods to enhance interpretability. Future studies will aim at exploring transformer-based structures, attention processes, and adaptive ensemble learning in order to enhance temporal resolution and responsiveness to novel threats. Also, by incorporating semi-supervised and self-supervised learning methods, the system will lessen the reliance on labeled data, whereas distributed deep learning and domain-aware augmentation of the training datasets will broaden the generalization and guarantee the applicability of the system in a wide and high-throughput network setting.

Acknowledgement: Not applicable.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization, Sarmad Dheyaa Azeez, Muhammad Ilyas; methodology, Sarmad Dheyaa Azeez; software, Sarmad Dheyaa Azeez; validation, Sarmad Dheyaa Azeez; formal analysis, Sarmad Dheyaa Azeez; investigation, Saadaldeen Rashid Ahmed; resources, Sarmad Dheyaa Azeez; data curation, Sarmad Dheyaa Azeez; writing—original draft preparation, Sarmad Dheyaa Azeez; writing—review and editing, Abu Saleh Musa Miah; visualization, Muhammad Ilyas; supervision, Muhammad Ilyas; project administration, Sarmad Dheyaa Azeez; funding acquisition, Fahmid Al Farid, Md. Hezerul Abdul Karim. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The dataset generated during the study is available and can be shared by the corresponding authors upon reasonable request.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

CNNs	Including Convolutional Neural Networks
RNNs	Recurrent Neural Networks
IDS	Intrusion Detection System
SIEM	Security Information and Event Management
IoT	Internet of Things
DoS	Denial of Service
LSTM	Long Short-Term Memory
SVM	Support Vector Machine

References

1. Vinayakumar R, Alazab M, Soman KP, Poornachandran P, Al-Nemrat A, Venkatraman S. Deep learning approach for intelligent intrusion detection system. IEEE Access. 2019;7:41525–50. doi:10.1109/ACCESS.2019.2895334.
2. Ahanger AS, Khanam A, Masoodi FS, Pandow BA. A deep learning approach for the detection of zero-day attacks. In: Deep learning for intrusion detection: techniques and applications. Hoboken, NJ, USA: John Wiley & Sons, Inc.; 2025. p. 267–83. doi:10.1002/9781394285198.ch13.
3. Wang W, Zhao M, Wang J. Effective Android malware detection with a hybrid model based on deep autoencoder and convolutional neural network. J Ambient Intell Human Comput. 2019;10(8):3035–43. doi:10.1007/s12652-018-0803-6.
4. Ebrahimian M. Efficient detection of shillings attacks in collaborative filtering recommendation systems using hybrid deep learning: a CNN-based model and architecture [master's thesis]. Toronto, ON, Canada: Ryerson University; 2024. doi:10.32920/25266652.
5. Wu J, Fu Q, Wang L. MARNet: an efficient two-stage intrusion detection model based on deep learning. IEEE Access. 2025;13:2377–88. doi:10.1109/ACCESS.2024.3523938.
6. Tian Z, Cui L, Liang J, Yu S. A comprehensive survey on poisoning attacks and countermeasures in machine learning. ACM Comput Surv. 2023;55(8):1–35. doi:10.1145/3551636.
7. Alghamdi R, Bellaiche M. An ensemble deep learning based IDS for IoT using Lambda architecture. Cybersecurity. 2023;6(1):5. doi:10.1186/s42400-022-00133-w.
8. Musafer H, Abuzneid A, Faezipour M, Mahmood A. An enhanced design of sparse autoencoder for latent features extraction based on trigonometric simplexes for network intrusion detection systems. Electronics. 2020;9(2):259. doi:10.3390/electronics9020259.
9. Awotunde JB, Misra S. Feature extraction and artificial intelligence-based intrusion detection model for a secure Internet of Things networks. In: Illumination of artificial intelligence in cybersecurity and forensics. Cham, Switzerland: Springer; 2022. p. 21–44. doi:10.1007/978-3-030-93453-8_2.
