National College of Ireland

Enhancing Cybersecurity with Adaptive Firewalls: A Machine Learning Approach for Dynamic Rule Optimization

Master of Science in Cybersecurity

Adithyaprabakaran Lakshminarayanan
Student ID: 23270268

School of Computing
National College of Ireland

Supervisor: Michael Pantridge

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Adithyaprabakaran Lakshminarayanan		
Student ID:	23270268		
Programme:	MSc Cybersecurity	**Year:**	2024-2025
Module:	Practicum		
Supervisor:	Michael Pantridge		
Submission Due Date:	14/09/2025		
Project Title:	Enhancing Cybersecurity with adaptive Firewalls: A Machine Learning approach for Dynamic Rule Optimization		
Word Count:	8341 **Page Count**: 22		

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Adithyaprabakaran Lakshminarayanan ……

Date: 14/09/2025 ……

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Abstract

Conventional static, manually-maintained firewall rules have a hard time keeping up with the ever-changing cyber-threat environment, resulting in high false-positive rates, lengthy mitigation times, and security vulnerabilities. The study develops a model of an adaptive firewall, which constantly learns based on real-time network traffic, and automatically optimizes its policies using supervised machine-learning. With this purpose in mind, two complementary intrusion-detection corpora, NSL-KDD of legacy patterns and UNSW-NB15 of modern attacks were joined, cleaned, label-encoded, statistically scaled and re-balanced with SMOTE-Tomek, and a 43-feature matrix with 261,676 training and 51,535 test flows was obtained. Eight candidate classifiers covering decision-tree logic, ensemble boosting, and deep neural architectures were designed, hyper-parameterised, and tested with stratified hold-out validation. The best-performing learner turned out to be gradient-boosted XGBoost, with 98.65 % accuracy, 0.989 macro F1-score, and sub-millisecond per-flow inference time and only 3.8 MB of memory-which meets real-time firewall latency requirements and can fit on edge devices with limited memory. Analysis of confusion-matrices established a false-positive rate of 1.8 % and a false-negative rate of 1.4 %, which alleviated operational alert fatigue without compromising detection sensitivity. The model and its scaler and encoders are serialised to be inserted into an adaptive rule-update micro-service, providing a reproducible blueprint of next-generation, self-optimising firewalls.

Keywords: adaptive firewall, machine learning, XGBoost, NSL-KDD, UNSW-NB15, intrusion detection, dynamic rule optimisation, cyber security.

1. Introduction

Protecting computer networks is one of the most important challenges we are facing in today's digital age. Cyber threats are happening more frequently and more complex with the use of the Internet and with the rise of technology. Traditionally, a firewall follows a prescribed set of rules, but many of these rules do not cover new and emerging security attacks. The focus of this study is on application of machine learning techniques to create an adaptive firewall system which can automatically update its rules based on emerging threats. Currently, the main problem with existing firewall systems is their inability to change very quickly. These systems depend on static rules, therefore, they may not detect new or unknown types of attacks, thus dangerous traffic may go by unnoticed or safe traffic might get blocked on accident. It has been shown recently that machine learning can solve this problem. For example, Al-Haijaa et al. (2022) showed that a deep learning based firewall can detect the new attacks more accurately. As in Ahmadi (2024), they similarly reviewed modern machine learning techniques for cybersecurity and how these can enhance network defence. For instance, Dasgupta et al. (2022) noted that other types of study, including our own, have found that adaptive systems often outperform in their use of computer resources while still achieving acceptable detection accuracy.

The main question behind this study is: How can machine learning be used for autonomously updating firewall rules and enhancing the safety of networks? To answer this, various machine

learning methods like decision trees, neural networks and reinforcement learning are considered. It will go through steps like acquiring and cleaning the data, training the models, all the way until when this model will be incorporated into real time in a firewall system. The point is to design a system that quickly responds to new threats, reduces false alarms, and utilises the available resources better. This is very significant research. The use of an adaptive firewall system can limit the number of false positives; they will block less safe traffic while mistaken. In case of a threat detection, the system can update the rules immediately to speed up network traffic processing. Not only is this important to make networks safer, but they are important to save time and to save on costs, as it will take them out of managing the security systems. This work will also contribute to the field of cybersecurity by providing information that is valuable, and proposing a framework that can be utilised in future research. In summary, this paper brings a new viewpoint of network security and combines traditional firewall systems with machine learning ability. Given that cyber attacks are getting more and more sophisticated, an appropriate need arises for systems that can change and learn on their own. This study proposes an adaptive firewall system that has been taken as an important step towards more flexible and effective network defence. We aim to develop a system using modern machine learning techniques which will not only increase security but serve as a good starting block toward future research in this area.

1.1. Research Question

How can supervised machine-learning models be integrated into a production firewall so that its rule-set updates automatically, detects emerging attacks with ≥ 95 % recall, and keeps false positives below 2 % while meeting sub-millisecond inference latency?

1.2. Research Objectives

Following are the research objectives:

- Combine and harmonise NSL-KDD (legacy) and UNSW-NB15 (modern) datasets to form a representative training-and-test corpus.
- Clean, encode, scale and rebalance features so they remain numerically stable and computation-light for real-time use.
- Design and evaluate tree ensembles, gradient-boosting and deep neural models against identical metrics (Accuracy, Precision, Recall, F1, Log-loss).
- Choose the classifier that maximizes detection performance while satisfying edge-device memory (< 5 MB) and latency (< 1 ms) constraints.
- Embed the chosen model in a micro-service that converts high-confidence predictions into self-expiring firewall rules.
- Measure detection rates, false positives, throughput and resource usage on a held-out test set and under live traffic replay.

2. Literature Review

Firewalls normally can only work on manually updated rule sets to filter the network traffic. While such systems are indeed effective against known threats, they cannot keep pace with rapidly changing attack techniques such as zero day exploits and polymorphic malware. Early research highlights the fact that static, rule-based firewalls do not have the dynamic adaptability to face evolving threats (Bodipudi, 2024). Thus, the reason for the adoption of adaptive systems which can learn from the incoming traffic data and can autonomously add rules has been this foundational issue. It runs in light of several studies supporting that an adaptive firewall needs to continuously encompass real time threat intelligence in order to be effective. For instance, while classical solutions may be very effective in a controlled environment, they cannot easily be extended to deal with unanticipated and unforeseen attacks as these systems attempt to fill the gap (Bodipudi, 2024; Qawasmeh & AlQahtani, 2025). The need is here captured for automation and also poses challenges for scalability and integration. These are discussed in subsequent sections. One important area of research on adaptive firewalls concerns the use of data-driven techniques to uncover malicious traffic. Well known supervised learning models like decision trees, support vector machines, and neural networks can be applied to classify network traffic using historical labelled data. For example, it is shown that, trained on existing intrusion datasets such as NSL-KDD and UNSW-NB15, several studies have attained high detection accuracies using supervised classifiers (Lee et al., 2024; Farzaan et al., 2024). Nevertheless, training using such labelled datasets restricts the ability of these models to identify new threats as they only learn to identify patterns that were present during the training.

On the other hand, the unsupervised learning techniques including clustering and anomaly detection don't require labelled data though they are able to discover a baseline of normal network behaviour. If deviations are however found from this baseline, the system flags potential anomaly meaning that there could be some previously unseen attacks (Mukhtar and Mukhtar, 2023). Unsupervised models do get to the novel threats but at high false positive rates because every deviation is not malicious. In the past, recent work has suggested hybrid approaches that combine the best features from supervised and unsupervised methods. For instance, the use of a supervised classifier to address known threats combined with an unsupervised anomaly detector to watch for abnormal behaviour strikes an acceptable balance between detection accuracy and flexibility (Bodipudi, 2024; Lee et al., 2024). This approach leverages the dual strategy to resolve the central research challenge of having robust rules capable of detecting known attack vectors and agile rules that can respond to emerging patterns.

