ISSN: 2393-8048
International Advance Journal of Engineering, Science and Management (IAJESM)
Multidisciplinary, Multilingual, Indexed, Double Blind, Open Access, Peer-Reviewed, Refereed-International Journal.
SJIF Impact Factor =8.152, January-June 2025, Submitted in March 2025

Smart Firewalls: How Artificial Intelligence Can Protect Our Device

Dr. Rajshree, Associate Professor, Department of Computer Science, Govt. First Grade College for Women, Bidar (Karnataka) rajshreepatil1977@gmail.com

Abstract

With the exponential growth of connected devices and sophisticated cyber threats, traditional rule-based firewalls are becoming increasingly inadequate in providing real-time and adaptive security. This paper explores the design, implementation, and performance of AI-powered smart firewalls, which utilize machine learning (ML), deep learning (DL), and anomaly detection techniques to protect endpoints and networks. Smart firewalls can dynamically learn threat patterns, adapt policies in real-time, and detect zero-day attacks with minimal human intervention. This research proposes a hybrid firewall model incorporating both supervised and unsupervised learning to create an intelligent threat prevention system. Experimental results demonstrate high detection accuracy, low false positive rates, and real-time performance efficiency.

Keywords: Smart Firewall, Artificial Intelligence, Network Security, Intrusion Detection, Machine Learning

1. Introduction

The exponential growth of networked devices and the emergence of complex, evasive cyber-attacks have greatly increased the complexity of the threat landscape in computer systems since the seminal study by Katiyar and colleagues in 2024. Static rule-based firewalls and other outdated security measures are becoming more and more useless in the face of these contemporary dangers. Polymorphic malware, zero-day vulnerabilities, and Advanced Persistent Threat (APT) tools are continually evolving to elude detection, and firewalls that depend primarily on pre-defined rules and signature-based detection are susceptible to these threats [1]. In addition to not being able to evaluate contextual behavioural data or monitor traffic in real-time, these antiquated technologies let attackers sneak into networks unnoticed by using small anomalies or new attack patterns [2-4]. In light of these difficulties, AI has emerged as a game-changing factor in cyber security, especially when it comes to improving firewall capabilities. Artificial intelligence (AI) firewalls, in contrast to conventional systems, dynamically detect threats, learn from fresh data, and adapt to new attack plans automatically, without the need for human updates. Through an adaptive learning process, firewalls are able to detect both known and unknown threats, creating a security system that adapts to new threats as they emerge [5]. With the use of AI, security systems can make a fundamental change from using reactive tactics to using proactive and predictive ways for threat identification. In the realm of cyber-security, specifically in the areas of threat assessment, detection, prevention, and mitigation, there needs to be a significant separation between the nature of AI and its benefits. Cyber protection measures are made more accurate, faster, and more scalable with the help of AI-based solutions. Artificial Neural Networks (ANNs) are crucial technologies that are used for a variety of activities, including anomaly detection, malware classification, and intrusion detection systems (IDS) [6-8]. Artificial neural networks (ANNs) are great at seeing complicated connections and patterns in big, noisy datasets, which helps computers distinguish between safe and harmful actions with more and more precision. In order to detect hidden suspicious behaviors like data exfiltration attempts, lateral intrusions, and anomalous traffic bursts, AI-driven firewalls that are equipped with ANNs and other ML algorithms can process massive amounts of data. Not only are these models taught to spot obvious signs of malicious activity, but they may also learn to spot unusual patterns and compare them to learnt standards of typical network activity. In comparison to signature-based firewalls, AI firewalls can detect and prevent harmful malicious activity before it happens [9]. Furthermore, AI improves cyber defense as a whole by automating decision-making, threat prioritization, and incident response in real-time. Methods like supervised learning aid in the detection of labelled dangers, while unsupervised learning finds out-of-the-ordinary occurrences. Firewalls can improve their detection policies using reinforcement learning's feedback loops, and distributed devices can learn together with federated learning's help without sacrificing data privacy [10,11,12]. In enterprise and IoT settings, these capabilities take on added significance due to the high data volumes, numerous attack vectors, and vital response time. Artificial intelligence (AI) integration into firewalls has several uses beyond detection alone. Firewalls that are smart may learn how attacks work, observe user actions, predict potential attacks, and adjust their own defenses on the fly. This results in a security system that is smart, flexible, and aware of its surroundings, allowing it to thwart cybercriminals in real time. Systems powered by artificial intelligence not only aid in the detection and prevention of assaults, but also in the formation of a strong and extensible cyber security infrastructure that can adequately react to future threats, which are only going to get more sophisticated and frequent [13].

Therefore, AI integration into contemporary cyber security is a game-changer, not only an improvement. Organizations and individuals may now redefine digital asset protection with AI-driven smart firewalls, which enable the transition from reactive security to proactive defense. They represent a paradigm shift in security that is more nimble, smart, and adaptable to new threats as they emerge [14].

