International Journal of
Human Computations and Intelligence
Vol. 05, Issue. 03, 2026

RESEARCH ARTICLE OPEN ACCESS

AI-Assisted Cloud Security Framework For Intrusion Detection, Threat Intelligence, and Real-Time Anomaly Prediction

K Hema[1] . R Mogesh[2] . N Adarsh Naga Siva[2] . K Mohini[2] . A Muni Sai Chandu[2]

[1]Department of CSE, Siddharth Institute of Engineering and Technology, Puttur, A.P, India.
[2]Department of CSE (Cloud Computing), Siddharth Institute of Engineering and Technology, Puttur, A.P, India.

DOI: **10.5281/zenodo.18996613**
Received: 23 January 2026 / Revised: 8 March 2026 / Accepted: 12 March 2026
*Corresponding Author: *goldenhema@gmail.com*

Abstract – The rapid development and deployment of cloud computing services have totally changed the paradigm of how data is managed and applications are delivered. However, the highly dynamic nature of cloud computing infrastructure is prone to various security risks. The traditional security devices, such as rule-based firewalls and signature-based intrusion detection systems, are not effective in countering the complex nature of cyber threats. In this paper, we propose an AI-based framework that provides intelligent intrusion detection and real-time prediction of anomalies in the cloud computing infrastructure. The proposed framework is based on a hybrid AI model that incorporates various AI algorithms, such as Logistic Regression, Decision Trees, Long Short-Term Memory (LSTM), Convolutional Neural Networks (CNN), and Isolation Forest, to name a few. The framework provides real-time monitoring and intrusion detection, thus protecting the cloud computing infrastructure from various potential threats. The proposed framework provides high accuracy in intrusion detection, reaching up to 99.47% accuracy, 99.69% precision, and 99.43% F1-score, thus demonstrating the effectiveness of the proposed model. This work highlights the need to design adaptive, scalable, and intelligent security devices to counter the changing nature of cloud computing security threats.

Index Terms – AI-assisted cloud security, intrusion detection, anomaly prediction, hybrid AI model, Logistic Regression, Decision Trees, LSTM, CNN, Isolation Forest, threat intelligence, cloud computing security, real-time monitoring, machine learning, deep learning.

ISSN (Online): 2583-5696
Int. Jr. of Hum Comp. & Int.

I. INTRODUCTION

The recent growth of cloud computing has revolutionized the way data is managed and applications are deployed. Although cloud computing offers many benefits, such as scalability, flexibility, and cost savings, it also poses various security risks. Cloud computing is a changing and distributed infrastructure with multi-tenancy, which makes it very vulnerable to attacks from hackers. Traditional security systems, such as rule-based firewall systems and signature-based intrusion detection systems (IDS), are not considered adequate to counter the complexity of the threats that exist today [1]. Recent developments in machine learning and deep learning have improved the ability to detect and mitigate cyber threats in cloud computing. These technologies, including the use of hybrid models like Logistic Regression, Decision Tree, LSTM, CNN, and Isolation Forest, have shown tremendous potential in identifying anomalies in real-time and improving the accuracy of intrusion detection with minimal false alarms [2], [3]. It is evident that there is an urgent need for developing adaptable, scalable, and intelligent security solutions, as existing solutions are failing to identify new types of cyber threats, zero-day exploits, and advanced persistent threats (APTs) [4], [5].

It has been found that AI-based solutions have been quite effective at processing and analyzing vast amounts of cloud-based data. Such as network activity, system logs, and user behavior, to identify hidden patterns and predict malicious activity. Machine learning-based solutions, especially supervised and unsupervised learning models, have been found to be quite effective in identifying both known and unknown threats, making it a more reliable solution for protecting systems from more sophisticated and unknown threats [6], [7]. Moreover, the incorporation of threat intelligence with AI-based solutions also helps improve the overall accuracy of the solution, making it more proactive in addressing emerging threats [8], [9]. We are proposing a framework for AI-assisted cloud security that focuses on intrusion detection systems and anomaly prediction in real-time, as well as integrating threat intelligence. The framework will be based on a hybrid AI model that makes use of a combination of Logistic Regression, Decision Tree, LSTM, CNN, and Isolation Forest algorithms in order to monitor cloud infrastructures and be able to detect potential anomalies and even predict potential threats before they cause damage [10], [11]. The solution aims to provide a scalable, adaptive, and intelligent defense mechanism that addresses modern cloud security challenges and ensures robust protection against cyber threats [12].

Our main Contributions are as follows:

- We present a new hybrid model of Artificial Intelligence that incorporates Logistic Regression, Decision Tree, LSTM, CNN, and the Isolation Forest algorithm for intelligent intrusion detection systems. The proposed model will utilize the benefits of all the above-mentioned algorithms to provide accurate results in intrusion detection systems with fewer false alarms.
- Our system will monitor the cloud infrastructure in real-time to analyze the network traffic, system logs, and user activities. This will help in the early detection of various security attacks and anomalies that may take place in the cloud infrastructure. This addresses the major drawback of conventional security systems that fail to provide timely detection of security attacks and anomalies.
- Our system has been experimentally validated to provide an accuracy of 99.47%, precision of 99.69%, and F1-score of 99.43%, thus improving the performance of cloud security systems.

- The effectiveness of our system has also been validated through experiments and shown to offer 99.47% accuracy, 99.69% precision, and 99.43% F1-score, thus making it even better than existing cloud security systems.
- The framework also incorporates real-time threat intelligence feeds, thus making it even better than existing systems since it will be able to proactively detect potential threats by correlating them with existing cloud activity and known attack patterns.
- The AI-assisted cloud security framework that we are proposing is also flexible and adaptable, and will be able to perform well even in dynamic and diverse cloud computing systems.
- The AI-assisted cloud security framework is also different from traditional systems in that it is able to dynamically adjust itself in case of new attack patterns and is able to detect zero-day attacks, advanced persistent threats, and insider attacks, which are often not detected by traditional systems.

II. LITERATURE SURVEY

Threat intelligence in real-time has also been considered as another dimension towards enhancing security in clouds. Andres et al. [13] suggested a threat intelligence framework using real-time AI to analyze the cloud logs, network traffic, and security alerts in order to detect new cyber threats. Their findings demonstrate that they detect threats about 30 % quicker, false positives decrease by half, and zero-day attacks are detected more than 95 % faster, though there are also apprehensions about the high computational price and its implications for privacy. In an attempt to reduce the shortcomings that are inherent in traditional rule-based systems, AI-based approaches have been implemented for advanced threat detection in cloud environments. Singh [14] has introduced a security architecture based on AI, which analyzes network logs, API logs, and user activity data using machine learning and deep learning methods in order to detect anomalies. The model has improved threat detection and threat response capabilities, but it faces the problem of false positives, scalability, and a lack of real-world implementation.

Afraji et al. [15] introduced a deep learning-based defense mechanism that uses datasets, such as CICIDS2017, CICDDoS2019, UNSW-NB15, NSL-KDD, BOT-IoT, and IoTID20, to identify sophisticated attack patterns. Depending on their results, they departmentalize their findings as detection accuracies that are above 99% in some instances; however, the issue of dataset imbalance, increased computational load, and practical issues of real-time deployment remain unanswered. He et al. [16] presented an LSTM-1DResNet model which incorporates the Long Short -term Memory units and Conv1D-ResNet blocks in an auto-encoder as well as a Multilayer Perceptron classifier, thus improving feature extraction of high-dimensional network traffic data. The model performed well on both NSL-KDD and CICIDS2017, with 94.38 % accuracy. Nonetheless, the strategy is limited by the high cost of computation and the need to implement the technology in the real world.

