Wiley
International Journal of Distributed Sensor Networks
Volume 2025, Article ID 5973480, 16 pages
https://doi.org/10.1155/dsn/5973480

Research Article

AI-Driven Adaptive Security for Sensor Networks: Next-Generation Firewalls for Attack Detection

Niranjan W. Meegammana and Harinda Fernando

Faculty of Computing, Sri Lanka Institute of Information Technology, Malabe, Sri Lanka

Correspondence should be addressed to Niranjan W. Meegammana; niranjan.meegammana@gmail.com

Received 3 March 2025; Revised 12 June 2025; Accepted 19 June 2025

Academic Editor: Kavita Pandey

Sensor networks are increasingly critical in modern smart environments; however, their limited computational resources expose them to sophisticated cyber threats. Traditional static firewalls and computationally intensive deep learning models are impractical for securing such networks. This study proposes an adaptive next-generation firewall (NGFW) that dynamically switches between shallow and deep AI models based on real-time network load and resource availability. Four neural network models were trained using 20 and 40-feature subsets of the UNSW-NB15 dataset. Two runtime strategies (i) on-demand model loading and (ii) preloaded model switching were developed and evaluated through simulation under real-time conditions. Experimental results indicate that the preloaded approach achieves up to 96% accuracy, 98% precision, and 4-ms inference latency, with a memory footprint of 19 MB, outperforming static AI firewalls in both efficiency and scalability. The proposed NGFW framework offers a resilient and scalable solution for real-time attack detection in resource-constrained environments without requiring frequent model retraining. Future enhancements include hybrid shallow–deep model architectures, continuous federated learning for decentralized adaptability, and the integration of explainable AI to enhance transparency and trustworthiness in edge security deployments.

Keywords: AI-driven security; attack detection; next-generation firewall; sensor networks; shallow–deep hybrid models

1. Introduction

Sensor networks are fundamental to modern smart environments, serving as the backbone for data aggregation and communication in IoT applications. In these environments, edge computing has become increasingly pervasive, enabling low-latency processing and real-time decision-making across a range of critical domains, including agriculture [1], smart grids [2], autonomous vehicles, healthcare, and industrial automation [3].

However, these networks operate in resource-constrained environments with limited computational power, minimal storage, and weak security mechanisms, making them highly susceptible to cyber threats such as unauthorized access, denial-of-service attacks, and data breaches [4]. The rapid integration of sensor-based technologies in Industry 4.0 has expanded the attack surface, exposing vulnerabilities that traditional security mechanisms struggle to address effectively [3]. Therefore, the need for comprehensive security frameworks that can identify, manage, and mitigate IoT and sensor network systems' specific threats was first identified more than a decade ago [5]. In the modern world, where different IoT and sensor network systems are increasingly critical, this kind of frameworks needs to be developed for each system based on its unique features and corresponding security requirements, and then, those frameworks must be implemented using novel techniques to combat the ever-evolving threat landscape that those systems face [6].

Conventional firewalls and intrusion detection system (IDS)/IPS rely on static, rule-based filtering mechanisms, which are insufficient for highly dynamic sensor networks. Their inability to adapt to evolving threats, coupled with high processing overhead, makes them ineffective in real-time attack detection and mitigation [7]. AI-driven approaches, particularly

deep learning (DL) models [8], have demonstrated superior threat detection capabilities [9]. However, computationally intensive models such as long short-term memory (LSTM) networks and transformers are impractical for real-time deployment in sensor networks due to their high resource demands. While lightweight feedforward neural networks (FFNs) offer a more efficient alternative, static AI models still struggle to maintain performance under fluctuating network conditions [10].

To address these challenges, next-generation firewalls (NGFWs) leverage machine learning (ML) for real-time threat detection, analyzing traffic patterns to identify anomalies and predict attacks beyond signature-based methods. They can continuously adapt to emerging threats, enhancing accuracy through AI-driven detection, real-time monitoring, and adaptive security enforcement [11]. However, existing NGFWs often rely on static AI models, which become performance bottlenecks during peak network traffic and fail to adapt dynamically to emerging threats.

The motivation behind this study stems from the urgent need to bridge the gap between high-accuracy threat detection and real-time fast responsiveness in sensor networks [4]. While DL models provide strong detection capabilities, their resource consumption limits practical deployment [10]. On the other hand, shallow models offer low-latency performance but compromise on detection depth and accuracy [9]. Striking a balance between these two extremes is a pressing challenge, particularly when network loads and resource availability can fluctuate significantly. Another key challenge is adaptability. Existing NGFWs often use fixed AI models that cannot respond to changing traffic patterns or emerging threat types [10]. Moreover, security systems must be scalable, lightweight, and privacy-aware, fitting seamlessly into decentralized sensor infrastructures without introducing latency or overhead that could compromise service delivery [7].