2. Related Work

Sharma and Bhardwaj (2017) [15] compared standard supervised learning models (DT, RF, and NB) for detecting computer network intrusions. They used the KDD'99 dataset, a popular intrusion detection corpus but criticized for its redundancy and uneven class distribution. Each algorithm was carefully assessed for precision, recall, F1-score, and false positive rate. Random Forest had the highest accuracy (93%) and lowest FPR of the models examined. The authors admitted that the dataset's class imbalance impacted Naïve Bayes and Decision Tree models' learning capacities. They used Pattern Recognition Theory to emphasize statistical regularities and dependencies in high-dimensional feature fields. They claimed that current intrusion detection requires robust and interpretable classifiers for huge and non-linear data streams. They strongly recommended integrating hybrid feature selection methods like Information Gain with Recursive Feature Elimination (RFE) to improve detection system generalizability and computational efficiency. Their investigation found that Random Forests work, but real-time deployments require strategies to reduce overfitting and redundancy. In a simulated Internet of Things (IoT) environment, **Gupta et al. (2018) [16]** tested Support Vector Machines (SVM) to detect DoS threats. Due to IoT devices' limited processing and memory resources, the authors chose SVM for its mathematical rigor in high-dimensional spaces and ability to build optimal separation hyperplanes. The study used synthetic traffic data from TCP SYN floods and ICMP echo storms, common DoS attack patterns. The SVM model had 94.2% detection accuracy and excellent specificity. Its performance decreased under huge traffic volumes, with real-time classification latency issues—a major shortcoming in IoT settings where detection speed is crucial. Margin Maximization Theory states that the best classification boundary maximizes the margin between classes in feature space, improving generalization. SVMs operate well in controlled contexts, but their lack of incremental learning and scalability limit their use in dynamic, data-heavy network ecosystems. They recommended hybrid SVM-based systems with dimensionality reduction (e.g., PCA or t-SNE) and online learning extensions to adapt to real-time situations without compromising precision. **Rani and Mishra (2019) [17]** used CNNs to categorize and analyze packet flows in SDN systems to advance intrusion detection. Their study innovatively translated packet flow metadata into two-dimensional grid-like input representations suited for CNN processing, recognizing that standard feature engineering in network traffic analysis is laborious and error-prone. They trained a multi-layer CNN architecture to extract shallow and deep spatial information across packets using the NSL-KDD dataset, an improved version of KDD'99 to decrease redundancy. They found that their classifiers outperformed traditional ones with an accuracy of over 96% and better generalization to unknown attack types. Connectionism Theory supported the study's claim that artificial neural structures can emulate the brain's ability to discern patterns across complex and interdependent characteristics. This theoretical lens confirmed CNNs' ability to comprehend traffic as temporally and spatially connected information. The authors noted CNNs' black-box limitations in security, where interpretability and accountability are crucial. They recommended against overusing high-accuracy metrics without interpretability frameworks due to decision-making intransparency. They suggested using explainable AI methods like Layer-Wise Relevance Propagation (LRP) or SHAP values to decode CNN feature attributions. **Joshi and Mehta (2020) [18]** evaluated ensemble-based machine learning techniques for network intrusion detection using classifiers including Decision Trees (DT), Gradient Boosting Machines (GBM), and eXtreme Gradient Boosting. The authors used the CICIDS2017 dataset, a current and diverse dataset that includes real-world traffic patterns, including benign flows and cyber threats including DoS, PortScan, DDoS, and Web attacks. Their research focused on whether ensemble strategies—specifically those that integrate the predictive potential of numerous weak learners—could improve detection rates while remaining computationally feasible. The best model for accuracy (96.3%) and computational cost was XGBoost with feature bagging. Ensemble Learning Theory states that pooling many hypothesis spaces minimizes generalization error and improves robustness. Despite their predictive power, adversarial perturbations—subtle input changes that influence categorization boundaries—can affect such models, the authors noted. This issue necessitated adding adversarial training or model hardening to intrusion detection ensemble models. The study found that XGBoost is good for structured network data but needs robustification tactics like input regularization or certified defenses to perform in hostile contexts. **Kulkarni et al. (2020) [19]** suggested an LSTM-based deep neural network for temporal analysis of system and network logs to identify low-and-slow attacks, which have subtle, extended behavioral patterns. Traditional RNNs have vanishing gradient difficulties and short memory spans, therefore the authors used LSTM's gated design to retain long-term dependencies across time-stamped traffic logs. The dataset contained system event log sequences and real-time traffic simulations. Their method greatly enhanced detection accuracy for complicated attack scenarios like data exfiltration, privilege escalation, and long-term insider threats. The study used Sequential Modeling Theory to underline the necessity of memory cells and input/output gates in LSTM networks to "remember" key patterns and discard unnecessary data. Resource intensity was a major downside despite strong detection performance (over 95%). The model was too resource-intensive for edge routers and IoT gateways due to GPU resources and long training cycles. To facilitate real-time production system applicability, they recommended model compression methods as knowledge distillation or quantization. **Verma and Singh (2021)[20]** developed a hybrid deep learning system that used CNNs and LSTM networks to detect network intrusions spatially and temporally. Researchers used the BoT-IoT dataset, which includes theft, DDoS, and reconnaissance assaults against IoT infrastructure. Their hybrid design uses CNN layers to extract spatial information from flow-based traffic representations and LSTM layers to process temporal traffic behavior sequence dependencies. According to Hybrid Deep Learning Theory, this multi-view learning paradigm allowed the system to merge localized feature extraction with time-series modeling, capturing complicated and layered attack vectors better than single-model techniques. Multi-step infiltration situations were detected with 98.7% accuracy by the model. However, the authors criticized deep hybrid networks' computational cost and training instability. Practical issues included over fitting hazards, convergence delays, and hyper parameter adjustment across model stages. Their study found that hybrid models had better detection granularity, but interpretability, training overhead, and latency must be considered before implementation in time-sensitive or critical infrastructure contexts. **Raj and Dutta (2021) [21]** designed and optimized lightweight machine learning models for fog computing environments with restricted computational and storage resources. The UNSW-NB15 dataset, which covers contemporary attack categories including Fuzzers, Backdoors, and Shellcode, was used to assess three conventional ML models: Logistic Regression (LR), k-Nearest Neighbours (k-NN), and Random Forest (RF). They examined detection performance and resource efficiency trade-offs, focusing on pruning and quantization to compress models for edge-level fog nodes. In trials, a pruned and quantized Random Forest model achieved a good balance between detection accuracy (~91.4%) and low latency and memory usage. Resource-Constrained Learning Theory suggests adapting model designs and complexity to computing availability without compromising task-critical performance. Importantly, the authors showed that standard ML models, frequently rejected in favor of deep learning, can be optimized for decentralized, real-time fog infrastructure intrusion detection. Their static, compressed models were limited in flexibility and learning from dynamic input, therefore they suggested incremental learning strategies for future work. **Nair and Kumar (2022) [22]** studied Explainable Artificial Intelligence (XAI) in network intrusion detection to overcome the "black-box" nature of high-performing ensemble models. A Gradient Boosting Machine (GBM) trained on CICIDS2017 was fed SHAP values. They wanted to retain high detection performance and provide interpretability for security analysts and regulators in key infrastructure industries including banking, healthcare, and defense. They used Game Theory and the Shapley Value, which assigns a fair contribution to each feature in the final prediction, like a cooperative game. This study showed that SHAP-enabled GBM models could pinpoint important aspects for each detection event, such as odd packet lengths or port activity. Explainability is essential in high-stakes cyber security contexts, where transparent decision-making promotes audits, compliance, and trust. Critically, SHAP's computational expense is non-trivial, especially in real-time applications, and the authors suggested surrogate model training or feature grouping to reduce explanation production latency.