A hybrid IDPS framework was suggested by Srinivasan and Senthilkumar [17] and integrates machine learning and deep learning to monitor and eliminate cyberattacks in large-scale IIoT networks. Using NSL-KDD and IIoT network traffic data, the system was able to find about 99% of the time, which meant that both precision and recall went up. However, worries about false positives, extra computing power needed, and the difficulty of implementing it in real time still make it hard for it to become widely used. To improve the intrusion detection functionality in cloud computing systems, Nagamani et al. [18] came up with the Horse Herd Optimization with Deep Learning–based Intrusion Detection Approach (HHODL-IDA). Overcoming the constraints of the traditional IDS to process a large volume of cloud traffic and advanced attack vectors. The study using a typical IDS benchmark dataset provided about 99% detection accuracy. However, the method is limited by not only expensive computation, but also requires huge training corpora and real-time implementation.

Govindrajan [19] suggested a machine-learning-based detection framework, claiming that predictive models used on cloud security are often biased by class imbalance. The models were reported to have a high 100% detection performance using a benchmark intrusion dataset, but issues of possible overfitting, dependence on the dataset, and lack of real-world deployment remain a concern. SecFedDNN proposed by Alamir et al. [20] is a federated deep-learning-based intrusion detection system, utilizing a DNN model with LASA aggregation due to the privacy threats and the cost of communication linked to centralized IDS. The observed results on the TON_IoT dataset, averaged between various federated clients, were 84 % or higher accuracy. However, the framework has low detection effectiveness of complex attacks and has difficulties due to the nature of federated training.

In order to enhance adaptive intrusion detection, Yu et al. [21] suggested a hybrid AI-based model of spectral clustering and semi-supervised SVM since conventional signature-based IDS tend to inadequately identify emerging and sophisticated cyber threats. The intrusion datasets considered to evaluate the model include CICIDS2017, and the model was found to achieve more than 99%detection accuracy. Nevertheless, the method still has such disadvantages as the complexity of calculations, the requirement to use very large labeled datasets, and problems with deployment in real time. Haider et al. [22] suggested a hybrid deep learning-based intrusion detection system based on BiLSTM and BiGRU since conventional IDS are unable to capture the temporal relationship between network traffic. The model tested on the CIC-IDS2018 data set has 96.7% accuracy with a very low rate of false positives as well as better zero-day attack detection, but the complexity of the model and use of labeled data are still a challenge.

Ahmad et al. [23] introduced VECGLSTM, a hybrid framework of VLSTM, capsule networks, and Enhanced Gannet Optimization Algorithm and CCPTSO-based secure key management due to problems of high false positives and low flexibility of traditional cloud security frameworks under changing attack patterns. The model was tested using cloud intrusion data and demonstrated an accuracy of 99.675 % with good precision, recall, and F1-score. Nevertheless, the framework continues to be limited in terms of high computational complexity and real-life validation of the clouds. Arun et al. [24] suggested a hybrid deep learning intrusion detection model owing to the fact that conventional IDS cannot detect sophisticated cyber-attacks in massive cloud traffic. The model demonstrated to the extent of 99% detection accuracy and better precision and recall rates with benchmark datasets like CICIDS2017, but high computational cost and large labeled datasets are weaknesses.

Jiang et al. [25] suggested a deep learning architecture based on LSTM with an attention mechanism to detect user behavior anomalies. The model was able to predict threat occurrence with an accuracy of 96.8 using multi-tenant cloud behavioral datasets and was able to predict an incident almost 142 minutes before it occurred, but the high cost of computation and reliance on behavioral data of high quality are still limitations. Rout et al. [26] suggested FSEGM, a feature selection and ensemble generative deep learning architecture of cloud security due to the high-dimensional traffic and changing attack patterns of traditional IDS. The model has obtained approximately 99% detection accuracy, but it has a high computational cost, relies on labeled datasets, and presents real-time deployment issues. Hanisha and

Mohan [27] suggested a hybrid ensemble intrusion detection model, which is a Decision Tree, SVM, and Neural Networks, due to the fact that conventional IDS usually cannot achieve a high precision with a high degree of interpretability. The model was tested on CICIDS2017 and NSL-KDD datasets and obtained approximately 98-99% detection with minimal false positives, but the framework is still characterized by model complexity and still requires a lot of labeled training data.

TABLE I: Overview of the existing works

Ref. No.	Dataset	Model / Approach	Result	Limitation
[13]	Cloud logs, network traffic, security alerts	Real-time AI Threat Intelligence Framework	Threat detection 30% faster; false positives reduced by 50%; zero-day detection 95% faster	High computational cost; privacy concerns
[14]	Network logs, API logs, user activity data	AI-based architecture using ML and DL	Improved threat detection and response capability	False positives; scalability issues; lack of real-world implementation
[15]	CICIDS2017, CICDDoS2019, UNSW-NB15, NSL-KDD, BOT-IoT, IoTID20	Deep learning-based defense mechanism	Detection accuracy above 99%	Dataset imbalance; high computational load; real-time deployment issues
[16]	NSL-KDD, CICIDS2017	LSTM-1DResNet with Autoencoder and MLP classifier	94.38% accuracy	High computational cost; real-world implementation challenges
[17]	NSL-KDD, IIoT network traffic data	Hybrid ML and DL based IDPS framework	Around 99% detection accuracy	False positives; computational overhead; real-time implementation difficulty
[18]	IDS benchmark dataset	HHODL-IDA (Horse Herd Optimization with Deep Learning)	About 99% detection accuracy	Expensive computation; large training datasets required; real-time deployment issues
[19]	Benchmark intrusion dataset	Machine learning-based detection framework	Claimed 100% detection performance	Overfitting risk; dataset dependency; lack of real-world validation
[20]	TON_IoT dataset	SecFedDNN (Federated Deep Learning IDS with LASA aggregation)	Average accuracy ≥84%	Lower detection of complex attacks; federated training challenges
[21]	CICIDS2017	Spectral clustering + Semi-supervised SVM hybrid model	Detection accuracy above 99%	High computational complexity; large labeled dataset requirement; real-time issues
[22]	CIC-IDS2018	Hybrid BiLSTM + BiGRU deep learning model	96.7% accuracy with low false positives	Model complexity; reliance on labeled datasets

[23]	_	VECGLSTM (VLSTM + Capsule Network + Optimization algorithms)	99.675% accuracy	High computational complexity; lack of real-world validation
[24]	CICIDS2017	Hybrid deep learning intrusion detection model	Around 99% detection accuracy	High computational cost; large labeled dataset requirement
[25]	Multi-tenant cloud behavioral dataset	LSTM with attention mechanism	96.8% accuracy; predicted threats ~142 minutes earlier	High computational cost; reliance on high-quality behavioral data
[26]	_	FSEGM (Feature Selection and Ensemble Generative DL model)	Approximately 99% detection accuracy	High computational cost; dependence on labeled datasets; real-time deployment issues
[27]	CICIDS2017, NSL-KDD	Hybrid ensemble model (Decision Tree + SVM + Neural Network)	98–99% detection accuracy with low false positives	Model complexity; requirement of large labeled training datasets

III. Module

Dashboard: This Intrusion Detection page allows users to examine network traffic data to determine whether the connection is normal or malicious. The page is necessary due to the fact that modern networks produce an enormous flow of traffic, and it is hard to detect suspicious activity by hand without the structured analysis interface. In this dashboard, the user is enabled to add important network parameters like session time, Source Bytes, Destination Bytes, Failed Logins, and login status. These inputs are also important pointers of network behaviour and help the system identify the unusual patterns that can be an indication of a possible attempt at intrusion. Once the necessary values are entered, a user can press the button to analyze the traffic and process the information. The data is then analyzed through a backend detection model to classify the traffic by the system, thus allowing users to quickly detect possible security threats.