10. Huang T, Chakraborty P, Sharma A. Deep convolutional generative adversarial networks for traffic data imputation encoding time series as images. Int J Transp Sci Technol. 2023;12(1):1–18. doi:10.1016/j.ijtst.2021.10.007.
11. Gao X, Chen K, Zhao Y, Zhang P, Han L, Zhang D. A zero-shot learning-based detection model against zero-day attacks in IoT. In: Proceedings of the 2024 9th International Conference on Electronic Technology and Information Science (ICETIS); 2024 May 17–19; Hangzhou, China. p. 309–14. doi:10.1109/ICETIS61828.2024.10593684.
12. Alsameraee AAA, Ibrahem MK. Toward constructing a balanced intrusion detection dataset: toward constructing a balanced. Samarra J Pure Appl Sci. 2021;2(3):132–42. doi:10.54153/sjpas.2020.v2i3.86.
13. Sharukh SM. A hybrid deep learning approach for detecting zero-day malware attacks. In: Machine learning technologies and applications. Singapore: Springer; 2021. p. 203–10. doi:10.1007/978-981-33-4046-6_20.
14. Sameera N, Jyothi MS, Lakshmaji K, Neeli VSRPK. Clustering based intrusion detection system for effective detection of known and zero-day attacks. J Adv Zool. 2023;44(4):969–75. doi:10.17762/jaz.v44i4.2423.
15. Sharipuddin S, Winanto EA, Purnama B, Kurniabudi K, Stiawan D, Hanapi D, et al. Enhanced deep learning intrusion detection in IoT heterogeneous network with feature extraction. Indones J Electr Eng Inform. 2021;9(3):747–57. doi:10.52549/.v9i3.3134.
16. Bharati S, Podder P. Machine and deep learning for IoT security and privacy: applications, challenges, and future directions. Secur Commun Netw. 2022;2022(1):8951961. doi:10.1155/2022/8951961.
17. Mohanta BK, Jena D, Satapathy U, Patnaik S. Survey on IoT security: challenges and solution using machine learning, artificial intelligence and blockchain technology. Internet Things. 2020;11(8):100227. doi:10.1016/j.iot.2020.100227.
18. Fatani A, Dahou A, Al-qaness MAA, Lu S, Elaziz MA. Advanced feature extraction and selection approach using deep learning and aquila optimizer for IoT intrusion detection system. Sensors. 2021;22(1):140. doi:10.3390/s22010140.
19. Liang R, Gao Y, Zhao X. Sequence feature extraction-based APT attack detection method with provenance graphs. Sci Sin-Inf. 2022;52(8):1463. doi:10.1360/ssi-2021-0252.

20. Aboaoja FA, Zainal A, Ghaleb FA, Ali Saleh Al-rimy B, Eisa TAE, Elnour AAH. Malware detection issues, challenges, and future directions: a survey. Appl Sci. 2022;12(17):8482. doi:10.3390/app12178482.
21. Kang G. Artificial intelligence for threat detection: leveraging deep learning to identify zero-day attacks in real time. Univers Libr Eng Technol. 2025;2(4):74–9. doi:10.70315/uloap.ulete.2025.0204013.
22. Wongvorachan T, He S, Bulut O. A comparison of undersampling, oversampling, and SMOTE methods for dealing with imbalanced classification in educational data mining. Information. 2023;14(1):54. doi:10.3390/info14010054.
23. Sapre S, Islam K, Ahmadi P. A comprehensive data sampling analysis applied to the classification of rare IoT network intrusion types. In: Proceedings of the 2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC); 2021 Jan 9–12; Las Vegas, NV, USA. p. 1–2. doi:10.1109/ccnc49032.2021.9369617.
24. Saikam J, Ch K. A comprehensive review of machine learning and deep learning techniques for addressing class imbalance issues in network intrusion detection systems. In: Proceedings of the 2023 6th International Conference on Recent Trends in Advance Computing (ICRTAC); 2023 Dec 14–15; Chennai, India. p. 678–83. doi:10.1109/ICRTAC59277.2023.10480850.