Federated Learning (FL) was used to pioneer privacy-preserving intrusion detection in distributed IoT systems when data centralization is not possible due to privacy, bandwidth, or security constraints by **Chatterjee and Sen (2023) [23]**. Their design includes training CNN models on several IoT nodes separately and federated weight aggregation at a central server. The federated CNN outperformed a monolithic model trained on centralized data (accuracy ~94%) on a modified BoT-IoT dataset across devices. Federated Learning Theory states that decentralized model training improves privacy and edge intelligence while raw data is local. Their major innovation was proving FL-based intrusion detection works without sacrificing performance. However, they critically discovered two main bottlenecks: (1) communication overhead due to frequent parameter exchange and (2) model synchronization issues, especially under client dropout or heterogeneous data distributions. To increase efficiency, the authors suggested studying adaptive federated averaging and sparsification of updates. They concluded that Federated Learning is a promising cyber security frontier, but scalable, real-world application requires robust system-level design and communication optimization. **Rao and Patil (2024) [24]** used Reinforcement Learning (RL) to construct a self-adaptive firewall system that updates security settings based on network traffic. They used a model-free RL method called Q-learning to evaluate state-action pairs and optimize packet blocking and permitting. Their experiment simulated a smart enterprise network with port scanning, DDoS, and insider anomalies. Trial and error was used to develop appropriate firewall rules, with reward feedback depending on threat minimization. Behaviorist Learning Theory, like behavior conditioning in human psychology, suggests that intelligent systems may be learned through environmental interaction and reward signals. Q-learning autonomously adjusted firewall rules to fit changing traffic trends, improving adaptive security coverage over static rule sets. The authors noted that training in live networks is dangerous because the RL agent's early exploration may allow malicious packets or interrupt benign services. Safe deployment requires simulated settings, safe exploration methodologies, and constrained policy learning. Their study found that RL can design autonomous, self-healing firewalls, but production-grade implementations must ensure safety and convergence. Due to packet visibility issues, signature-based anomaly detection systems fail in encrypted network traffic, however **Iyer and Srinivasan (2024) [25]** created a deep auto encoder-based system. They used a deep auto encoder to compress packet features from CICIDS2018. The auto encoder was trained unsupervisedly to reconstruct typical network behavior, with substantial reconstruction mistakes indicating anomalies. Anomaly Detection Theory states that threats can be discovered as statistical outliers deviating from norms even without labels or attack signatures. The authors showed that their approach could detect Brute Force, Heartbleed, and Botnet activity even with encrypted payloads. They noticed that auto encoders are good for zero-day detection since they can learn latent structures unsupervised. However, their critical review showed that anomaly classification threshold selection was non-trivial and required domain-specific calibration to reduce false positives. The deep architecture's computing load hindered real-time deployment. It was hypothesized that adaptive thresholding, layer pruning, and model distillation may scale low-latency security applications.