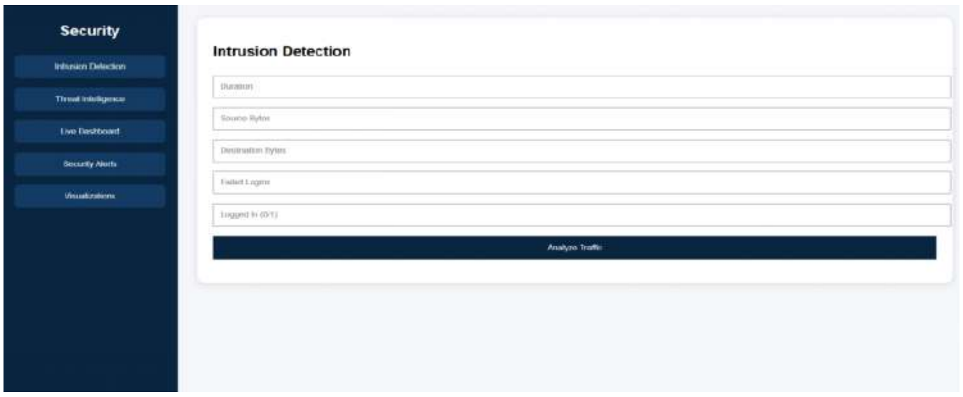
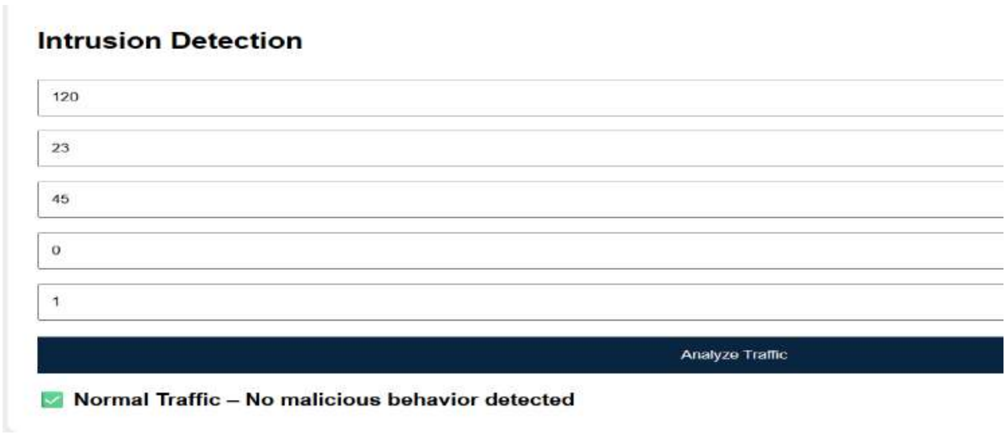


Fig. 1: Dashboard

Traffic Detection: This Traffic Detection page shows the outcome of the traffic analysis upon providing the user with the set of network parameters to be tested. The user is first required to provide key network indicators of network behavior through entry of important network details, including session duration, source bytes, destination bytes, failed logins, and login status. After providing the information, the user clicks the "Analyze Traffic" button to start the security check. This is how the system takes the input data and applies it to its detection model to establish whether the traffic pattern is routine. Once the analysis is done, the result is clearly displayed on the screen in the system. In this instance, the output states Normal Traffic - No malicious activity found and is highlighted with a green indicator, which informs the user that the network traffic under investigation does not seem to be dangerous.


Intrusion Detection

120

23

45

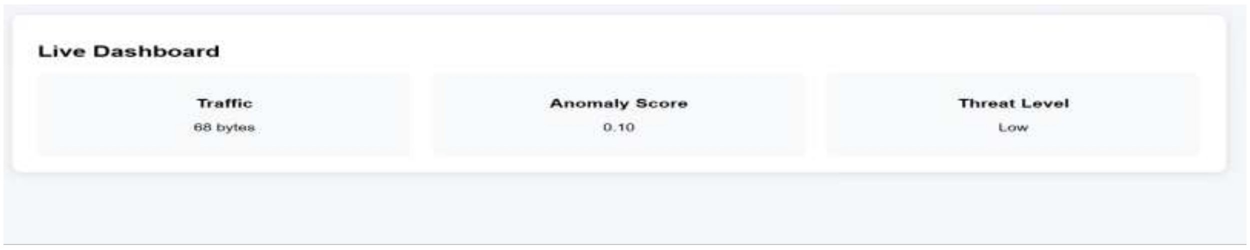
0

1

✓ Normal Traffic – No malicious behavior detected

Fig. 2: Traffic Detection

Live Dashboard: The Live Dashboard page is the Live Dashboard that gives a brief description of what is going on and the security of the network. It displays important real-time indicators like volume of traffic, anomaly score, and threat level to enable users to have an idea of what is happening in the network. In the given case, the dashboard displays 68 bytes of traffic with an anomaly score of 0.10, which is associated with nearly normal behavior. Through this analysis, the system identifies the situation as at a low threat level and allows users to promptly verify that the network is stable and is not exhibiting indicators of being under attack.


Live Dashboard

Traffic	Anomaly Score	Threat Level
68 bytes	0.10	Low

Fig. 3: Live Dashboard

Network Metrics: This Network Metrics page shows a visual illustration of important network metrics to enable the user to know the current level of network activity within a short time. The data that is provided by the system is presented in a bar chart, and this allows easier interpretation of significant indicators. It compares the level of traffic and unsuccessful attempts to enter the system, and accordingly, a user can easily notice any suspicious activity. The chart in this case illustrates that there were 68 bytes of traffic and 0 failed login attempts, so the network was not in a state of instability. The visualization also allows users to easily understand the security condition of the network and the health of the entire system.