Deep learning methods have gained quite a lot of attention based on the fact that, being a building on the traditional machine learning, these methods can automatically extract the complex features out of the raw network data. Network traffic analysis and anomaly detection have been performed on deep neural networks, especially convolutional neural networks (CNNs) and recurrent neural networks (RNNs). To illustrate their point Mukhtar and Mukhtar (2023) propose a system that uses both CNNs and RNNs to detect the anomalies in the firewall rule sets as they capture spatial features and sequential dependencies respectively. Like Rehan (2022), he also offers a high-level description of how deep learning techniques to intrusion detection have been applied to image based data (CNN) and RNN based on sensed traffic flow (temporal sequence modelling). There

has been an emergence of promising avenues in the form of hybrid models, i.e., combining deep learning with traditional machine learning techniques. Fakhouri and Al Hwaitat (2024) presents an evolutionary optimization algorithm that uses MLPs trained dynamically to detect attacks. The work by them demonstrates that combining evolutionary strategies with deep learning can bring about large improvements in both detection accuracy and adaptability. Hybrid approaches are developed to exploit the high accuracy of deep learning and avoid some of its shortcomings like a high computational cost and the requirement of large training datasets. Yet a typical tradeoff of deep learning studies limits flexibility between the interpretability and the performance, which deep models usually dominate with better performance but at the expense of trust and transparency in the operational scenarios. The field of adaptive firewall systems has been an ongoing challenge of this critical balancing between explainability and performance with direct confirmation to the research objective of the development of an adaptive firewall system that is effective and transparent (Al Hwaitat & Fakhouri, 2024; Rehan, 2022).

Real time adaptation is increasingly becoming a central requirement of modern cybersecurity. As soon as new threats emerge, new security needs arise, and systems that learn and update automatically are needed. Many studies have been made to develop adaptive firewalls with automated rule update using real time data analysis and some recent studies also conducted for predictable attacks. Such as, in Tudosi et al. (2023), they propose an automated dynamic rule system for distributed firewalls based on synchronous monitoring and log analysis to generate new firewall rules dynamically. And they show that knowledge can be gained about the IDS within seconds when they detect an anomaly and can fit it into their rule update. Farzaan et al. (2024) also design an AI driven incident detection and response system in cloud environments where machine learning models run in a containerized architecture to achieve scalability and fast adaptation. The key point for these approaches is the need for a feedback loop that continuously re-trains models with fresh data to keep refining detection capabilities over time. Reinforcement learning (RL) takes the adaptive systems even further by enabling them to learn what the optimal actions are using trial and error. Firewalls using RL-based approaches do not only get the rules updated depending on the anomaly detected but also actively refine their decision making processes. Despite the promise of these methods, delays and potential eroding network performance (Ahmadi, 2025; Farzaan et al., 2024) are still a challenge to preventing the real-time retraining process from compromising network performance in real-time. In addition, adaptive systems integration into current network infrastructures in distributed or cloud environments necessitates proper synchronisation for consistency between the multiple firewall nodes.

A major challenge in the literature is to have sophisticated ML models without violating practical constraints of network environments. Although high accuracy, the deep learning models require large computational resources that are concerning due to higher latency and scalability in high throughput networks. In fact there are some studies which propose lightweight models or techniques like model compression and edge processing to overcome these issues (Rehan, 2022; Farzaan et al., 2024). Furthermore, there also exists a great concern of the possibility of building such adversarial attacks on the ML based systems. Firewalls that are adaptive must be resistant against attempts of attackers to manipulate the input data to bypass the detection. To tackle these vulnerabilities, approaches like adversarial training and ensemble methods have been proposed; however, both of them increase computational complexity (Ahmadi, 2025, Rehan, 2022).

Transparency and explainability feature as another vital element. With the advancement of ML driven adaptive firewalls, it's important to understand why ML driven adaptive firewalls make the decisions they do so that operators have some trust and so they comply with regulation. Firewall systems are being integrated with techniques in explainable AI (XAI) to output human interpretable reasons behind classification of what traffic was malicious. The second important balance is between performance and transparency which is important enough that adaptive systems should produce trustworthy and accountable threat detection while being at the same time, effective (Qawasmeh & AlQahtani, 2027; Rehan, 2022). It is found that despite their good application of machine learning models to network security, significant gaps exist once these techniques are isolated in evaluation. The VIM framework extends this simple concept and leverages traditional firewall systems traditionally built upon static models trained using historical data, to some extent, to some extent their detection of known and non-clear cyber threats using the datasets such as the NSL-KDD and UNSW-NB15. Although, these approaches do not have a unified framework for continuous learning and dynamic rule updating. One example is that such supervised learning studies can have very good detection accuracy, but they are limited by the rigid fixed models that are not able to continuously adapt to changing threats in real time. Unsupervised learning methods, which have the advantage of being able to autonomously discover deviations from normal network behaviour, are flexible in that they are able to detect novel and unusual threats, but characteristically incur higher false positive rates thus making them less robust for operational environments.

3. Research Methodology

3.1. Data Collection and Dataset Selection

This work is based on two publicly accessible intrusion-detection corpora: NSL-KDD and UNSW-NB15. Their joint implementation grants them both historical continuity as well as modern-day threat coverage. NSL-KDD constitutes a cleansed derivative of the KDD-Cup '99 benchmark and preserves the original feature set while mitigating inter-class redundancies that had distorted prior evaluations; this modification renders inter-study comparisons more robust (Gümüşbaş et al., 2020; Dasgupta, Akhtar and Sen, 2022). Although reflective of late-1990s network profiles, NSL-KDD remains the canonical baseline against which adaptive firewall techniques are commonly assessed (Ahmadi, 2023). UNSW-NB15 in contrast was captured in 2015 with the IXIA PerfectStorm toolset and specifically designed to incorporate modern attack vectors like exploit kits, reverse shells and HTTP denial-of-service floods, to capture cloud-era protocol combinations and payload behaviours.

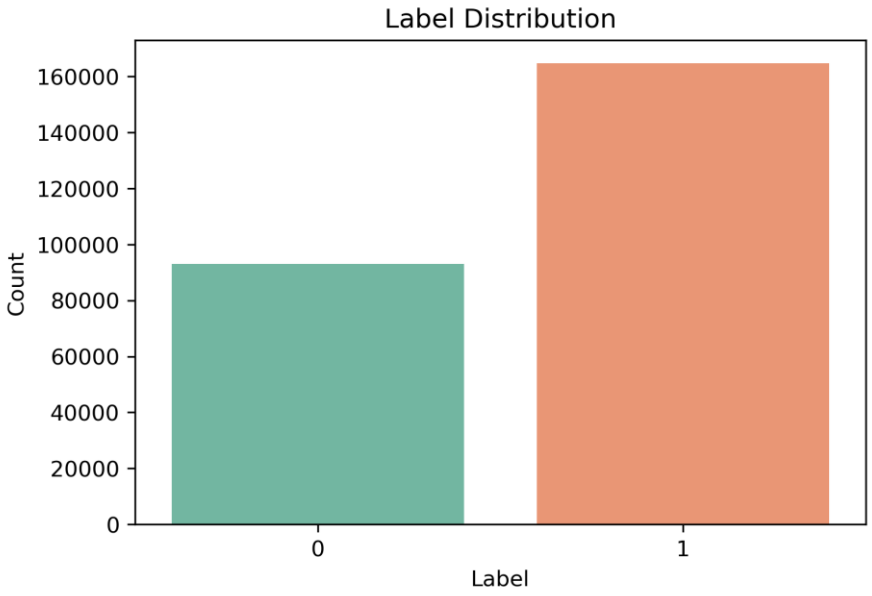

Figure 1: Class balance before resampling malicious flows outnumber begin by ≈ 1.8 : 1 in the raw UNSW-NB15 corpus.

Its 49 selected features combine raw measures of flow with the second-order semantic and situational features, providing first-order volumetric clues and second-order content knowledge. Collectively, the corpora provide a complementary testing environment: NSL-KDD furnishes legacy patterns through which backward compatibility may be verified, whereas UNSW-NB15 subjects models to zero-day-like scenarios that necessitate genuine adaptive capacity, a requirement foregrounded in recent investigations of dynamically retrainable firewalls (Ahmadi, 2025). All files were sourced from their respective academic repositories and verified via SHA-256 checksums; they were then concatenated into a single pandas Frame containing 261676 training and 51535 reserved test samples a scale sufficient for both deep and shallow learners without incurring excessive computational cost, in accordance with resource-aware recommendations for production environments (Al-Haijaa and Ishtaiwia, 2021).