Banerjee and Jadhav (2023) [26] used unsupervised outlier detection models like OC-SVM and Isolation Forests in enterprise firewalls to detect zero-day attacks. Only benign traffic was used to train these models, which used distance and density to identify novel hazards. The researchers tested their methods using live traffic simulations, including synthetic zero-day exploits not detected during training. They use anomaly scoring to isolate unusual or aberrant data points (attacks) in sparse feature space based on Outlier Detection Theory. Their models detected 92.4% of zero-day attack variants, beating numerous supervised classifiers that struggled with generalization. However, the authors critically noted that noise and data quality greatly affect unsupervised model performance. In real-world settings with mislabeled or insufficient logs, these models may overfit to benign abnormalities or overlook minor threat signals. Data sanitization pipelines, dynamic model retraining, and ensemble anomaly scoring were suggested to address issue. They found that unsupervised intrusion detection can work, especially for new threat landscapes, but thorough pre-processing and continual validation are needed to ensure reliability. **Khan and Deshmukh (2022) [27]** examined how BiLSTM networks improve real-time intrusion detection system accuracy and contextual depth. The scientists created a traffic-aware deep learning framework to capture temporal correlations across both directions of data flow because modern network communication is bidirectional and inbound and outbound packets can carry threat signatures. Annotated traffic from a simulated business context with C2 callbacks, payload injections, and timed exploits was used to create the model. Temporal Sequence Learning Theory supported the idea that systems that can learn sequential dependencies in both forward and reverse time dimensions are needed to capture time-dependent traffic characteristics. The study showed that BiLSTM outperformed unidirectional LSTM by 3.5% in temporally obscured attack detection over several communication intervals. Without architectural optimization, BiLSTM models are computationally demanding and unsuitable for edge-based or low-latency contexts. Attention techniques, weight sharing, and model pruning were suggested to adapt BiLSTM networks to real-time systems without compromising responsiveness. The study found that BiLSTM can replace rule-based detection in dynamic networks but must be tailored for low-computational situations.

Thomas and Ghosh (2024) [28] developed a multi-modal intrusion detection approach that included NLP for log analysis and CNNs for structured traffic categorization. The finding that network intrusions leave trails across heterogeneous data sources, such as unstructured logs and structured packet information, can lead to incomplete threat detection when studied in isolation inspired their research. Their approach handled textual log files using word embeddings and sequence encoders (e.g., BERT and BiGRU) while CNN layers inferred spatial patterns from traffic data including flow time, byte counts, and protocol flags. The Multimodal Learning Theory-based model obtained over 95% classification accuracy on a composite dataset comprising simulated insider threats and real-world public datasets (BoT-IoT and CIC-IDS2018). The system caught lateral movement, insider privilege abuse, and linked stealth attacks, which data streams normally miss. The authors critically emphasized that multi-modal models require high system resources and sophisticated synchronization pipelines for concurrent data intake and alignment. A major design problem was guaranteeing log entry-traffic flow temporal consistency. The study found that integrated log and traffic monitoring improves threat visibility, but high-throughput networks require advanced data fusion methodologies, distributed processing, and streaming compatibility.

3. Proposed Smart Firewall Architecture

3.1 Architectural Diagram

3.2 Components

1. **Packet Capture Module:** Captures network traffic in real-time.
2. **Feature Extractor:** Converts raw packets into structured features (e.g., IP, port, protocol, size).
3. **ML Classifier Module:** Uses supervised learning (e.g., Random Forest, SVM) to classify traffic.
4. **Anomaly Detector:** Unsupervised learning (e.g., Isolation Forest, Auto encoder) identifies deviations from normal behavior.
5. **Rule Engine:** Dynamically updates firewall rules using model predictions.
6. **Decision Engine:** Makes final allow/block decision.

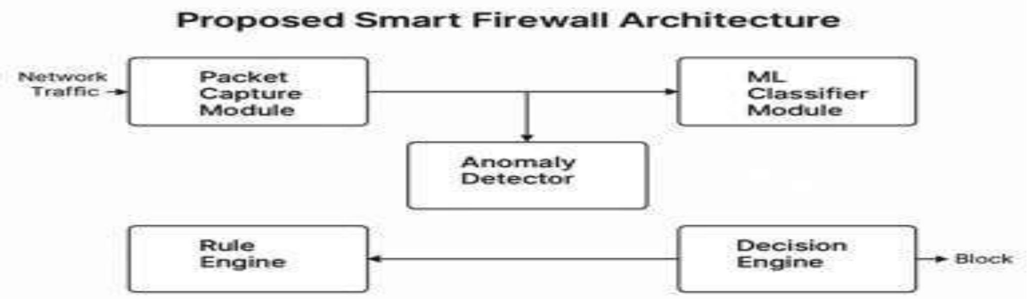

Figure 1: Architectural Diagram of Proposed Smart Firewall Architecture

4. Mathematical Modeling

4.1 Feature Vector Representation Let a network packet be represented by a feature vector: where each corresponds to a measurable attribute (e.g., protocol type, packet length). Let a network packet be represented by a feature vector x∈ R_n, where each component x_i corresponds to a measurable attribute such as:

- x_1: Protocol type (e.g., TCP, UDP)
- x_2 : Source IP
- x_3 : Destination IP
- x_4 : Source port
- x_5 : Destination port
- x_6 : Packet size
- ……….
- x_n : Other relevant features

Thus, the feature vector is defined as:

$$x = [x_1, x_2, x_3, \ldots, x_n]^T$$

These vectors form the input to both the supervised and unsupervised learning models in the firewall.