This paper addresses these challenges by introducing an adaptive NGFW that dynamically switches between shallow and deep AI models based on real-time conditions, optimizing both accuracy and efficiency while keeping resource usage in check. This study introduces two shallow and two deep models derived from a shallow architecture with a single hidden layer of 512 neurons and a deep architecture with seven layers (256, 128, 64, 32, 16, 8, and 4 neurons). They were trained on 20 and 40-feature datasets to optimize attack detection and security enforcement. The proposed NGFW, by intelligently switching models based on accuracy, latency, and resource availability, ensures efficient attack detection and prevention, overcoming the rigidness of static AI-based firewalls.

This paper is structured as follows: Section 2 reviews existing research on AI-driven security in sensor networks, highlighting key challenges and research gaps. Section 3 details the methodology for evaluating the feasibility of an adaptive NGFW using shallow and deep AI models. Section 4 presents experimental findings, including model performance analysis and real-time adaptability of the NGFW. Finally, Section 5 summarizes key insights, emphasizing the potential of adaptive AI-based NGFWs for sensor network security and future research directions.

2. Literature Review

This review follows the PRISMA approach and systematically selects publications on security in sensor networks from 2022 to 2024. The selection process focuses on keywords such as IoT, Edge Networks, Sensor Networks, Attack Detection, AI-driven Security, Next-Generation Firewalls, and Shallow-Deep Hybrid Models [12]. Given the limitations of traditional firewalls and static AI-based NGFWs in dynamic sensor environments, this review emphasizes adaptive security frameworks that leverage AI for real-time attack detection and resource-aware threat mitigation. The findings highlight research gaps in NGFW implementations and the potential of adaptive AI-driven models to enhance security in resource-constrained sensor networks.

Sensor networks form the backbone of critical systems in smart cities, healthcare, and industrial automation. They are essential for data aggregation, communication, and service delivery in IoT-based solutions. The rapid expansion of sensor networks across industries has significantly increased the global attack surface, posing severe risks to data security, privacy, business stability, and human safety [3]. Conventional firewalls and static IDS solutions struggle to offer real-time protection in these environments. Particularly, these security measures struggle with anomaly and network attack detection due to network complexity, device heterogeneity, increased physical exposure, and real-time monitoring challenges [13]. Given these limitations, AI-driven security solutions have emerged as a promising approach for enhancing sensor network protection [7]. While Majid et al., [4] review over 130 articles on sensor networks in Industry 4.0, covering solutions, coverage types, and limitations, their study overlooks AI applications, security issues, and mitigation strategies.

2.1. Advances in AI-Based Intrusion Detection. The growing demand for intelligent security in IoT has led to the development of numerous AI-driven frameworks. A hybrid FFN–LSTM model proposed by [14] achieved over 99% accuracy on benchmark datasets, yet failed to detail data preparation, making their approach difficult to replicate in real-world conditions.

Popular AI models, such as recurrent neural networks (RNNs), LSTM networks, and transformers, have been applied in network security research [15]. However, their high computational demands make them impractical for low-power sensor nodes. In contrast, FFNs offer efficient network attack detection due to their simplicity and adaptability [9]. By adjusting the number of hidden layers and neurons per layer, FFNs can be optimized for either low-latency shallow models or more complex deep models. Shallow architectures, with a single hidden layer and numerous neurons, reduce model complexity while retaining pattern recognition but struggle with hierarchical data representation. Deep models, consisting of multiple hidden layers, capture hierarchical structures but demand greater computational resources [16]. This trade-off makes adaptive shallow–deep models an effective solution for attack detection in sensor networks, balancing accuracy and computational efficiency.

An RNN–GRU hybrid model introduced by [17] allows strong detection at the application layer, but its computational complexity raises concerns for edge-based deployment. Comparing FFN, LSTM, and random neural network (RandNN) in the CIC-IoT2022 dataset [18] demonstrates impressive detection rates, though they neglected practical deployment factors like memory consumption. Focusing on energy-efficient DL, Ravikumar et al. [19] found that model complexity remained a barrier for constrained sensor nodes.

Other works have taken broader perspectives. Surveying 5G-enabled IoT security [20] offered few AI-based mitigation strategies. Exploring decision tree–based NGFWs for DDoS detection [21] archives higher speed, but their method lacked adaptability to changing attack patterns. Ahmadi [6] performed a comparative analysis of AI-powered NGFWs but noted limitations in empirical evaluation and deployment testing.