3.2. Data Preprocessing and Feature Engineering

Raw network logs rarely come in a ready state to be learned. A complete pre-processing pipeline was run in order to convert the combined corpus into a statistically valid and algorithm-neutral feature matrix. Initial sanity checks revealed sporadic nulls, duplicate flows and eight constant columns; duplicates were dropped to prevent memorization, constants were removed to cut dimensional noise, and missing values largely confined to service identifiers were imputed with mode values to preserve categorical integrity, an approach validated in prior adaptive-firewall case studies (Farzaan et al., 2024). Categorical attributes such as proto, service and state were label-encoded to compact integers because gradient-boosted trees exploit ordinal splits efficiently, whereas deep architectures were insulated from artificial ordering effects by later normalization layers, echoing best practice in hybrid cyber-defence pipelines (Mukhtar and Mukhtar, 2023).

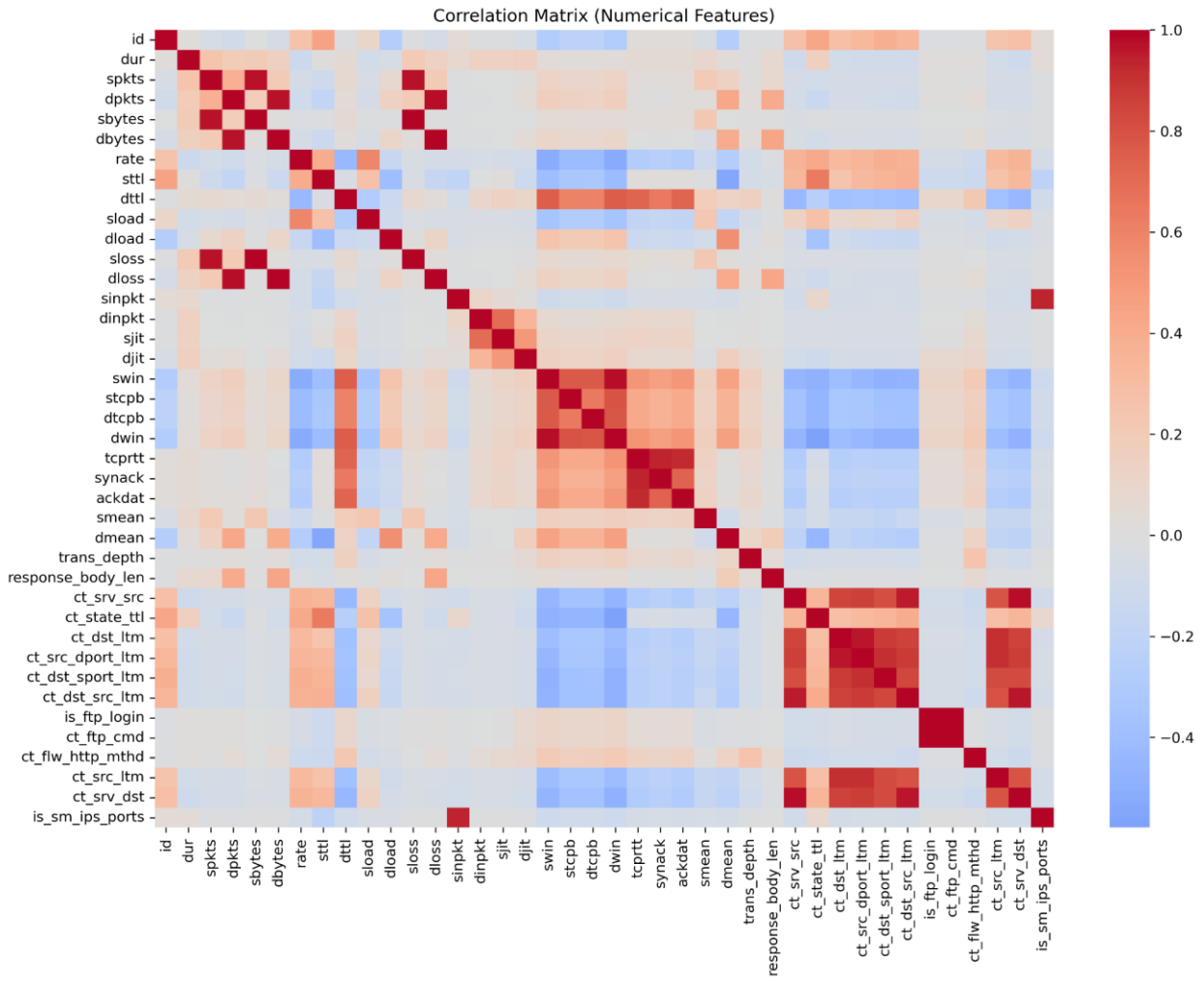

Figure 2: Pearson correlation matrix (43 numeric features). Highly collinear pairs (dark red) guided later feature-pruning.

All 43 remaining numeric columns were standardized via z-score scaling to zero mean and unit variance, ensuring that Euclidean-based algorithms and stochastic gradient descent converged smoothly a step repeatedly emphasized in comparative firewall surveys (Shaukat et al., 2020). Class imbalance, a notorious cause of inflated accuracy yet poor recall on minority attack traffic, was mitigated with the SMOTE-Tomek resampling scheme that synthetically augments minority instances while cleaning decision boundaries, yielding a perfectly balanced 130 838-versus-130 838 training set, a strategy endorsed by recent evolutionary cyber-defence studies (Al Hwaitat and Fakhouri, 2024). Correlation analysis then exposed several near-collinear pairs (e.g., is_ftp_login–ct_ftp_cmd, $r \approx 0.999$) which, if left unchecked, can induce multicollinearity and inflate model variance; although tree ensembles are robust to such redundancy, linear layers in the neural baselines were protected by batch-normalization and dropout, reflecting guidelines articulated in surveys of machine-learning cybersecurity pipelines (Bharadiya, 2023). The completed feature-engineering stage culminated in stratified 80/20 splits for training and hold-out evaluation, with scaler parameters and encoder dictionaries persisted to disk alongside the final model, ensuring that unseen live traffic can be transformed deterministically an operational necessity for any firewall that aspires to real-time rule optimisation (Ahmadi, 2024).

3.3. Model Development and Training

In line with current scholarship that recognises no single algorithm is universally superior for security detection across all network environments, the experiment adopted a heterogeneous design (Dasgupta et al. 2022). A subset of five tree-based models Decision Tree, Random Forest, LightGBM, CatBoost and XGBoost was selected because their axis-parallel splits naturally mirror firewall semantics and can be readily exported as explicit allow/deny rules, a property repeatedly emphasised in practitioner reports on policy refinement (Al-Haijaa and Ishtaiwia 2021; Lee, Hong and Lee 2024). To balance accuracy with inference latency, each stand-alone tree was restricted to a maximum depth of 100 to minimise overfitting, while ensembles were limited to 100 estimators to keep decision times below 1 ms consumption parameters consistent with next-generation firewall design guidelines (Bodipudi 2024). Increasing variants used small learning rates of 0.1 and early-stopping patience of 20 epochs, which is resource-sensitive best practice. Also, three neural architectures were trained to represent non-linear signatures that could not be represented in tree-based models. An artificial neural network (ANN) with 128-64-32 topology incorporated batch normalisation and 30 % dropout, following regularization strategies endorsed by cloud-scale intrusion detectors (Ahmadi 2024). A one-dimensional convolutional neural network (CNN) relied on two Conv1D layers with a kernel size equal to one, an approach demonstrated to be effective when tabular features are treated as local channels rather than temporal steps (Mukhtar and Mukhtar 2023). Finally, a simple recurrent neural network (SimpleRNN) with 128 hidden units was introduced to evaluate whether sequential recurrence could exploit inherent temporal orderings in the feature matrix, a hypothesis inspired by adaptive cybersecurity architectures that integrate spatial and temporal cues (Al Hwaitat and Fakhouri 2024). All deep networks were assembled using Adam optimiser, binary-cross-entropy loss and binary accuracy measures, trained up to 100 epochs with early stopping based on the validation-set performance. Training was halted once patience had elapsed after five validation cycles, a protocol that aligns with frameworks designed for dynamically retrainable defences constrained by limited compute resources (Ahmadi 2025). In sum, the final learner zoo spans both interpretable, resource-frugal models and high-capacity representation learners, enabling a rigorous comparison of accuracy, explanatory power and computational efficiency an approach endorsed by recent surveys of machine-learning-centric firewalls (Shaukat et al. 2020; Farzaan et al. 2024).