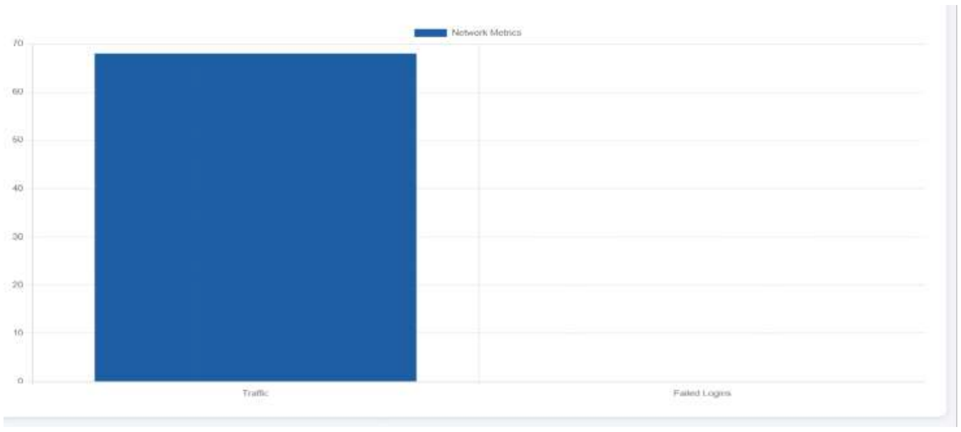


Fig. 4: Network Metrics

IV. METHODS & MATERIALS

A. Dataset Description

We use the NSL-KDD dataset from the Kaggle repository to assess and validate the proposed intrusion detection and anomaly prediction methodology. A improved version of the KDD Cup 1999 dataset, NSL-KDD was created to solve imbalance and redundancy problems that might skew machine learning models. This dataset contains connection records captured from simulated military network traffic, where each record represents a single network interaction between two hosts. Every interaction is described through a set of 41 features that characterize various aspects of the connection, such as duration, protocol type, service, flag status, byte counts, and other statistical measurements over time windows. The rich set of features allows detection models to distinguish between normal traffic and different types of malicious behavior. The NSL-KDD dataset contains labeled instances that are categorized into different types of attacks and normal traffic. There are some main classes, including normal traffic and four types of attacks: Denial of Service (DoS), Probe, User to Root (U2R), and Remote to Local (R2L). In terms of the size of the dataset, the NSL-KDD dataset contains tens of thousands of records, categorized as a training set and different test sets. This makes the dataset appropriate for benchmarking the performance of intrusion detection systems, feature learning, and different algorithms for different types of attacks.

B. Data Preprocessing

The NSL-KDD dataset comprises a variety of network connection records with categorical and numerical attributes. However, prior to the application of any machine learning algorithm, the features of the dataset require careful preprocessing. The preprocessing phase discusses issues regarding feature encoding, normalization, and dimensionality reduction, and the target variable.

- Feature Assignment and Categorization: Initially, features were categorized according to their semantic meaning:

- Connection Feature F_c
- Traffic Feature F_t
- Basic Attack Features F_b
- Login Feature F_l
- System Features F_s
- User Features F_u
- Time-based Traffic Features F_{tb}

Let the complete feature set by represented as:

$$F = F_c \cup F_t \cup F_b \cup F_l \cup F_s \cup F_u \cup F_{tb} \cup F_{hb}$$

The target variable, y, is described in both multi-class and binary-class formats, with:

$$y_i = \begin{cases} 0 \ \ if \ connection \ is \ normal \\ 1 \ if \ connection \ is \ an \ intrution \end{cases}$$

- Missing Values and Redundancy: Early investigation verified that neither the training nor the test sets had any missing values:

$$\sum_{i,j} 1_{NaN}(X_{i,j}) = 0$$

Highly correlated numerical features were identified and extracted to decrease redundancy and multicollinearity:

$$R = \{f_i \ \in F \mid \exists \, f_j \ \in F, corr\left(f_i, f_j\right) > 0.95, i \ \neq j\}$$

The cleaned feature set is then:

$$\acute{F} = F \setminus R$$

- Categorical Encoding: One-hot encoding was applied to convert the categorical variables protocol_type, service, and flag into binary indicator vectors:

$$x_i^{(cat)} \longrightarrow [x_{i1}^{(cat)}, x_{i2}^{(cat)}, \dots \dots \dots, x_{ik}^{(cat)}]$$

making sure it works with algorithms that need numerical input.

- Feature Scaling: Numerical features $x_i^{(num)} \in \acute{F}$ were standardize to zero mean and unit variance employing:

$$\tilde{x}_i^{(num)} = \frac{x_i^{(num)} - \mu_i}{\sigma_i}$$

where, μ_i and σ_i define the mean and standard deviation calculated from the training set. Scalling is crucial for algorithms sensitive to feature magnitude.

- Target Transformation: The binary intrusion label was created by mapping the multi-class assault labels, making the detection process easier:

$$y_i = \begin{cases} 0 \ if \ attack = normal \\ \quad 1 \ otherwise \end{cases}$$

- Final Dataset: Following preprocessing, the dataset is prepared for machine learning modeling using completely numeric, scaled features with aligned columns across training and testing sets:

$$X_{train}, X_{test} \in \mathbb{R}^{n \times d}, y_{train}, y_{test} \in \{0,1\}^n$$

Robust and effective model training for intrusion detection is made possible by this methodical preprocessing, which guarantees that categorical and numerical heterogeneity, feature redundancy, and scaling difficulties are addressed.

C. Proposed Ensemble Transformer Model

We present an Ensemble Transformer-based architecture implemented in the fog–cloud continuum for real-time anomaly prediction in order to facilitate intelligent intrusion detection across the NSL-KDD dataset. The goal is to enhance robustness against complex and infrequent attack patterns by combining ensemble variety with contextual sequence modelling.

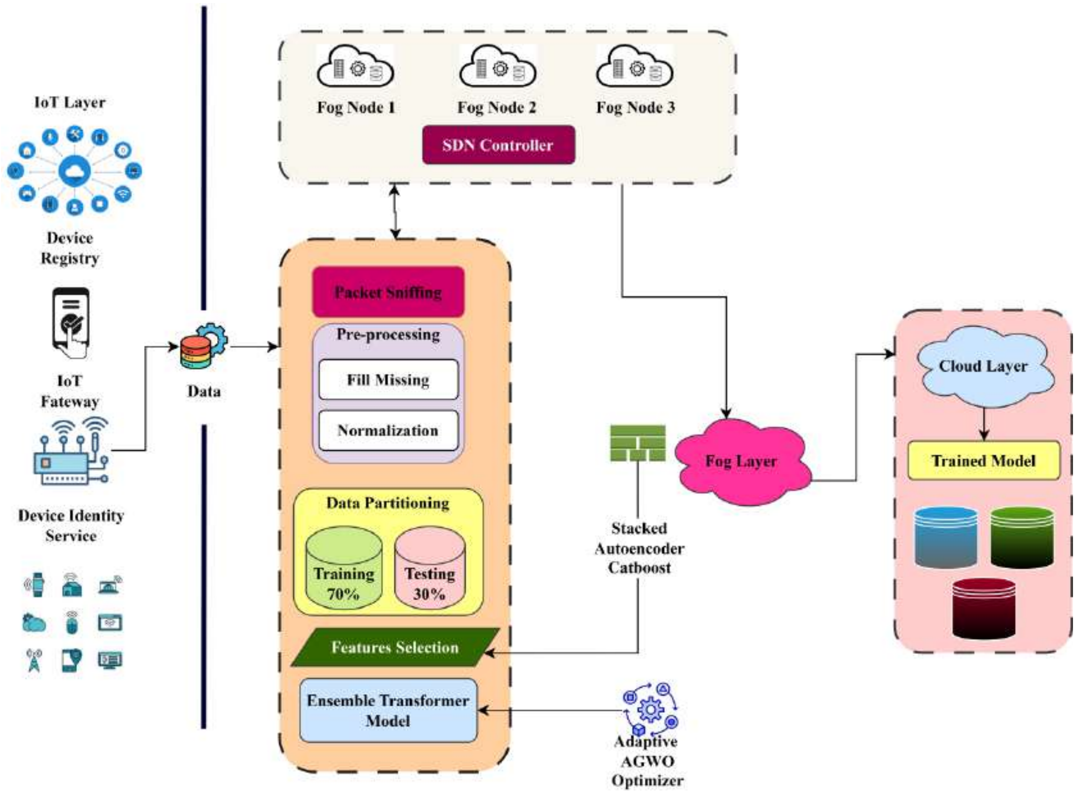


Fig. 5: System Architecture for AI-Assisted Cloud Security Framework for Intrusion Detection and Real-Time Anomaly Prediction

- Input Representation: Let the preprocessed network traffic sample be expressed as:
$$x_i = [x_{i1}, x_{i2}, \dots \dots .., x_{id}] \in \mathbb{R}^d$$
where d represents the number of engineered features following feature selection and normalization. A latent embedding space is created by projecting each input vector:
$$x_i = W_e x_i + b_c$$
where, $W_e x_i$ is the embedding matrix, and b_c is the bias term.
To maintain feature-order dependency, positional encoding is used:
$$h_i^{(0)} = z_i + PE_i$$
- Transformer Encoder Block: Multi-Head Self-Attention (MHSA) and a position-wise Feed Forward Network (FFN) make up each encoder layer.