Addressing data privacy, Dhakal et al. [22] used federated learning for distributed intrusion detection. While promising, the approach increased communication overhead and latency. Reinforcement learning (RL) was explored by [23] highlighting its potential in adaptive firewalls but at the cost of high training complexity. Emphasizing adversarial resilience, Khaleel et al. [24] advocate for defense strategies like adversarial training and defensive distillation, yet practical evaluation remains sparse.

2.2. AI-Powered NGFW Sensor Networks. Traditional network firewalls rely on perimeter-based security models, which are ineffective in sensor network environments due to the distributed and resource-limited nature of sensor nodes. In contrast, NGFWs integrated with AI models enable real-time attack detection with adaptive security enforcement [11]. Additionally, their use as reverse proxy systems helps mask sensor identities and reduce network congestion [25]. This study builds on [26] by leveraging shallow and deep AI models to develop an NGFW tailored for sensor networks. The proposed adaptive NGFW dynamically switches between shallow and deep AI models based on network conditions and resource constraints, ensuring real-time attack detection and effective attack prevention.

2.3. Similar Work in Sensor Network Security. Analyzing WSN application security [4] highlights the need for improved security techniques. However, it lacks identification of AI-based solutions to enhance security. Examining intrusion detection mechanisms in sensor networks [20] highlights challenges in handling large sensor data volumes and heterogeneous security protocols.

Conducting a literature review, Ali et al. [15] explore IoT security requirements and artificial neural network (ANN) approaches to address specific security features, comparing ANN-based security mechanisms for IDS to identify research gaps. However, the study overlooks issues such as overfitting, scalability, adversarial attacks, hyperparameter sensitivity, and vanishing gradient problems in ANN. Developing a hybrid FFN–LSTM model for sensor-based network attack detection, Jullian et al. [14] achieved 99.95% accuracy on NSL-KDD and BoT-IoT datasets; however, their methodology lacks clarity in data preprocessing and merging.

Discussing AI-powered IoT security, Ravikumar et al. [19] emphasize energy efficiency and adaptability, yet computationally intensive DL models remain impractical for low-power sensor nodes. Introducing an RNN–GRU hybrid model using the ToN-IoT dataset, Khan et al. [17] achieved 98% accuracy at the application layer but with increased latency and a higher risk of overfitting. Evaluating FFN, LSTM, and RandNN for IoT security using the CIC-IoT22 dataset, Bakhsh et al. [18] achieved high accuracy but overlooked resource efficiency and deployment constraints. Exploring ANNs in detecting network attacks on IoT application servers, Meeegammana and Fernando [9] highlight the potential of DL to enhance IoT security. However, it lacks a comparison with other ML models, scalability analysis, and testing in real-world IoT scenarios.

Conducting a review on NGFWs, Ahmadi [6] provides a comparative analysis of AI-based NGFW methodologies, highlighting their strengths and limitations. The study examines ML, DL, hybrid models, behavioral analysis, real-time threat intelligence, and adversarial defense mechanisms, evaluating detection accuracy, false positives, computational efficiency, adaptability, scalability, and robustness. However, it lacks discussion on explainability, scalability, and vulnerabilities to adversarial attacks and defenses, as well as limited empirical evaluation, unclear method selection, and insufficient benchmarking.

The universal approximation theorem establishes that a FFN with a single hidden layer can approximate any continuous function to arbitrary precision, provided it has a sufficient number of neurons and a suitable activation function [27]. However, FFN training faces gradient challenges and overfitting, which require mitigation through tuning activation functions, gradient clipping, batch normalization, and the use of regularization techniques. In their review of adversarial attacks and defense strategies in DL [24], they highlight evasion attacks, poisoning attacks, model inference attacks, and adversarial example transferability. They propose adversarial training, defensive distillation, and input preprocessing as defense mechanisms. However, the study lacks a deeper focus on practical applications with empirical evidence and overlooks key defenses, such as model ensembling and randomization. Table 1 compares the reviewed literature, where RUA denotes resource utilization assessment. Explainability and ethics are also critical in AI-based solutions. Sutaria [29] warned about bias and fairness in AI decision-making, especially when training data is unbalanced or nonrepresentative, which is an issue prevalent in many reviewed systems.

The above studies highlight several gaps. Majid et al. [4] call for improved security but overlook AI-based solutions, while Ali et al. [15] neglect issues like overfitting and scalability, Jullian et al. [14] and Khan et al. [17] face challenges with overfitting and latency, and Bakhsh et al. [18] and Meeegammana and Fernando [9] ignore resource efficiency and real-world deployment. Ahmadi [6] and Wang et al. [27] lack sufficient empirical evaluation and discussion of FFN challenges, limiting practical insights. Abid et al. [30],

TABLE 1: Comparison of literature.