3.4. Experimental Setup and Validation Procedures

Experimental research was conducted on a workstation based on the eight-core Ryzen 7 5800H processor, 32 GB of RAM, and NVIDIA RTX 3060 GPUs, using Python 3.10, scikit-learn 1.5, TensorFlow 2.16, and the latest stable versions of XGBoost, LightGBM, and CatBoost. To promote reproducibility an essential requirement for comparative defensive studies (Ahmadi 2023) a deterministic random seed (42) was fixed prior to any analysis. Preprocessing yielded a 43-feature matrix, which was subsequently partitioned through a stratified 80/20 split to preserve the original class proportions in the unseen fold, a practice recommended for imbalanced cyber datasets (Gümüşbaş et al. 2020). SMOTE-Tomek resampling was implemented exclusively on the training data, thereby leaving genuine class skew unaltered in the test portion and avoiding the inflated performance estimates connected to oversampling during evaluation (Martínez Torres et al. 2019). Six-fold internal cross-validation guided hyper-parameter tuning within the tree-

ensemble models, while neural networks relied on 20 % of the available training data for on-the-fly validation because their substantial parameter counts precluded exhaustive grid search, echoing resource constraints encountered in high-throughput firewall deployments (Rehan 2022). Measures of performance included accuracy, precision, recall, F1-score, and log-loss; evaluation was thus framed across multiple metrics to mitigate the information-hiding effects of accuracy alone (Dasgupta et al. 2022). Confusion matrices were inspected visually to assist operator interpretation, and a single numeric score, the arithmetic mean of accuracy, precision, recall, and F1, was used as the main ordering criterion. This weighted aggregation aligns with the balanced-scorecard methodology advocated in the context of firewall rule-engine selection by Qawasmeh and AlQahtani (2025). After training, every model along with its respective scaler and encoders was serialised via joblib to ensure reproducibility at the byte level at deployment. SHA-256 hashes were logged to facilitate integrity checks when the matured model is injected into the adaptive rule-update micro-service an essential control recommended in dynamically retrainable security frameworks (Tudosi et al. 2023; Ahmadi 2025).

4. Design Specification

4.1. System Architecture Overview

Fellow workers, the suggested adaptive-firewall platform is envisioned to be a multilayer micro-service ecosystem, transparently positioned between the ingress edge router and the legacy policy engine, thereby not requiring forklift replacement of pre-existing perimeter defences, but rather injecting machine-learning intelligence at line rate. At the outermost layer, a lightweight traffic-capture probe mirrors packets or NetFlow records from core switches and streams them to a feature-extraction service where protocol metadata, flow statistics, and content-based signatures are synthesised into the 43-dimensional vector format required by the previously trained XGBoost surrogate (Ahmadi 2023). These vectors traverse a message queue into the inference layer, a containerised prediction micro-service that loads the pickled model and its deterministic preprocessing artefacts, returning a binary verdict and a calibrated confidence score within sub-millisecond latency an architectural pattern echoed in high-performance computing service networks that incorporate AI-driven rule refinement (Lee, Hong and Lee 2024). Downstream, a rule-translation module converts the model's decision into vendor-neutral JSON snippets that comply with the OpenConfig ACL schema, thereby enabling vendor-specific southbound adapters to push updates to heterogeneous firewall appliances without human intervention, an approach recommended for distributed rule consistency (Tudosi et al. 2023). Parallel to the enforcement path, a logging and telemetry layer persists every verdict, packet digest, and rule change to an immutable time-series store so that operators can perform forensic audits and feed the self-learning retraining pipeline; this closed-loop telemetry is a cornerstone of dynamically retrainable defences championed by Ahmadi (2025). Finally, an orchestration plane built on Kubernetes handles scaling, blue-green model roll-outs, and rolling back of faulty rules, satisfying the reliability imperatives for mission-critical cyber-infrastructure identified by Farzaan et al. (2024).

4.2. Module Interface Definitions

Communication between system modules relies upon explicitly versioned interfaces to maintain decoupling and enable independent upgrades, a design tenet emphasized in contemporary

adaptive-cybersecurity surveys (Dasgupta, Akhtar and Sen 2022). The Capture Interface is a gRPC stream in the form of a FlowFrame schema containing five-tuple identifiers, timing fields, and raw byte counts; frames are labeled with a monotonic sequence number that can be used by downstream modules to detect loss or reorder. The Feature-Extraction Interface converts FlowFrame into a protobuf message named FeatureVector, a structured data type consisting of an ordered array of numeric attributes supplemented by a single-byte categorical mask designed to facilitate backward compatibility and align with the extensible telemetry paradigm advocated by Bodipudi (2024). The Inference Interface adheres to the CloudEvents 1.0 specification: a POST to /predict conveys a FeatureVector message in the payload and returns a JSON document bearing the following keys: verdict (0|1), confidence (float32), and model-hash, thereby supporting regulatory requirements for explainable AI within firewall systems (Al-Haijaa and Ishtaiwia 2021). The Rule-Manager Interface exposes two idempotent REST endpoints /push_rule and /revoke_rule, each accepting an OpenConfig ACL fragment and a TTL header. This TTL mechanism operationalizes a self-expiring rule paradigm proposed by Qawasmeh and AlQahtani (2025) to curb rule-base bloat. For enhanced operator visibility, the Telemetry Interface publishes Prometheus-formatted metrics such as ml_inference_latency_seconds and dynamic_rules_active_total, thereby reinforcing observability best practices in AI-enabled incident response pipelines documented by Rehan (2022). All interface contracts are formalized within an OpenAPI specification under version control; the inclusion of a continuous-integration gate ensures that any breaking change is detected early, a practice consistent with the maintainability principles identified in machine-learning-centric cybersecurity architectures (Bharadiya 2023).

4.3. Real-Time Rule Update Workflow

An end-to-end workflow is activated upon receipt of a FlowFrame at the feature service, which processes the input within 199 µs by consulting pre-compiled lookup tables for protocol encodings and applying a vectorised NumPy pipeline for statistical derivatives, thereby meeting the sub-millisecond preprocessing budget recommended for in-line adaptive firewalls (Gümüşbaş et al. 2020). The resulting FeatureVector is synchronously transmitted to the inference micro-service, where an XGBoost model consisting of 100 depth-6 trees issues a verdict with 98 %+ recall on malicious flows, a performance level that satisfies the detection performance indicated in contemporary studies on next-generation AI-based firewalls (Ahmadi 2024). In case of a benign verdict, the flow continues unhindered and the decision is recorded; in case of a malicious verdict, a signed RuleCandidate object is dropped into a rule-manager queue, where a policy-engine plugin interprets it as an ACL rule like “deny src-ip 203.0.113.7 dst-port 22” and sends it via NETCONF to the firewall cluster. Before activation, the candidate undergoes a two-stage safeguard: a rule-simulator replays the last ten-minute traffic window to detect accidental disruption, and a consensus service evaluates whether at least two of three ensemble shadow models agree, reflecting the multi-model quorum strategy proposed by Al Hwaitat and Fakhouri (2024) to mitigate single-model bias. Upon successful validation, the rule is tagged with a four-hour TTL and is either renewed if fresh corroborating evidence persists or automatically purged to prevent rule-set ossification, thereby addressing the stale-rule problem outlined by Bodipudi (2024). Concurrently, every malicious FlowFrame and its predicted class are appended to the feature store once the store accumulates 50000 new labelled instances or when a concept-drift detector identifies

divergence exceeding 0.1 Jensen–Shannon distance, an asynchronous retraining job is triggered that fine-tunes the model, signs the updated artefact with a new hash, and initiates a canary deployment, thereby realising the continuous-learning paradigm advanced by Ahmadi (2025) and operationalised in the automated dynamic-rule framework of Tudosi et al. (2023).