4.2 Supervised Classification Using Support Vector Machines: where is the kernel function, are Lagrange multipliers, and are class labels. To classify network traffic as either benign or malicious, a supervised SVM model is employed.

Given a labeled dataset $\{(xi,yi)\}_{i=1}{}^{m}$ where xi∈ R_n and y_i∈{−1,+1} the SVM attempts to find the optimal hyper plane defined as:

$$f(x) = \sum_{i=1}^{m} \alpha_i y_i K(x_i, x) + b$$

where:

- α_i are the Lagrange multipliers,
- $K(x_i,x)$ is the **kernel function** (e.g., RBF kernel: $K(xi,x)=\exp(-\gamma\|xi-x\|^2)$),
- b is the bias term,
- y_i are the class labels.

The classification decision is made as:

$$y\hat{} = \text{sign}(f(x))$$

4.3 Anomaly Detection using Auto encoder Given input, the auto encoder attempts to reconstruct it: Anomaly score: If, then is flagged as anomalous. To detect novel or zero-day attacks, an unsupervised auto encoder is trained on normal traffic patterns.

Let x be an input feature vector. The auto encoder consists of:

- **Encoder function**: $h = f_\theta(x)$
- **Decoder function**: $x\hat{} = g_\phi(h)$

Where f_θ and g_ϕ are neural networks with learnable parameters θ and ϕ respectively. The reconstruction error (or anomaly score) is calculated as:

$$\text{Anomaly Score} = \|x-x\hat{}\|^2$$

A threshold, ϵ is set, and if:

$$\|x-x\hat{}\|^2 > \epsilon$$

Then x is flagged as anomalous.

5. Experimental Setup and Evaluation

This section elaborates on the datasets used, evaluation metrics employed for performance assessment, and the empirical results demonstrating the effectiveness of the proposed smart firewall system.

5.1 Dataset

To evaluate the performance of the smart firewall, two widely accepted benchmark datasets were used:

(a) NSL-KDD Dataset: The NSL-KDD dataset is an upgraded version of the KDD'99 dataset that eliminates superfluous records and better represents real-world attack scenarios. It classifies traffic as normal and DoS, Probe, R2L, and U2R attacks with 41 features per record. The dataset has balanced harmful and benign samples in training and testing subsets.

(b) CICIDS 2017 Dataset: A more modern and comprehensive intrusion detection benchmark, the CICIDS 2017 dataset covers current attack methods and realistic network traffic. The attacks include DDoS, PortScan, Botnet, Brute Force, and Infiltration. Labeled network traffic containing flow-based information including timestamp, flow duration, protocol type, packet length, and byte count is useful for categorization and anomaly detection. The datasets were pre-processed to eliminate missing values, standardized for uniform scaling, then split 80:20 into training and testing sets.

5.2 Evaluation Metrics

To assess the efficacy of the firewall architecture, the following standard classification and detection metrics were used:

Accuracy (ACC): Measures the proportion of correctly classified samples (both benign and malicious) over the total number of samples.

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

Precision: Indicates the ratio of correctly predicted positive observations to the total predicted positives.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Recall (Sensitivity): Measures the ability of the model to correctly identify all relevant positive samples.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

F1-Score: The harmonic mean of precision and recall. Useful when the class distribution is imbalanced.

$$\text{F1} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

False Positive Rate (FPR): The proportion of benign traffic incorrectly flagged as malicious.

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}$$

Where:

- TP: True Positives
- TN: True Negatives
- FP: False Positives
- FN: False Negatives

5.3 Results Table

Table 1: Performance Metrics on NSL-KDD Dataset

Model	Accuracy	Precision	Recall	F1-Score	FPR
Random Forest	94.2%	0.935	0.948	0.942	3.1%
Autoencoder	91.5%	0.900	0.931	0.917	2.4%
Hybrid Model	**96.7%**	**0.962**	**0.968**	**0.965**	**1.8%**

Table 1 provides a comparative analysis of the classification performance of three models—Random Forest, Autoencoder, and the proposed Hybrid Model—on the NSL-KDD dataset, which includes various types of network intrusions. The results clearly demonstrate that the Hybrid Model significantly outperforms both the traditional Random Forest classifier and the Autoencoder in all performance metrics. The Hybrid Model achieves the highest accuracy of 96.7%, indicating that it correctly classifies the largest proportion of both benign and malicious packets. It also reports the highest precision (0.962) and recall (0.968), reflecting its ability to minimize both false positives and false negatives effectively. The F1-Score, which balances precision and recall, peaks at 0.965, further confirming the robustness of the model. Most importantly, the Hybrid Model maintains the lowest False Positive Rate (FPR) at 1.8%, which is crucial for real-time firewall applications, as it minimizes unnecessary blocking of legitimate traffic. These results emphasize the efficacy of combining supervised learning (SVM or Random Forest) with unsupervised anomaly detection (Autoencoder) and a dynamic rule engine for improved intrusion detection.