25. Narender M, Yuvaraju BN. Deep regularization mechanism for combating class imbalance problem in intrusion detection system for defending DDoS attack in SDN. J Comput Sci. 2023;19(3):334–44. doi:10.3844/jcssp.2023.334.344.
26. Koca M, Avcı İ. Enhancing network security: a comprehensive analysis of intrusion detection systems. Yüzüncü Yıl Üniversitesi Fen Bilimleri Enstitüsü Dergisi. 2024;29(3):927–38. doi:10.53433/yyufbed.1545033.
27. Koca M, Ali Aydin M, Sertbaş A, Zaİm AH. A new distributed anomaly detection approach for log IDS management based on deep learning. Turk J Elec Eng Comp Sci. 2021;29(5):2486–501. doi:10.3906/elk-2102-89.
28. Koca M, Avci I. A novel hybrid model detection of security vulnerabilities in industrial control systems and IoT using GCN+LSTM. IEEE Access. 2024;12(1):143343–51. doi:10.1109/ACCESS.2024.3466391.
29. Avcı İ, Koca M. Predicting DDoS attacks using machine learning algorithms in building management systems. Electronics. 2023;12(19):4142. doi:10.3390/electronics12194142.
30. Zhang X, Wu T, Zheng Q, Zhai L, Hu H, Yin W, et al. Multi-step attack detection based on pre-trained hidden Markov models. Sensors. 2022;22(8):2874. doi:10.3390/s22082874.
31. Gamal M, Abbas HM, Moustafa N, Sitnikova E, Sadek RA. Few-shot learning for discovering anomalous behaviors in edge networks. Comput Mater Contin. 2021;69(2):1823–37. doi:10.32604/cmc.2021.012877.
32. Sarhan M, Layeghy S, Moustafa N, Gallagher M, Portmann M. Feature extraction for machine learning-based intrusion detection in IoT networks. Digit Commun Netw. 2024;10(1):205–16. doi:10.1016/j.dcan.2022.08.012.
33. Soltani M, Ousat B, Jafari Siavoshani M, Jahangir AH. An adaptable deep learning-based intrusion detection system to zero-day attacks. J Inf Secur Appl. 2023;76(1):103516. doi:10.1016/j.jisa.2023.103516.
34. Park YS, Lim YS. Hybrid multi-stage framework for identifying zero-day attacks and known threats in network traffic. Comput Netw. 2026;275:111875. doi:10.1016/j.comnet.2025.111875.
35. Sheluhin OI, Rybakov SY, Zvezhinsky SS. Detection of zero-day cyber attacks and zero-day malware using machine learning methods. Radioengineering. 2025;89(8):184–98. doi:10.18127/j00338486-202508-21.
36. Omran Almagrabi A. A deep CNN-LSTM-based feature extraction for cyber-physical system monitoring. Comput Mater Contin. 2023;76(2):2079–93. doi:10.32604/cmc.2023.039683.
37. Pandi VS, Shobana D, Prabhu V. Hybrid deep learning architectures for scalable security in IoT and edge computing environments. In: Hybrid machine learning techniques for data encryption, anomaly detection, and zero-day attack prevention. Coimbatore, India: RADemics Research Institute; 2025. p. 440–71. doi:10.71443/9788197933608-16.
38. Mvula PK, Branco P, Jourdan GV, Viktor HL. Evaluating word embedding feature extraction techniques for host-based intrusion detection systems. Discov Data. 2023;1(1):2. doi:10.1007/s44248-023-00002-y.
39. Corizzo R, Zdravevski E, Russell M, Vagliano A, Japkowicz N. Feature extraction based on word embedding models for intrusion detection in network traffic. J Surveill Secur Saf. 2020;1(2):140–50. doi:10.20517/jsss.2020.15.
40. Zou Q, Guan W. Intrusion detection method based on Wasserstein generative adversarial network. In: Proceedings of the 2022 2nd International Conference on Frontiers of Electronics, Information and Computation Technologies (ICFEICT); 2022 Aug 19–21; Wuhan, China. p. 599–603. doi:10.1109/ICFEICT57213.2022.00109.