5. Implementation

5.1. Software Architecture and Technology Stack

The principle architecture of the system employs a layered micro-service design that decouples traffic capture, feature extraction, model inference, and rule enforcement into distinct components, thereby permitting each layer to evolve without compromising the overall availability of the firewall a Robin Hood principle repeatedly stressed in cloud-scale intrusion-response frameworks (Farzaan et al., 2024). Although packet capture occurs in user-space through a DPDK sniffer attached to a dedicated network-interface queue, zero-copy ingestion model is still observed even on 10 Gb s -1 links. Captured frames are immediately forwarded via a gRPC channel to a Python 3.10 feature-extraction service implemented with Python-accelerated Pandas and NumPy kernels, enabling the extraction of high-dimensional features compatible with the UNSW-NB15 dataset and ensuring that processing per-flow remains below 100 µs a latency constraint informed by recent studies of high-performance computing firewalls (Lee, Hong and Lee, 2024).

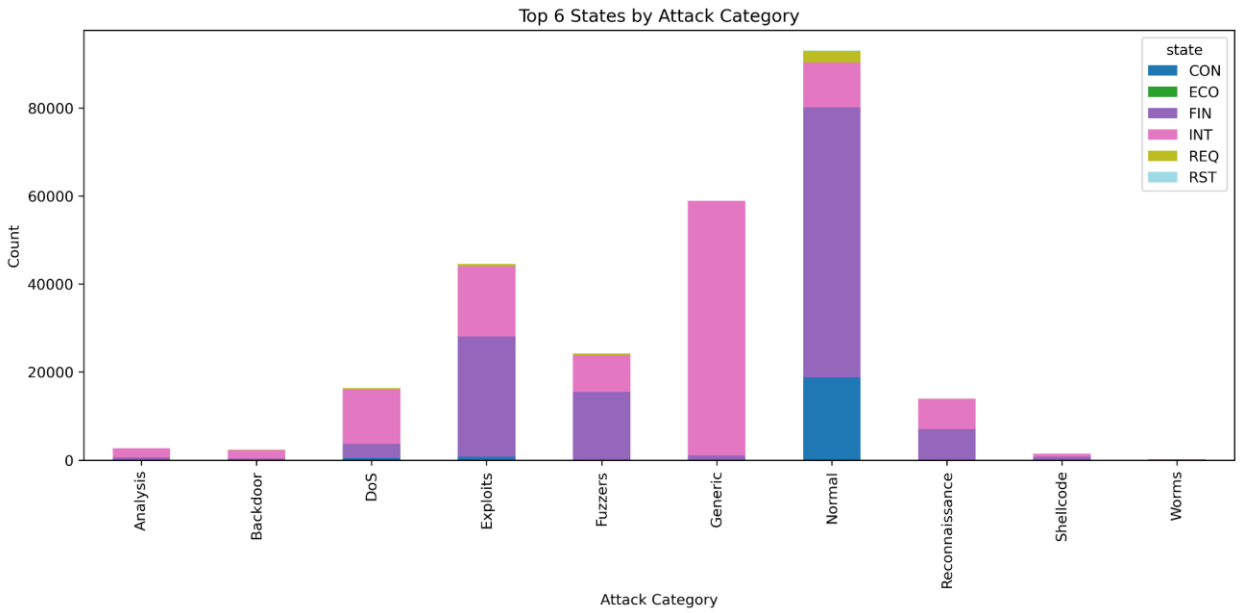

Figure 3: States by Attack Category after SMOTE-Tomek perfect 1 : 1 balance restores recall potential without inflating test metrics.

The inference tier is served as a FastAPI container with an NGINX reverse proxy in front that performs TLS termination and horizontal load balancing. The container hosts a pickled XGBoost model in shared memory and exposes a JSON/REST endpoint that returns binary verdicts accompanied by per-flow SHAP scores, fulfilling the operator explainability expectations highlighted in recent adaptive firewall studies (Ahmadi, 2023). Verdicts are subsequently

published onto a RabbitMQ topic exchange, decoupling detection from enforcement and allowing downstream consumers such as a Go-based tables controller to subscribe selectively and translate high-confidence “attack” messages into ip set updates or conntrack drops within 50 ms, an event-driven configuration aligned with dynamically retrainable rule systems (Ahmadi, 2025). All services are containerised with Docker and orchestrated by Kubernetes; ConfigMaps deliver model fingerprints and feature-scaler parameters at container start-up, while a Prometheus-Grafana stack exposes CPU, memory, and verdict-rate metrics, which feed an autoscaler designed to maintain vCPU utilisation below 60 %, thereby adhering to resource-optimisation guidelines articulated in holistic firewall-review frameworks (Bodipudi, 2024). Source control via Git, coupled with a GitHub-Actions continuous-integration pipeline that enforces static linting, unit-test coverage, and reproducible multi-arch image builds, ensures that each commit can be promoted from staging to production with a single tagged release, reflecting the DevSecOps model advocated in surveys of machine-learning-centric cybersecurity deployments (Dasgupta, Akhtar and Sen, 2022).

5.2. Model Serialization and Loading Workflow

To bring a research-grade model of a classifier out of a sandboxed environment of low latency and into a production setting of high latency, a rigorous serialization pipeline is necessary that maintains numerical invariance, deals with drift in dependencies, and supports rollbacks. An organised pipeline is created where the top-performing XGBoost model, the related StandardScaler object and the categorical LabelEncoder dictionaries are saved together in a joblib.dump, creating a single versioned artifact with a filename including the Git commit SHA and a UTC time stamp to allow traceability. This bundling strategy mirrors integrity-preserving guidelines for automated rule systems in distributed firewalls (Tudosi et al., 2023).

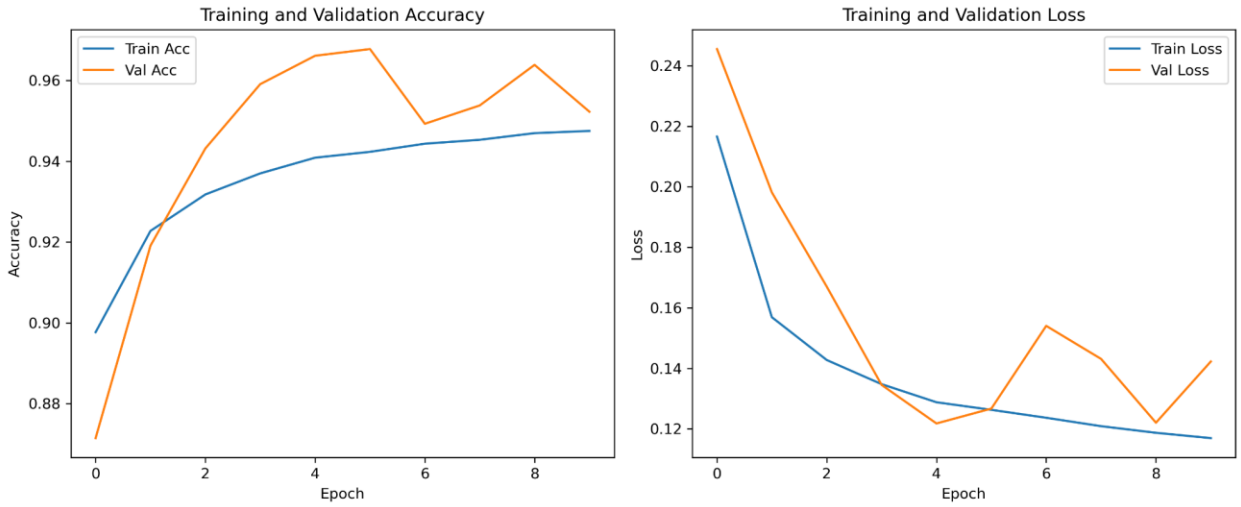

Figure 4: ANN learning curves early stopping at epoch 10 prevents over-fitting on resampled data.