Paper	Technique	Findings	Limitations	Research gaps
[14]	FFN–LSTM hybrid	High detection accuracy (99.95%)	Methodology lacks transparency	Real-world deployment, scalability
[17]	RNN–GRU	Strong app-layer detection	High latency, overfitting	Not optimized for edge devices
[18]	FFN, LSTM, and RandNN	Good performance on CIC-IoT22	No resource usage analysis	Real-time, low-power application
[19]	DL for IoT	Energy-aware designs	High complexity remains	Lightweight model integration
[21]	Decision trees	Fast inference, DDoS detection	Static rules, poor adaptability	Lacks dynamic model selection
[6]	NGFW comparative study	Broad ML/DL coverage	Limited empirical testing	Deployment feasibility
[20]	Survey	Security/privacy in IoT/5G	No AI-based solutions tested	Needs practical frameworks
[22]	Federated learning	Decentralized intrusion detection	High bandwidth usage	Real-time efficiency at the edge
[23]	RL for NGFW	Adaptive learning from traffic	Long training time	Stability in dynamic networks
[24]	Adversarial defense	Explains DL vulnerabilities	Limited empirical evidence	Resilience in live environments
[28]	AI rule refinement	Fast HPC firewall tuning	Static, lacks switching logic	Not suitable for edge networks
[11]	NGFW evolution	Role of AI in NGFWs	Theoretical, lacks practical evaluation	Real-time adaptive switching
[9]	Shallow vs. deep FFN	Model efficiency comparison	No multimodel switching	Adaptive deployment
[26]	Model benchmarking	Performance profiling of FFNs	No field deployment	Integration with NGFW logic
[29]	ML bias review	Ethical AI lens	No technical framework	Bias mitigation in IoT security
[30]	CNN–LSTM hybrid	Improved detection of DDoS attacks in SDN-IoT	High training time, lacks edge optimization	Needs lightweight, real-time IoT-friendly implementation

while advancing detection performance with the CNN–LSTM hybrid model, do not address edge optimization and lightweight deployment concerns. Most studies fail to address model bias from data imbalance and fail to address data balancing or hyperparameter tuning. These gaps indicate the need for comprehensive AI-driven security solutions in sensor networks.

2.4. AI-Based Attack Detection Solutions. Table 2 presents a comparison of AI-based attack detection solutions against the proposed NGFW, highlighting their strengths, weaknesses, and overall effectiveness in real-time, resource-constrained environments.

Unlike traditional AI-based firewalls that rely on static models or require frequent retraining, the proposed NGFW dynamically switches between shallow and deep models based on real-time network conditions. This adaptability ensures a balance between accuracy and efficiency, making it more suitable for resource-constrained environments. Compared to RL and FL-based firewalls, which demand continuous learning and high computational power, the proposed NGFW is optimized for low-resource settings. It also maintains high accuracy with minimal latency while eliminating the need for frequent retraining, ensuring scalability and efficiency in sensor networks with fluctuating resource availability.

2.5. Contribution of This Study. This study examines the deployment of a lightweight, energy-efficient AI-based adaptive NGFW for attack detection in resource-constrained sensor networks. It is an area largely unexplored. By integrating a defense-in-depth strategy, the adaptive NGFW enhances real-time attack detection while optimizing security, resource efficiency, and adaptability in distributed sensor environments. The proposed solution sets a novel future direction for adaptive security in sensor networks.

3. Methodology

3.1. Foundation and Prior Work. The adaptive NGFW proposed in this study builds upon the prior work of [26], who developed four AI models (shallow20, deep20, shallow40, and deep40) optimized for attack detection in edge networks. These models incorporate both shallow and deep architectures to balance computational efficiency and detection accuracy. The shallow models, intended for low-resource environments, consist of a single hidden layer with 512 neurons, whereas the deep models feature seven hidden layers with progressively fewer neurons (256, 128, 64, 32, 16, 8, and 4) to capture complex attack patterns. Two feature sets (20 and 40 features) were used to enhance adaptability across varying resource conditions. To optimize model performance and fairness, hyperparameter tuning including activation function optimization and dropout regularization was conducted using KerasTuner's random search method. To address potential data imbalance, the UNSW-NB15 dataset [31] was preprocessed using random undersampling, ensuring equal representation of attack and benign classes and thereby improving model generalization. The dataset was ultimately narrowed down to 186,000 instances. Further preprocessing involved min–max scaling of feature values,

TABLE 2: Comparison of the proposed NGFW versus AI-based attack detection solutions.