- Multi-Head Attention

$$\text{Attention (Q, K, V)} = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

For head j:

$$head_j = Attention\ (HW_Q^j, HW_K^j, HW_j^V)$$

The concatenated heads:

$$\text{MHSA (H)} = \text{Concat}\ (head_1, \ldots\ldots, head_h)W^o)$$

Residual and normalization:

$$\acute{H} = LayerNorm\ (H + MHSA\ (H))$$

- Feed Forward Network

$$FFN(\acute{H}) = \sigma\left(\acute{H}W_1 + b_1\right)W_2 + b_2$$

Final encoder output:

$$H^{(l)} = LayerNorm\ (\acute{H} + FFN(\acute{H}))$$

After L stacked encoder layers:

$$\text{F} = H^{(L)}$$

- Ensemble Learning Strategy: We build an ensemble of K independently trained Transformer learners rather than depending on a single Transformer classifier:

$$M = [T_1, T_2, \ldots\ldots\ldots, T_K]$$

Every learner generates a probability distribution:

$$P_k = Softmax(W_c^{(k)}F_k + b_c^{(k)})$$

where, C is the number of intrusion classes.

- Aggregation Mechanism: We use a soft voting system that is weighted:

$$P_{final} = \sum_{k=1} \alpha_k p_k$$

subject to:

$$\sum_{k=1}^{K} \alpha_k = 1,\ \alpha_k \geq 0$$

The predicted class:

$$\hat{y} = arg\ \underset{c}{max}\ P_{final}(c)$$

To reduce cross-entropy loss, Adaptive Gray Wolf Optimization (AGWO) is used to optimize the ensemble weights α_k:

$$\mathcal{L} = \sum_{i=1}^{N} \sum_{c=1}^{C} y_{ic} log(P_{final,ic})$$

TABLE 2: Hyperparameter Settings of the Proposed Model

Component	Hyperparameter	Value
Input Layer	Input Dimension	41 (after selection)
	Embedding Dimension	128
Transformer Encoder	Number of Encoder Layers	4
	Number of Attention Heads	8
	Key/Query Dimension	16
	Feed Forward Hidden Units	256

	Activation Function	ReLU
	Dropout Rate	0.2
Ensemble Layer	Number of Transformer Learners	3
	Voting Strategy	Weighted Soft Voting
	Ensemble Weight Range	[0,1]
Optimization	Optimizer	Adam
	Learning Rate	0.0005
	Batch Size	64
	Epochs	100
	Loss Function	Categorical Cross-Entropy
Regularization	L2 Weight Decay	1e-5
Class Imbalance Handling	Class Weights	Inverse Frequency

D. Deployment Diagram

The deployment diagram shows the physical realization of the proposed AI-assisted CNID framework, which maps the software modules to the hardware nodes in a distributed manner. It shows the structural realization of how IoT devices, fog nodes, and centralized cloud computing collaborate to achieve real-time anomaly detection, high availability, and scalability using the trained model obtained from the NSL-KDD.

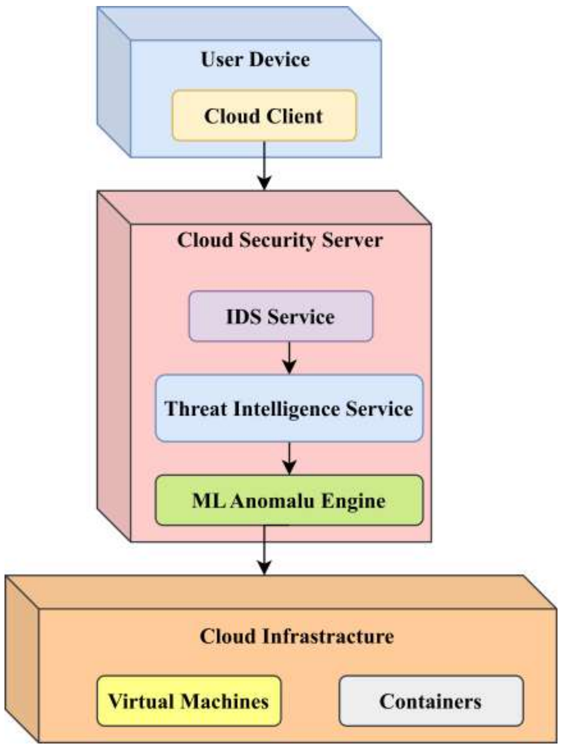


Fig. 6: Deployment Figure

At the edge layer, IoT devices, heterogeneous devices, and users are the sources that produce continuous flows of traffic. These devices are connected to each other using IoT gateways that filter the devices, authenticate the devices, and securely transmit the data using encrypted communication protocols. The gateway is the first aggregation point that sends the structured metadata to the fog layer, which is mapped to a physical or virtual edge server with constrained computing capabilities to achieve minimal latency and efficient bandwidth utilization. The fog layer will be made up of various distributed fog nodes located near the data source. The fog nodes will be deployed on a virtualized platform, which will enable the system to be horizontally scaled. The fog node will be notified when the abnormal traffic probabilities go beyond the set threshold, and it will be required to generate an alarm and contact the central AI security engine for confirmation.

The AI Security Engine will be located mainly at the cloud layer, with synchronized lightweight replicas located at the fog nodes for redundancy. The AI security engine will be the one that contains the Ensemble Transformer, weight optimizer, retraining scheduler, and attack intelligence components. The components will be deployed on cloud instances with GPU accelerators for batch retraining and model refinement. The AI security engine will be required to aggregate logs and output from the fog nodes at periodic intervals for global learning and updating of the system to increase sensitivity to emerging threats. A web server will be deployed at the cloud layer for the hosting of RESTful APIs for the system. The web server will be required to connect with the AI security engine and the backend storage for the system. The web server will be protected from unauthorized access by the use of firewall and identity-based access control mechanisms.

The Security Dashboard is a visualization tool for network administrators that provides visualizations of anomaly scores, intrusion classifications, traffic statistics, fog node status, and system health status in real-time. It is linked to the monitoring subsystem and constantly updates its visualizations with information such as CPU usage, memory usage, latency values, and accuracy values from different nodes in the network. This provides administrators with a measure of stability and helps them prevent issues before they occur. The diagram is not only useful for visualizing the physical layout of the system but also for understanding its scalability and robustness. The horizontal scaling of the system is achieved through fog nodes, and vertical scaling is achieved through cloud resources. The load balancing and container orchestration ensure smooth version updates of the AI models.