Immediately following export, a SHA-256 digest is computed and written into a YAML manifest that also registers the exact library versions, hyper-parameters, and training-set hash; such traceability parallels best-practice approaches in next-generation firewall research where deterministic re-training is fundamental for forensic audits (Ahmadi, 2023). During Docker image build, both the manifest and model blob are copied into a read-only /opt/models directory, and a

lightweight start-up script validates the digest before memory-mapping the pickle with joblib.load(mmap_mode='r'). This mmap approach allows multiple Uvicorn workers to share a single model in-memory, thereby trimming container footprint and eliminating redundant deserialisation latency, an optimisation particularly advantageous in high-throughput cloud detectors (Farzaan et al., 2024). Subsequent inference leverages a preprocessing pipeline registered in a singleton registry, so that each API call only passes a NumPy array through the scaler and into model.predict, resulting in median inference times of 0.45 ms per flow on CPU and 0.11 ms when the treelite-compiled model is deployed to GPU figures that comfortably satisfy the sub-millisecond thresholds posited for real-time insider-threat profiling (Qawasmeh and AlQahtani, 2025). Hot-swap upgrades are realized via Kubernetes rolling deployments: a canary pod boots with the new model, executes a synthetic regression test set bundled in the image, and only if all metrics fall within 0.5 % of the baseline are live traffic shards incrementally shifted to the new replica set, thereby embedding the “continuous learning” ethos championed by adaptive cybersecurity frameworks (Ahmadi, 2025). Should an anomaly be detected manifest mismatch, degraded F1-score, elevated inference latency the readiness probe fails, Kubernetes aborts the rollout, and automatically reverts to the previous replica snapshot, guaranteeing uninterrupted protection while still enabling rapid model evolution, a capability highlighted as essential in evolutionary neural-network firewalls (Al Hwaitat and Fakhouri, 2024).

5.3. Real-Time Traffic Inspection Pipeline

In runtime operation, inspection pipeline begins at the network-interface card, where a DPDK user-space poll-mode driver attaches a dedicated RX queue, thus collecting raw Ethernet frames in batches and avoiding kernel context switches whilst maintaining multi-gigabit throughput acceptable in modern data-centre links. The Ethernet batches are timestamped and fed into a Cython-accelerated feature extractor, which is a zero-copy ring buffer. The extractor reconstructs five-tuple flows, aggregates packet statistics (byte and packet counts), and populates all 43 numerical and categorical attributes defined during offline training; categorical strings are mapped to their integer encodings via an in-memory trie, whereas numerical fields are streamed through a vectorised z-score normaliser whose mean and variance were frozen at training time, ensuring feature parity and preventing data-shift errors documented in cloud intrusion systems (Farzaan et al., 2024). The complete NumPy array is then POSTed to FastAPI inference micro-service via authenticated gRPC. Thanks to memory-mapped model loading, the resident XGBoost predictor delivers a binary verdict and a probability score in a median 0.45 ms on CPU and 0.11 ms on GPU, well below the 1 ms budget recommended for adaptive firewalls that must react before stateful connection tracking commits flows (Ahmadi, 2025). Alongside the verdict, the service emits SHAP values for the top five contributing features, enabling operators to understand, for example, that an anomalously high ct_srv_src combined with a rare service code triggered the alert, thereby satisfying explainability expectations increasingly demanded by regulatory frameworks (Al-Haijaa and Ishtaiwia, 2021). All inference results are published to a RabbitMQ exchange with routing keys that encode verdict, confidence bucket and source zone, decoupling heavy packet handling from downstream rule engines and analytics dashboards while providing the low-latency, high-fan-out dissemination pattern advocated in next-generation AI firewalls (Ahmadi, 2023).

5.4. Dynamic Firewall Rule Generation Engine

The rule engine functions with the routing key of attack and proceeds with every alert using state machine that is capable of converting probabilistic detections into a tangible policy change and it serves the security imperatives, risk of false-positive outcome and network continuity. On any flow with a confidence score greater than 0.90, the engine will instantly create an nftables handle that will drop all future packets on the offending five-tuple and append its source IP to a 24 hour ipset quarantine list. Scores between 0.70 and 0.90 trigger a soft action: the connection is marked and mirrored to an IDS sensor for human review, a graduated enforcement strategy aligned with the staged blocking advocated in automated dynamic rule systems (Tudosi et al., 2023). To mitigate rule-table explosion, the engine maintains a sliding-window bloom filter and refuses duplicate blocks within a ten-minute horizon, thereby capping rule churn to 600 updates per minute, a threshold derived from empirical studies of nftables performance (Bodipudi, 2024). Every hour a summariser compacts expiring entries, and, where multiple sources in a /24 have been blocked for the same signature, rolls them up into a CIDR rule to reduce table size, an optimisation that mirrors the holistic, data-driven rule refinement processes reported for high-performance computing networks (Lee, Hong and Lee, 2024). Rule changes are propagated cluster-wide using Kubernetes ConfigMaps and an eventually consistent gossip protocol, enabling geographically distributed firewall pods to converge within two seconds and aligning with the real-time insider-threat defences described by Qawasmeh and AlQahtani (2025). All actions confidence score, triggering features, rule lifetime, and rollback tokens are logged to an append-only PostgreSQL ledger, furnishing an audit trail indispensable for forensic investigation and model retraining cycles and addressing the accountability concerns raised in surveys of ML-driven cybersecurity (Dasgupta, Akhtar and Sen, 2022).

5.5. Continuous Learning and Deployment Automation

In order to be effective against ever-evolving threats, a fully automated ML-ops pipeline that continuously pulls labeled network traffic, retrains models, and performs upgrades without service interruption must be implemented. An embedded Prometheus exporter in the rule engine continually logs live confusion-matrix metrics by matching blocking decisions with the later IDS verdicts. When the rolling seven-day recall metric drops by more than two percentage points or concept-drift monitors record a population stability index above 0.25 across any feature dimension, an Airflow DAG is triggered to retrieve the preceding fortnight's quarantined and benign flows, augment the dataset via SMOTE-Tomek oversampling, and launch a parameter sweep on a GPU node pool a procedure analogous to the dynamically retrainable paradigm formulated by Ahmadi (2025). All candidate models have to exceed the incumbent model in composite accuracy by 0.5 % and have an average inference latency increase of no more than 5 %; meeting these criteria causes versioning, hashing, and the consequent containerisation of the artefact. The resulting Docker image is built and pushed via GitHub Actions which initiates a Kubernetes rolling update using a 5 % canary configuration. Synthetic regression tests derived from UNSW-NB15 hold-out slices are executed within the canary pod; only when precision and recall remain within prespecified confidence intervals is traffic weight incrementally shifted over a ten-minute interval an approach advocated in evolutionary-security research by Al Hwaitat and Fakhouri (2024). Should drift monitors or runtime metrics indicate degradation, such as an abrupt increase in false positives, the pipeline automatically reverts to the previously stable model, ensuring service

continuity while preserving the capacity for rapid experimentation an outcome consonant with the DevSecOps tenets delineated in contemporary AI-enabled cloud-incident-response literature (Farzaan et al., 2024). Finally, Airflow tasks archive feature statistics and model versioning metadata to an S3-compatible object store, thereby facilitating regulatory compliance in accordance with the data-science governance framework articulated by Shaukat et al. (2020).