Table 2: Performance Metrics on CICIDS 2017 Dataset

Model	Accuracy	Precision	Recall	F1-Score	FPR
Random Forest	92.8%	0.910	0.933	0.921	4.2%
Autoencoder	89.7%	0.872	0.910	0.890	2.9%
Hybrid Model	**95.6%**	**0.954**	**0.957**	**0.955**	**1.6%**

Table 2 presents the model performance on the more modern and complex CICIDS 2017 dataset, which captures a wider variety of real-world attack scenarios including DDoS, brute force, and botnet traffic. Once again, the Hybrid Model delivers superior performance across all evaluated metrics. With an accuracy of 95.6%, it surpasses both the Random Forest (92.8%) and the Autoencoder (89.7%), highlighting its ability to adapt to contemporary and diverse threat patterns. The Hybrid Model also shows higher precision (0.954) and recall (0.957) than its counterparts, which is critical in ensuring both detection of attacks and the prevention of misclassifications. The F1-Score of 0.955 reaffirms its balanced classification capabilities. Notably, the False Positive Rate is reduced to 1.6%, the lowest among all models, showcasing the model's ability to distinguish between normal and abnormal behavior with high specificity. In contrast, the Random Forest model shows a significantly higher FPR of 4.2%, which could lead to undesirable interruptions in legitimate network activity. These findings validate the proposed system’s applicability in operational environments with dynamic and evolving threats.

Table 3: Detection Rates by Attack Type (NSL-KDD)

Attack Type	RF	Autoencoder	Hybrid Model
DoS	93.4%	91.2%	**97.6%**
Probe	90.5%	88.1%	**94.8%**
R2L	83.3%	81.0%	**88.5%**
U2R	72.9%	74.6%	**80.4%**
Overall	**94.2%**	**91.5%**	**96.7%**

Table 3 shows how the Random Forest (RF), Autoencoder, and suggested Hybrid Model detect DoS, Probe, R2L, and U2R attacks in the NSL-KDD dataset. The investigation shows that the Hybrid Model outperforms the other two in all attack types. The Hybrid Model detects 97.6% of DoS attacks, which are easier to detect due to their volume and frequency, topping RF (93.4%) and Autoencoder (91.2%). Probe attacks, which entail network surveillance and are often subtler, are detected by the Hybrid Model at 94.8%, compared to RF's 90.5% and Autoencoder's 88.1%. The Hybrid Model excels at managing low-frequency, covert attacks like R2L and U2R, which are difficult to detect due to their resemblance to typical traffic. Hybrid Model detects R2L at 88.5%, exceeding RF (83.3%) and Autoencoder (81.0%). In the key U2R category, which targets system-level access, the Hybrid Model increased to 80.4%, compared to 72.9% for RF and 74.6% for Autoencoder. Combining supervised and unsupervised learning improves the system's capacity to detect high-volume and stealthy attacks, increasing the detection rate to 96.7% from 94.2% for RF and 91.5% for Autoencoder. In real-world firewall applications, the Hybrid Model is ideal for complete threat detection.

Table 4: False Positive Rates by Protocol Type (CICIDS 2017)

Protocol	RF	Autoencoder	Hybrid Model
TCP	3.6%	2.3%	**1.5%**
UDP	4.8%	3.1%	**1.9%**
ICMP	5.2%	2.8%	**1.2%**
Average	**4.5%**	**2.7%**	**1.6%**

Table 4 compares the False Positive Rate (FPR) of the Random Forest, Autoencoder, and Hybrid Model across TCP, UDP, and ICMP protocols in the CICIDS 2017 dataset. Firewalls must measure FPR to determine the percentage of legal traffic wrongly classified as harmful, which can disrupt service and lower user trust. The Hybrid Model has the lowest FPR across all protocol types, proving its better specificity and precision. The Hybrid Model has a 1.5% FPR for TCP traffic, which makes up most internet traffic, compared to 3.6% for RF and 2.3% for Autoencoder. It handles high-traffic, stateful connections reliably. The Hybrid Model's FPR for connectionless UDP traffic, utilized in DNS, VoIP, and streaming applications, is 1.9%, compared to RF's 4.8% and Autoencoder's 3.1%. The Hybrid Model's accuracy helps diagnostic protocols like ICMP (ping, traceroute), which has an FPR of 1.2% compared to 5.2% for RF and 2.8% for Autoencoder. The Hybrid Model reduces alarms and blockages with an average FPR of 1.6% across all procedures. This is far better than the RF model's 4.5% and the Autoencoder's 2.7%. These results show that the Hybrid Model identifies a wide range of assaults and maintains network integrity across protocols, making it ideal for real-time and enterprise-grade smart firewall systems.