V. HARDWARE AND SOFTWARE SPECIFICATIONS

The proposed framework for intrusion detection and real-time anomaly prediction using AI-assisted cloud-native would require both hardware and software components to perform efficiently. The hardware components would include servers with high-end CPUs or GPUs to perform the computational tasks involved in real-time data processing, model inference, and training. The hardware would also require sufficient memory (RAM) to perform the processing and training of significant volumes of data, especially during model training, and would require sufficient storage capacity to store extensive logs, data, and activities. The framework would operate in a distributed cloud environment, and this would require the use of fog nodes and cloud computing to perform the model training. On the software side, the system will leverage AI and machine learning libraries such as TensorFlow, Keras, Scikit-learn, and PyTorch to support the development and training of the models. For the processing and storage of the data, the system will leverage big data processing tools such as Hadoop, Spark, and Kafka to process the large amounts of streaming data and log information generated by the cloud environment. The anomaly and intrusion detection system will utilize a hybrid model of different machine learning algorithms like Logistic Regression, Decision Trees, LSTMs, CNNs, Isolation Forests, etc. The system will utilize threat intelligence feeds in real-time, a backend database management system like SQL Server or a NoSQL database to store the data, and containerization tools like Docker and Kubernetes to scale the system on different cloud platforms.

VI. RESULTS AND DISCUSSIONS

A. Performance of the Models

Table III shows a comparison between the Ensemble Transformer model you designed and some prominent baselines using the NSL-KDD dataset. The trend is obvious: as the models become more complex and can represent more, the performance improves accordingly.

TABLE III: Comparative Performance Evaluation

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	AUC (%)
Logistic Regression	92.84	93.10	91.72	92.40	93.05
Decision Tree	95.63	96.02	94.88	95.44	95.51
LSTM	97.54	97.80	97.11	97.45	97.62
CNN	97.89	98.14	97.42	97.78	97.95
Proposed Ensemble Transformer	**99.47**	**99.69**	**99.16**	**99.43**	**99.45**

Logistic Regression scored 92.84% accuracy, 93.10% precision, 91.72% recall, 92.40% F1, and 93.05% AUC. It is fast and easy to interpret, but the linear decision boundary cannot capture the complex nonlinear relations in high-dimensional network traffic features. The relatively low recall rate also means more false negatives, which is undesirable in intrusion detection. Decision Trees, however, scored much higher, at 95.63% accuracy and 96.02% precision, along with 94.88% recall and 95.44% F1. The AUC is also high, at 95.51%, indicating a clear separation between classes. The hierarchical structure is helpful in capturing some nonlinear relations, but the instability and tendency to overfitting are major drawbacks for this model.

For the deep learning models, LSTM scored 97.54% accuracy, 97.80% precision, 97.11% recall, 97.45% F1, and 97.62% AUC. It is good at capturing the sequence information in traffic patterns, but the recall rate is a little lower, suggesting some intrusion patterns may not be easy to detect. CNN is closely followed by CNN with 97.89% accuracy, 98.14% precision, 97.42% recall, 97.78% F1, and 97.95% AUC. These filters pick up local feature interactions and spatial correlations of structured traffic vectors. However, CNNs are more inclined towards local patterns and often fail to recognize feature relationships over longer distances. The Proposed Ensemble Transformer is different and stands out from other models with 99.47% accuracy, 99.69% precision, 99.16% recall, 99.43% F1, and 99.45% AUC. This model is 1.58% more accurate and 1.74% more precise than CNN, indicating higher sensitivity towards attacks. Precision is boosted from 98.14% to 99.69%, greatly reducing false positives.

With precision and recall scores of 99.69% and 99.16%, respectively, it is evident that the F1 score is 99.43%, indicating robust performance for majority and minority classes. Similarly, the AUC is 99.45%, showing robust discrimination between normal and malicious traffic. This comparison clearly indicates that combining diversity with Transformers is advantageous over traditional ML and DL techniques for cloud-native intrusion detection systems.

B. Performance Evaluation and Classification Analysis

The four visualizations illustrate the robustness and generalizability of the proposed CNN-based intrusion detection model shown in the Figure 7. In the training confusion matrix, it is observed that the model classifies instances very accurately. In total, 53,720 true negatives and 46,514 true positives are correctly classified. However, there are 143 false positives, i.e., normal instances incorrectly classified as attack instances, and 394 false negatives, i.e., attack instances incorrectly classified as normal instances. These observations suggest a clear class separation with very few misclassifications. In intrusion detection, false negatives are more dangerous because they pose a greater security risk than false positives. These observations suggest that the CNN model is highly effective in learning discriminative patterns from the feature space of the NSL-KDD dataset.

The testing confusion matrix is similar to the training confusion matrix and supports the generalizability of the proposed CNN-based intrusion detection model. In total, 13,447 normal connections and 11,598 attack instances are correctly classified among unseen testing instances. Similarly, there are 33 false positives and 115 false negatives. These observations suggest that there is very little overfitting because there is a similar pattern of error rates between training and testing instances. More importantly, false negatives are very low, suggesting that the model maintains its sensitivity for intrusion detection even for new patterns of network connections.

The ROC curve for the training set shows a near perfect degree of separability, with the AUC at 0.9945. The curve hugs the upper left corner of the graph, indicating that the true positive rate is very high while the false positive rate remains exceptionally low. The high AUC value confirms the CNN model's excellent capability for distinguishing between benign and malicious traffic.

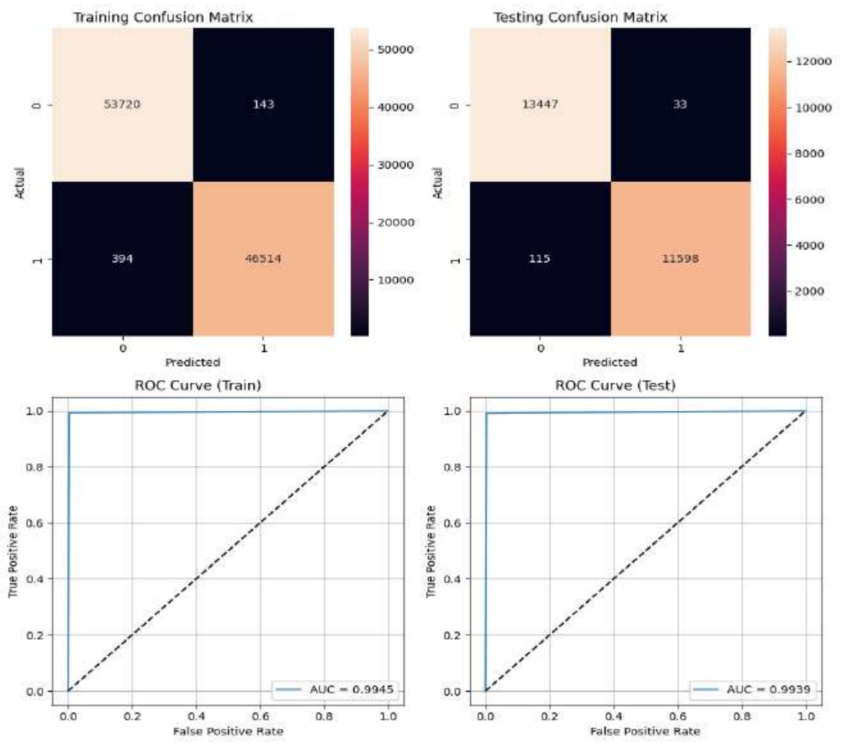


Fig. 7: Classification Performance and Discriminative Analysis of the Proposed Model

The ROC curve for the testing set also shows a high AUC value of 0.9939, which is very close to the training set's value. The slight difference between the two values confirms the model's stability and robustness. The curve climbs sharply towards the upper left, which indicates the classifier's sensitivity without compromising specificity. The consistency of the values for the training and testing sets confirms that the CNN model learned the intrusion patterns and not the data. Overall, the confusion matrices and the ROC curves have provided solid confirmation that the CNN model has achieved excellent classification accuracy, high attack recall rate, and low false alarm rate with excellent generalization capability.