41. Wang C, Wang W, Dong J, Guo G. Research on network intrusion detection technology based on DCGAN. In: Proceedings of the 2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC); 2021 Mar 12–14; Chongqing, China. p. 1418–22. doi:10.1109/IAEAC50856.2021.9390891.
42. Li F, Ma W, Li H, Li J. Improving intrusion detection system using ensemble methods and over-sampling technique. In: Proceedings of the 2022 4th International Academic Exchange Conference on Science and Technology Innovation (IAECST); 2022 Dec 9–11; Guangzhou, China. p. 1200–5. doi:10.1109/IAECST57965.2022.10062178.
43. Cirillo F, Esposito C. Intrusion detection system based on quantum generative adversarial network. In: Proceedings of the 17th International Conference on Agents and Artificial Intelligence; 2025 Feb 23–25; Porto, Portugal. p. 830–8. doi:10.5220/0013397800003890.
44. Dener M, Al S, Orman A. STLGBM-DDS: an efficient data balanced DoS detection system for wireless sensor networks on big data environment. IEEE Access. 2022;10(1):92931–45. doi:10.1109/ACCESS.2022.3202807.
45. Boopathi M. Hybrid optimization based deep stacked autoencoder for routing and intrusion detection. Web Intell. 2025;23(1):3–22. doi:10.3233/web-230109a.
46. Bi J, Xu L, Yuan H, Zhang J. Web traffic anomaly detection using a hybrid spatio-temporal neural network. In: Proceedings of the 2023 IEEE International Conference on Systems, Man, and Cybernetics (SMC); 2023 Oct 1–4; Honolulu, HI, USA. p. 5009–14. doi:10.1109/SMC53992.2023.10394549.
47. Kilichev D, Turimov D, Kim W. Next-generation intrusion detection for IoT EVCS: integrating CNN, LSTM, and GRU models. Mathematics. 2024;12(4):571. doi:10.3390/math12040571.
48. Ngo VD, Vuong TC, Van Luong T, Tran H. Machine learning-based intrusion detection: feature selection versus feature extraction. Clust Comput. 2024;27(3):2365–79. doi:10.1007/s10586-023-04089-5.
49. Xing L, Wang K, Wu H, Ma H, Zhang X. Intrusion detection method for internet of vehicles based on parallel analysis of spatio-temporal features. Sensors. 2023;23(9):4399. doi:10.3390/s23094399.
50. Yousefnezhad M, Hamidzadeh J, Aliannejadi M. Ensemble classification for intrusion detection via feature extraction based on deep learning. Soft Comput. 2021;25(20):12667–83. doi:10.1007/s00500-021-06067-8.
51. Pavan Jyothi Swaroop T, Dileep S, Leela Krishna Murthy S, Rajesh A, Pachhala N. Enhanced intrusion detection system using SVM and random forest on UNSW-NB15 dataset. IJARCCE. 2025;14(2):529–33. doi:10.17148/IJARCCE.2025.14272.
52. Barach J. Enhancing intrusion detection with CNN attention using NSL-KDD dataset. In: Proceedings of the 2024 Artificial Intelligence for Business (AIxB); 2024 Dec 2–4; Laguna Hills, CA, USA. p. 15–20. doi:10.1109/AIxB62249.2024.00009.
53. Qiu J, Yang W. Optimization of UAV intrusion detection based on LSTM-RNN. In: Proceedings of the 2024 8th International Conference on Electrical, Mechanical and Computer Engineering (ICEMCE); 2024 Oct 25–27; Xi'an, China. p. 1436–42. doi:10.1109/ICEMCE64157.2024.10862571.
54. Putro IH. Performance comparison of SVM kernels for intrusion detection system using UNSW-NB15 dataset. J Tek Elektro. 2024;17(2):55–61. doi:10.9744/jte.17.2.55-61.