Paper	Approach	Adaptability	Real-time performance	Resource efficiency	Retraining requirement	Scalability
[21]	Decision tree–based firewalls	Static rules do not adapt to changing threats	Fast inference but lacks real-time adaptability	Low resource usage	Requires frequent retraining	Poor scalability to evolving attack types
[23]	Reinforcement learning (RL)–based firewalls	Can adapt to new attack strategies	High training time, slow response	Computationally expensive for sensor networks	Continual learning but requires retraining	Scalable but impractical for low-power devices
[22]	Federated learning (FL)–based firewalls	Distributed learning, updates across devices	High communication overhead	Requires computational power at edge nodes	Reduces the need for centralized retraining	Scalability is limited by communication constraints
[28]	Traditional AI-based firewalls	Uses a fixed model, no real-time switching	Can detect anomalies quickly	High resource consumption for deep models	Requires retraining for new attacks	Poor adaptability to varying network constraints
This work	Proposed adaptive NGFW	Dynamically switches models based on network constraints	Optimized real-time performance by adapting model complexity	Balances accuracy and resource usage	Uses pretrained models and does not require frequent retraining	Scalable to different sensor network conditions

followed by a split into training (90%), validation (5%), and testing (5%) sets for robust and unbiased evaluation.

The four trained models were evaluated based on key performance metrics (accuracy, precision, recall, $F1$ score, and ROC-AUC), alongside resource utilization metrics (model size, memory, and CPU usage). The models achieved test accuracies ranging from 0.93 to 0.98, with deep models excelling in classification (ROC-AUC: 0.996–0.998) but requiring higher computational resources. In contrast, shallow models offered more efficient real-time attack detection, making them suitable for deployment in low-power sensor networks. These findings highlight the trade-off between accuracy and efficiency, reinforcing the need for dynamic model switching in the NGFW to optimize security and performance based on real-time network conditions.

3.2. Adaptive NGFW Algorithm for Securing Sensor Networks. This study developed an adaptive NGFW algorithm that integrates shallow20, deep20, shallow40, and deep40 optimized AI models to secure HTTP, MQTT, and WebSocket services in sensor networks. MQTT facilitates real-time data exchange in smart homes, critical infrastructure, and industrial automation, while HTTP supports web services, RESTful APIs, and cloud applications. WebSocket is essential for real-time sensor data transmission. The NGFW monitors network traffic in real time and dynamically switches between four AI models based on network conditions. In high-traffic, low-resource scenarios, shallow models are prioritized for efficiency, while deep models are activated for higher accuracy. This adaptive selection optimizes detection accuracy while minimizing resource consumption. Benign packets are forwarded, while malicious packets are dropped and logged for continuous training, enhancing security [11]. The model-switching mechanism and classification performance were evaluated using recorded data to assess the algorithm's effectiveness. Figure 1 illustrates the NGFW architecture securing application servers in sensor networks. The NGFW filters inbound and outbound traffic based on AI model predictions, adapting dynamically to network load and resource constraints. It balances speed and precision by switching between low-latency shallow models and high-accuracy deep models. In conjunction with system firewalls like nftables, it blocks persistent malicious traffic, while dropped packets are stored for adversarial training to enhance model robustness. Additionally, the NGFW functions as a reverse proxy, anonymizing the server by concealing its IP and ports, adding an extra layer of security [25].

3.3. Adaptive NGFW Model Switching. The NGFW switches between deep and shallow AI models based on inference speed. Two runtime strategies were developed and tested: (i) on-demand model loading and (ii) preloaded model switching. Both strategies use shared logic to determine the optimal model based on network load conditions and then switch accordingly. Algorithms 1–6 outline the key steps for each approach. The NGFW algorithm includes six key processes: initialization, preloading, on-demand loading, changing inference speed, model switching, and traffic prediction. Figure 2 illustrates its high-level diagram.

Table 3 describes the algorithms illustrated in Figure 2.

The pseudocode for the key algorithms is given below.