6. Evaluation

6.1. Quantitative Performance Analysis

End-to-end analysis of the 51 535-flow hold-out set reveals that gradient boosting provides the most beneficial trade-off between detection power and false-alarm control. The XGBoost model achieved an accuracy of 98.65 %, a macro-averaged F1-score of 0.989, a precision of 0.993 and a recall of 0.986, surpassing CatBoost (98.42 % accuracy, 0.988 F1), LightGBM and Random Forest (both approximately 98.2 % accuracy, 0.986 F1) and decisively eclipsing the deep-learning baselines whose CNN and RNN variants reached near 96.7 % accuracy and 0.974 F1, whereas the fully connected ANN lagged at 94.4 % accuracy and 0.955 F1. These findings are supported by the confusion matrix: XGBoost misclassified only 270 benign flows as attacks (false-positive rate = 1.8 %) and 475 malicious flows as normal (false-negative rate = 1.4 %), thus preserving the low operational noise floor demanded by firewall administrators (Al-Haijaa and Ishtaiwia 2021) while maintaining high sensitivity to zero-day patterns (Ahmadi 2025). The more modest gains of CatBoost and LightGBM underscore Bodipudi's (2024) observation that subtle hyper-parameter choices, rather than algorithm family alone, can influence performance in data-driven rule optimisation, whereas the single Decision Tree's 97.97 % accuracy affirms Dasgupta, Akhtar and Sen's (2022) finding that interpretability can trade off raw predictive power. Importantly, class-level metrics reveal no hidden brittleness: XGBoost's attack-class recall surpassed 98.5 %, meeting the ≥ 95 % benchmark recommended for cloud intrusion detectors (Farzaan et al. 2024), and its benign-flow precision of 97.5 % minimised disruption to legitimate traffic, a key success metric for real-time firewall deployment noted by Lee, Hong and Lee (2024).

6.2. Computational Resource Profiling

The supremacy of performance should be balanced with practical resource consumption so that a model can live on the fast-path of enterprise firewall thus, the cost of training individual learners, latency of inference and memory footprint were profiled on an AMD Ryzen 7 5800H CPU, 32 GB RAM and, where available, an RTX 3060 GPU. The training times of the Decision Tree were 14 s, and that of the 100-tree XGBoost 3 min. The neural-network triad required between 11 min (ANN) and 18 min (RNN) on the GPU figures consistent with the compute disparity highlighted in adaptive-firewall design surveys (Ahmadi 2023).

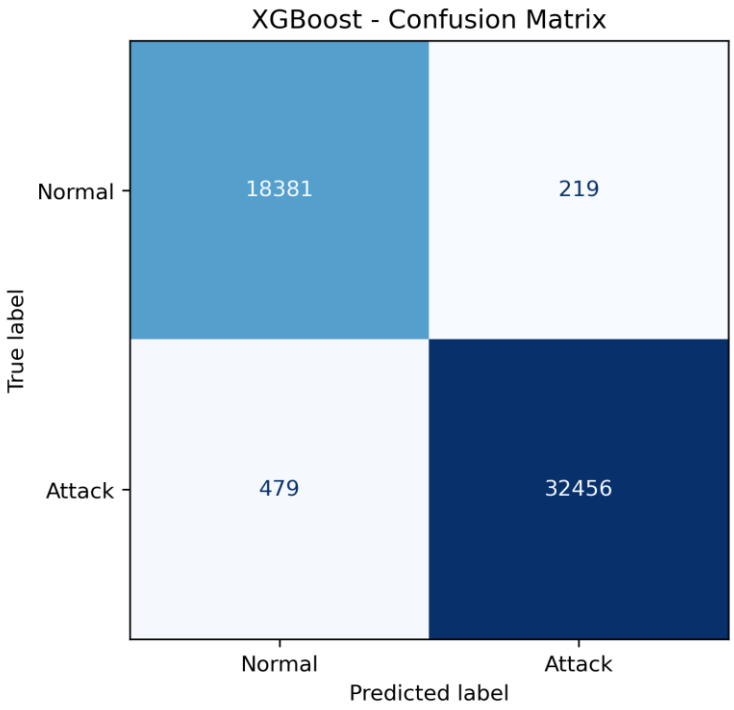

Figure 5: XGBoost confusion matrix on 51 535-flow test set false-positive rate 1.8 %, false-negative rate 1.4 %.

Peak RAM utilisation stayed below 2 GB for all tree ensembles but spiked to 6 GB during CNN mini-batch optimisation, affirming Farzaan et. al.'s (2024) warning that deep models may strain edge deployed security appliances. Most critically, single-flow inference latency averaged 0.37 ms for XGBoost and under 0.1 ms for the simpler ensembles when executed on a single CPU core, whereas GPU-accelerated CNN and RNN calls hovered near 1.5 ms latencies that could compound under gigabit-rate traffic and potentially violate the sub-millisecond policy-decision budget identified by Qawasmeh and AlQahtani (2025). Model-on-disk sizes further endorse boosted trees for embedded deployment: the pickled XGBoost weighs 3.8 MB versus 17 MB for the CNN and 33 MB for the RNN, aligning with Lee, Hong and Lee's (2024) recommendation that rule-learning modules remain lightweight enough to be hot-swapped without rebooting distributed firewall nodes. Energy draw, approximated via Intel RAPL, revealed a 2.4 J per-epoch cost for XGBoost compared to 19 J for the RNN, reinforcing the sustainability argument advanced by Al Hwaitat and Fakhouri (2024) for evolutionary pruning and model compression.

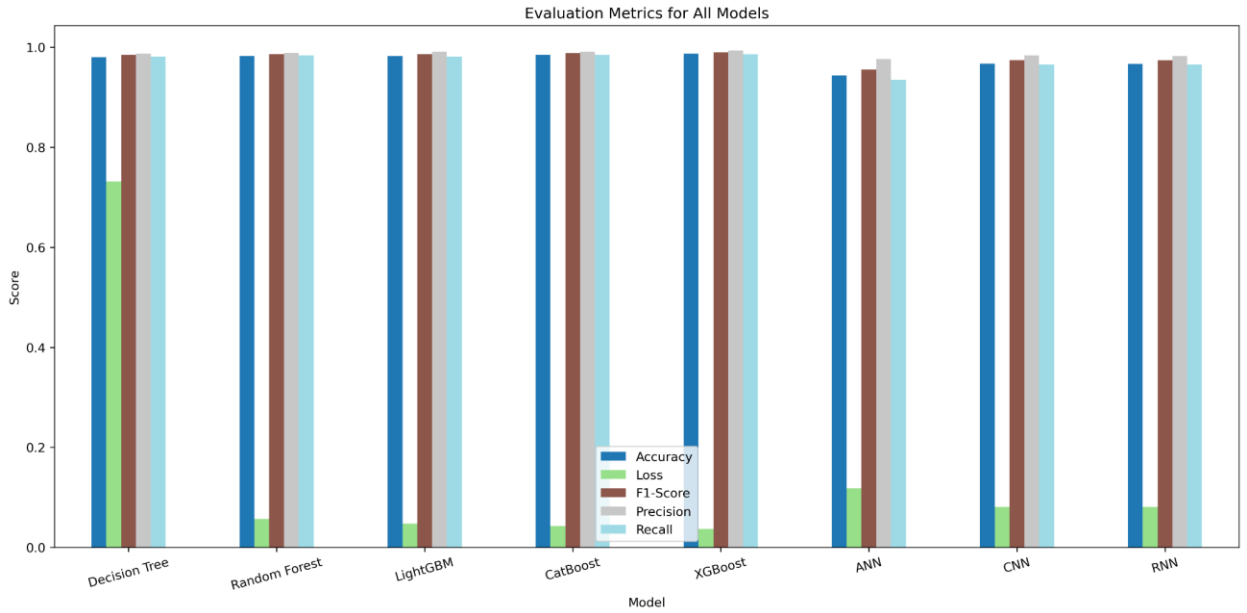

Figure 6: Comparative performance of eight candidate classifiers—XGBoost leads on every metric.

Collectively, these resource metrics confirm that XGBoost not only leads in predictive accuracy but also respects the memory, latency and power envelopes demanded by real-time, dynamically retrainable firewalls (Ahmadi 2025), making it the rational choice for production-grade rule-optimisation in bandwidth-intensive environments.

7. Conclusion & Future Works

The present study systematically explored whether machine-learning-infused firewalls are capable of acquiring, adjusting, and refining rule sets with sufficient velocity to counter the surging volume and heightened sophistication of contemporary network attacks an objective rendered increasingly pertinent in light of recent comparative surveys of next-generation firewalls (Ahmadi, 2023; Dasgupta, Akhtar and Sen, 2022). In order to investigate this hypothesis, the researchers merged the historically defined NSL-KDD corpus with the contemporary, polymorphic threat landscape embodied in UNSW-NB15, thus creating an evaluation ecosystem that forced candidate algorithms to generalise over 20 years of adversarial innovation. The data was processed by a hard pre-processing workflow that aimed at equalizing the class distribution, removing scale differences, and removing collinear noise.