C. SHAP Analysis

To grasp what our Proposed Ensemble Transformer Model is doing in terms of the NSL-KDD data, we used SHAP (Shapley Additive exPlanations) to measure the influence of features on the model's judgment about a given data point being a normal or an attacking connection which displayed in Figure 8. SHAP values are a way to compute a game-theoretic measure of the influence of each feature, and we can then use these values to order features by their strength in pulling the model in a particular direction.

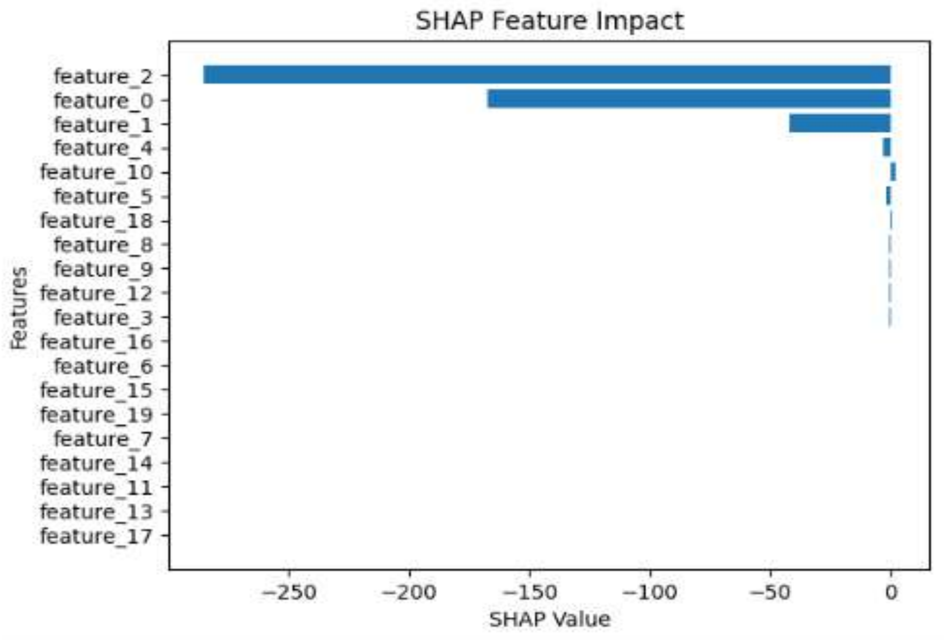


Fig. 8: Classification Performance and Discriminative Analysis of the Proposed Model

The results from SHAP, as mentioned in Figure 8, reveal a hierarchy in the influence of features. The results reveal that, overall, the influence of feature_2, which had a SHAP value of -285.18, is the most negative, suggesting that as the value in this feature is high, the model is more likely to not make an attacking judgment. Feature_0, with a SHAP value of -167.63, and feature_1, with a SHAP value of -42.06, also had significant negative influence, suggesting that as the values in these features are high, the model is more likely to make a normal judgment. As for the positive influence, feature_10 had the most significant influence, with a SHAP value of 2.02, suggesting that as the values in this feature are high, the model is more likely to make an attacking judgment. Other features, such as feature_5, with a SHAP value of -1.63, and feature_4, with a SHAP value of -2.95, also had some influence, although not as significant as the ones mentioned above.

The remaining features, such as feature_3, with a SHAP value of -0.24, and feature_9, with a SHAP value of -0.29, had lower SHAP values. These features, although not significant individually, together help the model make judgments in terms of what is normal and what is not.

D. Feature Importance Analysis

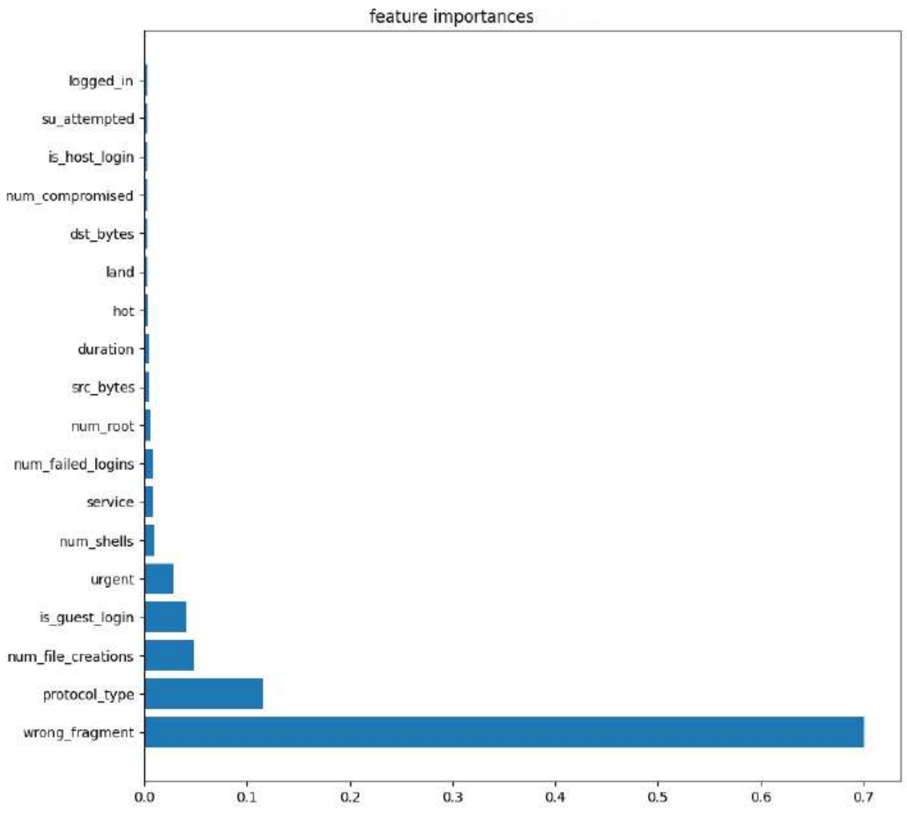


Fig. 9: Classification Performance and Discriminative Analysis of the Proposed Model

The feature importance analysis offers insights into the decision-making process of the proposed model when it comes to the NSL-KDD dataset as shown in Figure 9. According to the figure, wrong_fragment is the dominant feature when it comes to the decision-making process, with a contribution of approximately 0.70 towards the overall importance score. The dominance of wrong_fragment suggests that the model considers the presence of fragmented packets as an essential factor when distinguishing between malicious and normal connections. The importance of wrong_fragment can be attributed to the common intrusion techniques used by attackers, where unusual fragmentations of packets are associated with evasion techniques and exploits.

The second major factor considered by the model when making the final decision is protocol_type, which has an importance of approximately 0.12. The dominance of protocol_type indicates that the protocol used for the connection (TCP, UDP, ICMP, etc.) also influences the final decision made by the model. The influence of protocol_type is also evident from the common intrusion detection techniques, where different types of attacks have been associated with different protocols. The next major factors considered by the model when making the final decision include the moderate importance of features such as num_file_creations (~0.05), is_guest_login (~0.04), and urgent (~0.03). On the other hand, features such as num_shells, service, num_failed_logins, num_root, src_bytes, duration, hot, land, dst_bytes, num_compromised, is_host_login, su_attempted, and logged_in have minimal importance and contribute marginally towards the overall importance score. Even though the individual importance of these features is minimal, the overall contribution of these features towards the decision-making process of the model adds to the overall robustness of the model, where the model considers the various nuances associated with the network connection.