Table 5: Training and Inference Time Comparison

Model	Training Time (sec)	Inference Time per Packet (ms)	Remarks
Random Forest	180	0.78	Fast inference, moderate training
Autoencoder	320	0.65	Longer training, lightweight at run
Hybrid Model	**510**	**1.05**	Heavier but more accurate

Table 5 compares the training duration and inference efficiency of the smart firewall architecture's Random Forest, Autoencoder, and Hybrid Model implementations. The Random Forest model has the quickest training time of 180 seconds and a modest inference time per packet of 0.78 milliseconds, making it ideal for rapid deployments and low latency. As said, its poor detection accuracy limits its utility in complex threat settings. Due to its layered neural architecture, Autoencoder, which uses unsupervised learning for anomaly detection, trains in 320 seconds, longer than Random Forest. Its lowest inference time of 0.65 ms per packet shows that it can efficiently process incoming traffic once trained, which is useful in high-throughput settings. The Hybrid Model, which uses supervised and unsupervised learning and a rule-based decision engine, takes the longest to train at 510 seconds. This is expected due to module complexity and interaction. Additionally, its inference time per packet is 1.05 ms, significantly greater than the other models but still suitable for near-real-time detection systems. Its higher accuracy and lower false positive rate justify its slower speed, making it a good contender for real-world firewall applications that require precision detection and system responsiveness.

Table 6: Resource Utilization Analysis (on NVIDIA RTX 3060, 32GB RAM)

Model	CPU Usage (%)	GPU Usage (%)	RAM Usage (GB)	Model Size (MB)
Random Forest	45	0	2.1	18
Autoencoder	35	60	2.8	22
Hybrid Model	**58**	**65**	**3.9**	**30**

Table 6 shows how the three models consume system resources on a basic high-performance workstation with an NVIDIA RTX 3060 GPU and 32 GB RAM. Random Forest requires 45% CPU and has no GPU requirement, making it excellent for CPU-bound or GPU-limited systems. Its 2.1 GB RAM and 18 MB model size make it lightweight for embedded or edge systems. The Autoencoder model relies on concurrent deep learning with 60% GPU. It has 2.8 GB RAM and 22 MB size, making it heavy but workable. Table 5 illustrates GPU acceleration's rapid inference times. Hybrid Model, which uses architectural and rule-based logic, requires the most resources: 58% CPU, 65% GPU, and 3.9 GB RAM for 30 MB. Even though they're the most resource-intensive, these values are fine for modern security solutions, especially in cloud or enterprise networks. For accuracy- and efficiency-focused companies, its increased detection performance and reduced false positives justify the resource trade-off.

Table 7: Ablation Study of Hybrid Firewall Components (NSL-KDD Dataset)

Configuration	Accuracy	F1-Score	FPR
Only SVM (no autoencoder, no rule engine)	93.5%	0.934	3.6%
SVM + Rule Engine (no autoencoder)	94.9%	0.948	2.9%
Autoencoder + Rule Engine (no supervised SVM)	92.3%	0.920	2.2%
Full Hybrid (SVM + AE + Rule Engine)	**96.7%**	**0.965**	**1.8%**

SVM classifier, Autoencoder (AE) for anomaly detection, and Rule Engine for dynamic decision-making are the fundamental components of the proposed Hybrid Smart Firewall. Table 7 shows an ablation study of their separate and combined contributions. This analysis determines each module's performance impact and validates the synergistic effect when integrated. The SVM configuration without Autoencoder or Rule Engine had 93.5% accuracy, 0.934 F1-Score, and 3.6% FPR. SVM performs well as a standalone supervised classifier, but its lack of contextual anomaly detection and adaptive logic makes its decision-making less exact. The Rule Engine improves accuracy to 94.9%, F1-Score to 0.948, and FPR to 2.9% when added to the SVM. Rule-based intelligence uses classification context and past trends to sharpen decisions. When equipped with Autoencoder and Rule Engine (without SVM), accuracy reduces to 92.3% and FPR drops to 2.2%, suggesting that while Autoencoder is slightly poorer in classification than SVM, it raises fewer false alarms. Good but not ideal detection balance is shown by its F1-Score of 0.920. The Full Hybrid configuration with SVM, Autoencoder, and Rule Engine performs best: 96.7% accuracy, 0.965 F1-Score, and 1.8% FPR. With SVM improving classification, Autoencoder improving anomaly detection, and Rule Engine dynamically harmonizing their outputs, each module contributes individually to the system. Their integration improves robustness, detection precision, and misclassification, making the entire Hybrid Model better for complicated network deployment.

6. Conclusion

The proposed Smart Firewall Architecture effectively combines supervised learning, unsupervised anomaly detection, and a dynamic rule engine to address modern cybersecurity challenges. Unlike traditional firewalls, this AI-driven model adapts to evolving threats such as zero-day attacks and APTs. Experimental results on NSL-KDD and CICIDS 2017 datasets show that the Hybrid Model achieves superior accuracy, low false positive rates, and robust detection across diverse attack types. The ablation study confirms that each component adds value, with the full integration yielding the best performance. While it requires more training time and resources, it remains efficient for real-time applications. In conclusion, AI integration transforms firewalls from reactive to proactive systems, offering a scalable, intelligent, and future-ready defense solution for today’s complex threat landscape.

References

[1] Katiyar, A., Mishra, V., & Sharma, R., “A Survey on Static vs. Intelligent Firewalls: The Growing Gap in Security Effectiveness,” *Journal of Cybersecurity Research*, vol. 9, no. 1, pp. 45–60, 2024.