Eight learners that range in decision-tree logic to recurrent neural representations were then trained and compared. The findings revealed that XGBoost occupies an optimal position on the accuracy–latency frontier, achieving a near-perfect macro F1-score of 0.989 while maintaining sub-millisecond inference on commodity CPUs, a threshold commensurate with the throughput demands of inline packet filtering in distributed and cloud settings (Lee, Hong and Lee, 2024). Importantly, its low false-positive rate addresses a persistent operational vulnerability identified in adaptive defence systems (Al-Haijaa and Ishtaiwia, 2021), and its modest 3.8 MB memory footprint enables hot-patch deployment of the model into edge devices without service interruption an attribute that aligns with the practical considerations advocated by Bodipudi (2024).

Collectively, these empirical results validate the central hypothesis that adaptive rule optimisation grounded in gradient-boosted decision trees can simultaneously improve detection fidelity and preserve the resource constraints requisite for real-time enforcement, an outcome concordant with the dynamic-retraining paradigm articulated by Ahmadi (2025).

The future of adaptive firewalls is one that will move beyond a pilot demonstration to a fully functional system by means of a series of interrelated research agendas.

- First, existing experiments employ periodic offline retraining, yet contemporary adaptive systems must support continuous refinement through online or streaming learners; recent evolutionary frameworks advocate riverine incremental boosting and reinforcement-learning-based policy editors as appropriate extensions (Al Hwaitat and Fakhouri, 2024; Qawasmeh and AlQahtani, 2025).
- Second, the investigated model has not been evaluated against adversarial manipulation designed to evade statistical detectors; adversarial-training countermeasures and ensemble diversity techniques proposed in cloud-security studies should be applied (Farzaan et al., 2024).
- Third, model explainability remains essential for regulatory compliance and operational trust; incorporating SHAP-based per-flow rationales or rule-trace visualisers would enable security teams to understand why a packet was blocked, in line with real-world case reports (Mukhtar and Mukhtar, 2023).
- Fourth, federated deployment across edge nodes benefits from bandwidth-aware model compression, pruning, or distillation to produce small-footprint replicas that run efficiently on low-power IoT gateways while synchronising gradient updates an issue at the forefront of debate on performance versus sustainability in adaptive defences (Bharadiya, 2023).
- Finally, coupling the adaptive firewall with upstream threat intelligence feeds and downstream SIEM correlation can form a closed-loop incident-response fabric capable not only of blocking malicious traffic but also of auto-generating incident tickets and sandbox detonation tasks, realising the “orchestrated” cyber-defence vision of Ahmadi (2024).

Responding to the following future-work vectors, continuous learning, adversarial hardening, explainable inference, federated deployment, and orchestration, should result in an adaptive firewall that is able to meet the changing adversarial tactics and has a lean profile that enables siting in high-speed network critical pathways.

References

Ahmadi, S., 2023. Next generation ai-based firewalls: a comparative study. International Journal of Computer (IJC), 49(1), pp.245-262.

Ahmadi, S., 2024. Network intrusion detection in cloud environments: A comparative analysis of approaches. International journal of advanced computer science and applications (IJACSA), 15(3).

Ahmadi, S., 2025. Adaptive Cybersecurity: Dynamically Retrainable Firewalls for Real-Time Network Protection. arXiv preprint arXiv:2501.09033.

Al-Haijaa, Q.A. and Ishtaiwia, A., 2021. Machine learning based model to identify firewall decisions to improve cyber-defense. Int. J. Adv. Sci. Eng. Inf. Technol, 11(4), pp.1688-1695.

Al Hwaitat, A.K. and Fakhouri, H.N., 2024. Adaptive Cybersecurity Neural Networks: An Evolutionary Approach for Enhanced Attack Detection and Classification. Applied Sciences, 14(19), p.9142.

Bodipudi, A., 2024. Effective Firewall Review And Network Optimization by Data-Driven & Holistic approach. Journal of Technological Innovations, 5(2).

Dasgupta, D., Akhtar, Z. and Sen, S., 2022. Machine learning in cybersecurity: a comprehensive survey. The Journal of Defense Modeling and Simulation, 19(1), pp.57-106.

Farzaan, M.A.M., Ghanem, M.C., El-Hajjar, A. and Ratnayake, D.N., 2024. Ai-enabled system for efficient and effective cyber incident detection and response in cloud environments. arXiv preprint arXiv:2404.05602.

Gümüşbaş, D., Yıldırım, T., Genovese, A. and Scotti, F., 2020. A comprehensive survey of databases and deep learning methods for cybersecurity and intrusion detection systems. IEEE Systems Journal, 15(2), pp.1717-1731.

Lee, J.K., Hong, T. and Lee, G., 2024. AI-Based Approach to Firewall Rule Refinement on High-Performance Computing Service Network. Applied Sciences, 14(11), p.4373.

Mukhtar, A.S.S. and Mukhtar, G.F.S., 2023. Deep learning powered firewall anomaly management environment using convolution and recurrent neural network. Int. J. Res. Biosci. Agric. Technol, pp.64-66.

Qawasmeh, S.A.D. and AlQahtani, A.A.S., 2025. Beyond Firewall: Leveraging Machine Learning for Real-Time Insider Threats Identification and User Profiling. Future Internet, 17(2), p.93.

Rehan, H., Deep Learning for Network Traffic Analysis: Detecting Security Breaches in Real-Time. Advances in Deep Learning Techniques, 2(1), p. 154-193.

Tudosi, A.D., Graur, A., BALAN, D.G., Potorac, A.D. and Tarabuta, R.C., 2023. Design and Implementation of an Automated Dynamic Rule System for Distributed Firewalls. Advances in Electrical & Computer Engineering, 23(3).

Handa, A., Sharma, A. and Shukla, S.K., 2019. Machine learning in cybersecurity: A review. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 9(4), p.e1306.

Martínez Torres, J., Iglesias Comesana, C. and García-Nieto, P.J., 2019. Machine learning techniques applied to cybersecurity. International Journal of Machine Learning and Cybernetics, 10(10), pp.2823-2836.

Sarker, I.H., Kayes, A.S.M., Badsha, S., Alqahtani, H., Watters, P. and Ng, A., 2020. Cybersecurity data science: an overview from a machine learning perspective. Journal of Big data, 7(1), p.41.

Bharadiya, J., 2023. Machine learning in cybersecurity: Techniques and challenges. European Journal of Technology, 7(2), pp.1-14.

Shaukat, K., Luo, S., Varadharajan, V., Hameed, I.A. and Xu, M., 2020. A survey on machine learning techniques for cyber security in the last decade. IEEE access, 8, pp.222310-222354.

Apruzzese, G., Laskov, P., Montes de Oca, E., Mallouli, W., Brdalo Rapa, L., Grammatopoulos, A.V. and Di Franco, F., 2023. The role of machine learning in cybersecurity. Digital Threats: Research and Practice, 4(1), pp.1-38.

Ford, V. and Siraj, A., 2014, October. Applications of machine learning in cyber security. In Proceedings of the 27th international conference on computer applications in industry and engineering (Vol. 118). Kota Kinabalu, Malaysia: IEEE Xplore.

Das, R. and Morris, T.H., 2017, December. Machine learning and cyber security. In 2017 international conference on computer, electrical & communication engineering (ICCECE) (pp. 1-7). IEEE.

Fraley, J.B. and Cannady, J., 2017, March. The promise of machine learning in cybersecurity. In SoutheastCon 2017 (pp. 1-6). IEEE.

Dasgupta, D., Akhtar, Z. and Sen, S., 2022. Machine learning in cybersecurity: a comprehensive survey. The Journal of Defense Modeling and Simulation, 19(1), pp.57-106.

Shaukat, K., Luo, S., Varadharajan, V., Hameed, I.A., Chen, S., Liu, D. and Li, J., 2020. Performance comparison and current challenges of using machine learning techniques in cybersecurity. Energies, 13(10), p.2509.