3.4. Performance Analysis. The NGFW algorithm was evaluated under near real-world conditions through simulation on a single CPU with 2 GB of memory, dynamically adapting to varying inference speeds to maintain responsiveness under fluctuating network loads. The evaluation utilized two Python scripts, each implementing two asynchronous processes: one for adjusting inference speed for model switching and the other for inference. The inferencing process tested the 20-feature and 40-feature datasets in batches

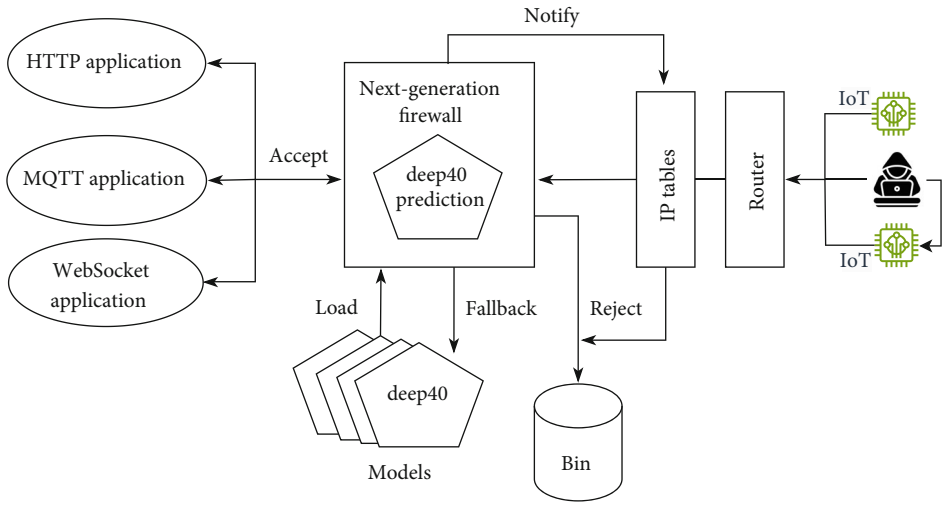

Figure 1: Proposed solution.

```
input: approach either "Pre-Loaded" or "On-Demand",
output: active_model which is the loaded model
Set up global parameters # inference_speed, models, datasets, scalars
load_dataset("20-features")
load_dataset("40-features")
if approach = "Preload" then:
        Deep40 ← load_model("Deep40")
        Shallow40 ← load_model("Shallow40")
        Deep20 ← load_model("Deep20")
        Shallow20 ← load_model("Shallow20")
return active_model ← Deep40
```

Algorithm 1: Initialize ().

```
input: approach as "Pre-Loaded",
       models which is the list of models
output: active_model which is the loaded model
active_model ← initialize ("Pre-Loaded")
while traffic do:
        if inference_speed_change() then:
                switch_model("Pre-Loaded", inference_speed) # inference_speed is a global variable
        predict traffic(traffic)
```

Algorithm 2: Switch models ("preloaded," models).

```
input: approach as "On-Demand",
       models which is the list of models
output: active_model which is the loaded model
active_model ← initialize ("On-Demand")
while traffic do:
        if inference_speed_change() then:
                switch_model("On-Demand", inference_speed)
        predict(traffic)
```

Algorithm 3: Switch models ("on-demand," models).

```
input: none
output: inference_speed # a global variable
random_number ← generate_random_number(1 to 100)
    if random_number mod 4 < 3 then:
        inference_speed ← random_number+1
        switch_model(approach, inference_speed)
```

Algorithm 4: Change inference speed ().

```
input: approach either "Pre-Loaded" or "On-Demand",
    inference_speed as a global variable
output: active_model which is the loaded model,
    updated system variables
if inference_speed = 4 then:
    model = "shallow20"
else if inference_speed =3:
    model ← "deep20"
else if inference_speed =2:
    model ← "shallow40"
else:
    model ←"deep40"
if approach = "On-Demand" then:
    active_model←loadmodel(model)//load model from disk
else:
    active_model ← change_model(model)//Switch model
scaler_data ← model relevant scalar data
Update system state variables
```

Algorithm 5: Switch model (approach, inference_speed).

```
input: none
output:none
data_batch ← extract-test-data-batch()
predictions ← perform-inference(active_model, data_batch)
true_positives←0, false_positives ←0, true_negatives←0, false_negatives ←0//Initialize counter
for each data_point do:
    update counters for true_positives, false_positives, true_negatives, false_negatives
        if prediction = "Benign" then:
            forward-packet(data_point)
        Else:
            log-attack(data_point)
            drop-packet(data_point)
        if traffic-ended() then:
            Calculate classification metrics
            Log classification results
            exit
```

Algorithm 6: Predict traffic().

of 50 samples using dynamically loaded shallow and deep models. Performance metrics [32] including model loading/switching time, memory usage, and inference performance were logged to assess the algorithm's efficiency and adaptability. This evaluation was aimed at demonstrating that the NGFW effectively balances security, resource efficiency, and real-time adaptability in sensor networks.