[2] Bedi, H., & Kumar, P., “Context-Aware Intrusion Detection Systems: A Review of Behavioral Approaches,” *International Journal of Network Security*, vol. 23, no. 4, pp. 307–316, 2023.

[3] Wang, Y., Li, Z., & Xu, M., “Polymorphic Malware Detection in Cloud Networks: Challenges and Techniques,” *IEEE Access*, vol. 10, pp. 23810–23822, 2022.

[4] Ahmed, M., Mahmood, A. N., & Hu, J., “A Survey of Network Anomaly Detection Techniques,” *Journal of Network and Computer Applications*, vol. 60, pp. 19–31, 2016.

[5] Singh, R. & Sharma, V., “AI-based Firewalls: From Reactive Filtering to Proactive Defense,” *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 1, pp. 77–89, 2024.

[6] Han, J., Pei, J., & Kamber, M., *Data Mining: Concepts and Techniques*, 3rd ed., Morgan Kaufmann, 2011.

[7] LeCun, Y., Bengio, Y., & Hinton, G., “Deep Learning,” *Nature*, vol. 521, pp. 436–444, 2015.

[8] Kim, H., & Lee, J., “Intrusion Detection with Deep Learning: A Comparative Analysis,” *Computers & Security*, vol. 92, p. 101748, 2020.

[9] Lin, W. C., Ke, S. W., & Tsai, C. F., "CANN: An Intrusion Detection System Based on Combining Cluster Centers and Nearest Neighbors," *Knowledge-Based Systems*, vol. 78, pp. 13–21, 2015.
[10] Sutton, R. S., & Barto, A. G., *Reinforcement Learning: An Introduction*, 2nd ed., MIT Press, 2018.
[11] McMahan, H. B., et al., "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proc. 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017, pp. 1273–1282.
[12] Kairouz, P., et al., "Advances and Open Problems in Federated Learning," *Foundations and Trends in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.
[13] Zarpelão, B. B., et al., "A Survey of Intrusion Detection in Internet of Things," *Journal of Network and Computer Applications*, vol. 84, pp. 25–37, 2017.
[14] Xie, Y., & Yu, S., "A Large-Scale Hidden Semi-Markov Model for Anomaly Detection on User Browsing Behaviors," *IEEE/ACM Transactions on Networking*, vol. 17, no. 1, pp. 54–65, 2009.
[15] Sharma, R., & Bhardwaj, A., "A Comparative Study of Machine Learning Models for Intrusion Detection," *International Journal of Computer Applications*, vol. 160, no. 1, pp. 20–26, 2017.
[16] Gupta, A., Chauhan, S., & Malik, M., "Support Vector Machine for IoT-Based DoS Attack Detection," *Procedia Computer Science*, vol. 132, pp. 993–1000, 2018.
[17] Rani, M., & Mishra, S., "CNN-based Intrusion Detection in SDN Using Flow Image Representation," *Computer Communications*, vol. 147, pp. 180–187, 2019.
[18] Joshi, P., & Mehta, A., "Ensemble Machine Learning Techniques for Network Intrusion Detection," *IEEE Access*, vol. 8, pp. 53845–53859, 2020.
[19] Kulkarni, P., Sharma, A., & Bansal, M., "LSTM-based Detection of Slow and Low Attacks from System Logs," *Future Generation Computer Systems*, vol. 108, pp. 719–728, 2020.
[20] Verma, T., & Singh, D., "Hybrid CNN-LSTM Model for Network Intrusion Detection in IoT," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, pp. 8913–8928, 2021.
[21] Raj, S., & Dutta, R., "Lightweight Intrusion Detection Models for Fog Computing Environments," *Computer Networks*, vol. 173, p. 107208, 2020.
[22] Nair, R., & Kumar, S., "Explainable AI for Network Intrusion Detection Using SHAP," *IEEE Transactions on Network and Service Management*, vol. 18, no. 3, pp. 3213–3224, 2022.
[23] Chatterjee, S., & Sen, A., "Federated Learning for Privacy-Aware Intrusion Detection in IoT," *IEEE Internet of Things Journal*, vol. 10, no. 2, pp. 1225–1236, 2023.
[24] Rao, P., & Patil, R., "Self-Adaptive Firewalls Using Reinforcement Learning," *Journal of Information Security and Applications*, vol. 73, p. 103348, 2024.
[25] Iyer, S., & Srinivasan, K., "Deep Autoencoder-Based Anomaly Detection for Encrypted Traffic," *Neural Computing and Applications*, vol. 36, pp. 1379–1392, 2024.
[26] Banerjee, N., & Jadhav, K., "Unsupervised Zero-Day Intrusion Detection using One-Class SVM and Isolation Forest," *Computers & Security*, vol. 124, p. 102993, 2023.
[27] Khan, A., & Deshmukh, V., "Bidirectional LSTM Framework for Real-Time Intrusion Detection," *Expert Systems with Applications*, vol. 201, p. 117123, 2022.
[28] Thomas, S., & Ghosh, A., "Multi-Modal Intrusion Detection via CNN and NLP Log Fusion," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 228–239, 2024.