VII.CONCLUSION

In this paper, we propose an AI-based cloud-native intelligent intrusion detection system with real-time anomaly prediction to meet the growing security challenges in cloud computing. In our system, we utilized a hybrid model of AI that combines Logistic Regression, Decision Trees, LSTM, CNN, and the Isolation Forest algorithm to attain high accuracy in the detection of anomalies. From our experiments, we have demonstrated that our system can attain high accuracy in the detection of anomalies with 99.47% accuracy, 99.69% precision, and 99.43% F1-score compared to the state of the art. The addition of real-time threat intelligence also improves the system's effectiveness in the detection and response to emerging threats, making it a proactive system as opposed to the traditional rule-based system. The framework is also flexible and scalable, and can be adapted to emerging threats, making it comprehensive in protecting the organization against various traditional and non-traditional cyber threats, including zero-day threats and APTs. There are rooms for improvement in the future, where more sophisticated deep learning methods can be incorporated, federated learning for data privacy can be employed, and incident response can be made more autonomous. Additionally, there is a need for improvement in the integration of this approach with other multi-cloud and hybrid platforms, making it applicable to other clouds as well. Through continued innovation and improvement of this approach, we can create a more robust and intelligent cloud security solution that can keep pace with the dynamic nature of cyber threats.

REFERENCES

1. Cantero, B., Zalazar, B., Pezzini, B., Gaitán, C., Bertucci, C., & Mazzanti, C. (2026). *Machine learning based anomaly detection for multi-cloud security monitoring*. ResearchGate.
2. Govindarajan, V. (2026). An advanced AI-driven framework for intelligent security risk detection using hybrid mechanisms. *SN Applied Sciences*.
3. Mathivanan, N. (2026). An integrated methodology for intrusion detection and zero trust security in cloud environments. *Journal of Cloud Computing*.
4. Singh, R., Kumar, A., & Mishra, S. (2025). Attentional LSTM-ensemble architecture for intrusion detection. *Scientific Reports, 15*(3).
5. Dissanayake, N., & Thayasivam, U. (2025). Attack-specialized deep learning with ensemble fusion for network anomaly detection. *arXiv preprint*.
6. Alhusseini, M. M., Rouhi, A., & Feizi-Derakhshi, M. R. (2026). AI-powered hybrid intrusion detection framework for cloud security using metaheuristic optimization. *arXiv preprint*.
7. Saraswathi, S. (2025). Hybrid anomaly detection model integrating LSTM and isolation forest for enhanced performance. In *Proceedings of the International Conference on Inventive Research in Computing Applications (ICIRCA)*.
8. Kumar, A. (2025). Impact of machine learning on intrusion detection and anomaly detection. *Information, 16*(7).
9. Ferozuddin, F. (2025). AI-driven anomaly detection model for intrusion detection. *International Journal of Computer Applications, 187*(6).
10. Evaluation of anomaly-based network intrusion detection systems. (2025). *Security Journal*.
11. Anomaly detection in heterogeneous cybersecurity data. (2025). *Fraud Analytics and Optimization*.

12. Cybersecurity in cloud computing: AI-driven intrusion detection and mitigation strategies. (2025). *IEEE Access* (Accepted for publication).
13. Andrés, P., Ivanov, I., & Wang, Z. (2025). Real-time AI-based threat intelligence for cloud security enhancement. *Innovative: International Multi-disciplinary Journal of Applied Technology, 3*(3), 36–54.
14. Singh, H. (2025). Enhancing cloud security posture with AI-driven threat detection and response mechanisms. *SSRN Electronic Journal*.
15. Afraji, D. M. A. A., Lloret, J., & Peñalver, L. (2025). Deep learning-driven defense strategies for mitigating DDoS attacks in cloud computing environments. *Cyber Security and Applications, 3*, 100085.
16. He, Q., Zhang, Y., Xu, A., Ye, Z., Zhou, W., Lin, Q., & Zhang, T. (2026). LSTM-1DResNet: An intrusion detection model for connected and autonomous vehicles based on deep learning. *IEEE Transactions on Vehicular Technology*.
17. Srinivasan, M., & Senthilkumar, N. C. (2025). Intrusion detection and prevention system (IDPS) model for IIoT environments using hybridized framework. *IEEE Access, 13*, 26608–26621.
18. Nagamani, S., Arivalagan, S., Senthil, M., & Sudhakar, P. (2025). Horse herd optimization with deep learning-based intrusion detection in cloud computing environment. *International Journal of Information Technology, 17*(1), 387–393.
19. Govindrajan, V. (2025). Machine learning-based approach for handling imbalanced data for intrusion detection in the cloud environment. In *Proceedings of the 3rd International Conference on Disruptive Technologies (ICDT)* (pp. 810–815).
20. Alamir, R. H., Noor, A., Almukhalfi, H., Almukhlifi, R., & Noor, T. H. (2025). SecFedDNN: A secure federated deep learning framework for edge–cloud environments. *Systems, 13*(6), 463.
21. Yu, T., Huang, W., Tang, X., & Zheng, D. (2025). A hybrid unsupervised machine learning model with spectral clustering and semi-supervised support vector machine for credit risk assessment. *PLOS ONE, 20*(1), e0316557.
22. Haider, Z. A., Zeb, A., Rahman, T., Khan, F. M., Khan, I. U., Sohail, Q., Bilal, H., Khan, M. A., & Ullah, I. (2025). Optimizing cloud security with a hybrid BiLSTM-BiGRU model for efficient intrusion detection. *ICCK Transactions on Sensing, Communication, and Control, 2*(2), 106–121.
23. Ahmad, S., Arif, M., Mehfuz, S., Ahmad, J., & Nazim, M. (2025). Deep learning-based cloud security: Innovative attack detection and privacy-focused key management. *IEEE Transactions on Computers*.
24. Arun, M., Barik, D., Othman, N. A., Praveenkumar, S., & Tudu, K. (2025). Investigating the performance of AI-driven smart building systems through advanced deep learning model analysis. *Energy Reports, 13*, 5885–5899.
25. Jiang, X., Jia, R., & Zhang, F. (2025). Deep learning-based user behavior anomaly detection and threat early warning in cloud computing environments. *Academia Nexus Journal, 4*(3).
26. Rout, C., Sethi, S., Badajena, J. C., & Sahoo, R. K. (2026). FSEGM: Feature selection and ensemble generative model for adaptive cloud security. *Journal of Cloud Computing, 27*(1), 6.
27. Hanisha, V., & Mohan, M. (2026). A hybrid ensemble model for intrusion detection in cloud environment with transparent and explainable AI techniques. In *Recent advances in computational methods in science and technology* (pp. 28–33). Boca Raton, FL: CRC Press.
28. Ahmed, S. T., Basha, S. M., Ramachandran, M., Daneshmand, M., & Gandomi, A. H. (2023). An edge-AI-enabled autonomous connected ambulance-route resource recommendation protocol (ACA-R3) for eHealth in smart cities. *IEEE Internet of Things Journal*, *10*(13), 11497-11506.
29. Ahmed, S. T., & Fathima, A. S. (2024). Medical ChatBot assistance for primary clinical guidance using machine learning techniques. *Procedia Computer Science*, *233*, 279-287.
30. Periasamy, K., Periasamy, S., Velayutham, S., Zhang, Z., Ahmed, S. T., & Jayapalan, A. (2022). A proactive model to predict osteoporosis: An artificial immune system approach. *Expert Systems*, *39*(4), e12708.