4. Results and Discussion

The model switching process dynamically selected AI models based on randomly generated inference loads. During high traffic, the system prioritized the lightweight shallow20 model for efficiency, while in lower traffic conditions, it switched to deep20, shallow40, or deep40 models,

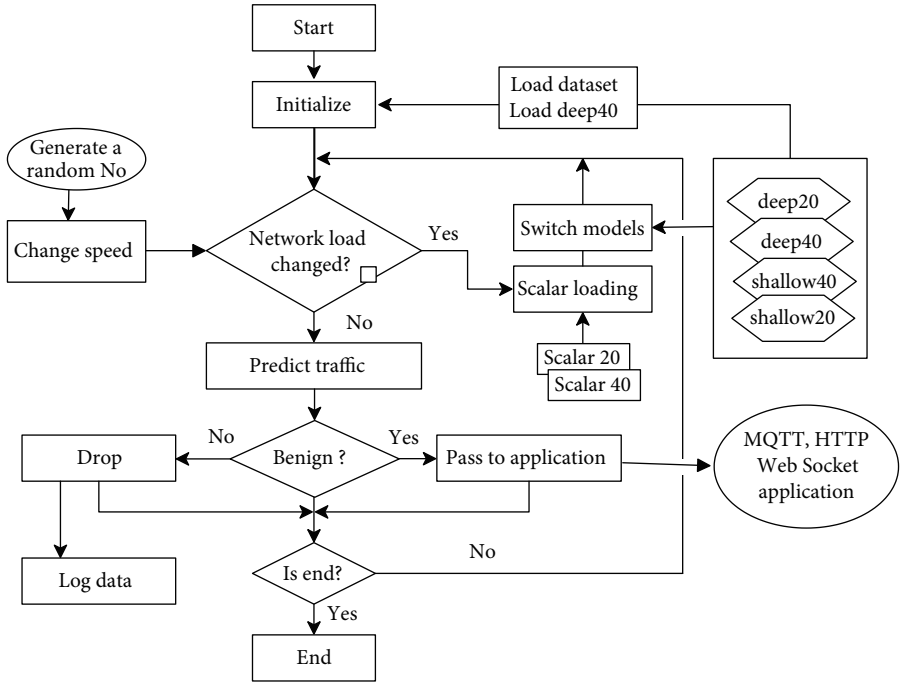

Figure 2: High-level flowchart of the NGFW algorithm.

TABLE 3: NGFW algorithm descriptions.

No.	Process	Description
1	Initialize	The system initializes by loading both the 20-feature and 40-feature datasets and configuring key parameters, including traffic load, inference speed, and the list of available models. In the preloaded approach, all four models are loaded into memory at startup, with deep40 set as the initial active model.
2	Switch models	Upon detecting changes in network load, the system temporarily pauses prediction, loads the appropriate scale data, and activates the relevant AI model.
2.1	Switch models ("preloaded")	Faster switching between AI models is enabled by preloading all required models at program startup. With models already resident in memory, I/O latency during runtime is minimized, allowing for seamless transitions in response to dynamic network conditions. This approach offers significant performance advantages but incurs a higher memory footprint, potentially limiting its suitability for low-resource environments. Therefore, it is best suited for high-performance sensor networks that can support persistent model loading.
2.2	Switch models ("on-demand")	AI models are loaded into memory selectively, using an on-demand strategy that minimizes memory usage, ideal for deployment on devices with limited storage capacity. However, this approach introduces I/O latency during model switching, which can impact responsiveness. To address this, the system carefully balances memory efficiency with switching overhead to ensure real-time detection remains effective without overburdening the device's limited resources.
3	Change speed	During testing, the inference speed is randomly varied to simulate changes in network load.
4	Predict traffic	The asynchronous attack detection process continuously analyzes network traffic and temporarily pauses during model switching. It classifies incoming traffic as either benign or malicious. Benign traffic is forwarded to the application for processing, while malicious traffic is dropped, logged for further analysis, and retained for adversarial training to enhance future detection accuracy. The active model performs batch-based prediction, enabling efficient traffic classification under dynamic network conditions.

respectively, for improved accuracy. The NGFW tested two model-switching strategies. The on-demand loading approach loads models only when needed to minimize memory usage. In the preloading approach, models were loaded at the initialization and remained in memory to reduce switching time.

4.1. Model Loading Time Analysis. The on-demand model loading approach loaded models 129 times across four inference speeds, 1 to 4, distributed randomly but nearly equally in the NGFW framework. Table 4 shows the count of models loaded and loading time statistics in seconds.

TABLE 4: On-demand model loading times of models.

Model	Count	Mean (s)	Std (s)	Min (s)	Max (s)
deep20	31	0.178	0.046	0.142	0.240
deep40	31	0.176	0.029	0.126	0.359
shallow20	35	0.056	0.013	0.038	0.087
shallow40	32	0.057	0.015	0.036	0.091
Overall	32.25	0.117	0.026	0.085	0.194

In the loading-on-demand approach, the deep20 and deep40 models demonstrate similar mean values, but the deep40 model exhibits a significantly higher maximum loading time. The shallow20 and shallow40 models exhibited much lower mean values. The deep40 and deep20 recorded a mean loading time of around 0.177 s, while the shallow models' mean falls around 0.056 s, demonstrating a significant difference. The overall mean loading time of 0.117 s represents the combined performance of the preloading approach, with deep models contributing more due to their complexity.

Figure 3 illustrates the distribution and variability of model loading times in the on-demand model loading approach. The deep models demonstrate higher mean values and greater variability compared to the shallow models, which are more consistent but consume lower resources.

Figure 4 illustrates that shallow models load better in the on-demand approach compared to deep models. They handled more than 60% of the test traffic.

The preloading approach reduced the delay typically associated with on-demand loading. As shown in Table 5, when preloaded, the deep40 model consumed significantly more memory than other models due to its complexity and number of layers. The memory used by shallow20 is negligible. The shallow models were significantly faster. Garbage collection plays a role in optimizing memory usage, but deep models still require substantial memory after cleanup.

While preloaded model switching minimizes latency by keeping all models resident in memory, it incurs a fixed memory cost of approximately 19 MB. This approach is well suited for edge gateways with sufficient resources. In contrast, the on-demand strategy dynamically loads models based on network conditions, resulting in variable memory usage ranging from 0.001 MB (shallow20) to 17.8 MB (deep40). Although this introduces model switching latency, it offers a memory-efficient alternative ideal for low-memory sensor nodes where resource constraints are critical. Hence, the preloaded approach is optimal for performance-critical environments, while the on-demand approach is better for memory-constrained deployments.

Overall, as shown in Table 6, the preloading approach offers a significant advantage over the on-demand approach in model switching during inference. It incurs virtually no delay (0.0 s), ensuring instant transitions. In contrast, the on-demand approach introduces noticeable latency and variability in loading times, leading to performance issues when switching between models. Thus, preloading reduces delays and enhances performance, especially in scenarios requiring frequent model switching. The CPU usage in both approaches was nearly the same, 0.32% in the preloaded approach and 0.33% in the on-demand approach.

4.2. Prediction Time Analysis. As shown in Table 7, the comparison of prediction times for preloaded and on-demand approaches reveals some interesting insights. During on-demand model loading, the deep models exhibited slightly higher mean prediction times and greater variability due to their complexity. In contrast, the shallow models were generally faster, with shallow20 being the most efficient choice for low-resource environments. The preloading approach had a mean prediction time of 0.087624 s, which is marginally lower than the mean prediction time of 0.087694 s for the on-demand loading approach.

The preloaded models demonstrated greater stability with a lower standard deviation, ensuring more consistent prediction times across all models. This makes preloading of models preferable for real-time applications where reliability is crucial. In contrast, on-demand models exhibit higher variability, illustrated in Figure 5. The higher standard deviation and larger maximum values of the on-demand approach indicate occasional performance spikes and potential latency issues.

Figure 6 illustrates the mean variability of prediction times of on-demand and preloading approaches.

Despite their similar mean prediction times, the increased variance in the on-demand approach suggests that, while they perform comparably on average, they may introduce delays in certain cases. As a result, preloaded models are generally more efficient when consistency and predictability are key factors.

4.3. Time Complexity Analysis. The time complexity analysis of the on-demand approach (ngfw_multi_model1.py) and the preloaded approach (ngfw_multi_model2.py) highlights the efficiency gains of preloading models. The preloaded approach is significantly more efficient as it reduces the model switching overhead from $O(m)$ to $O(1)$ by initializing all models at startup. In contrast, the on-demand approach loads models dynamically each time a switch occurs, introducing additional computational overhead. However, both approaches share the same inference complexity of $O(n \times f \times w \times l)$, where n is the number of samples, f is the number of features, w is the number of neurons per layer, and l is the number of layers. Additionally, logging, data preprocessing, and adaptive speed functions remain identical in both implementations, ensuring consistency in data handling. Overall, the preloaded approach is the preferred choice for performance optimization due to its reduced switching overhead and increased efficiency, as shown in Table 8.

Figure 7 illustrates the comparison of time complexities between the on-demand model loading approach and the preloaded model approach. The plot demonstrates that the preloaded approach offers superior performance by reducing the model switching overhead from $O(m)$ to $O(1)$, while maintaining similar inference complexities. This confirms that preloading models is a more efficient strategy for reducing time complexity in real-time applications.

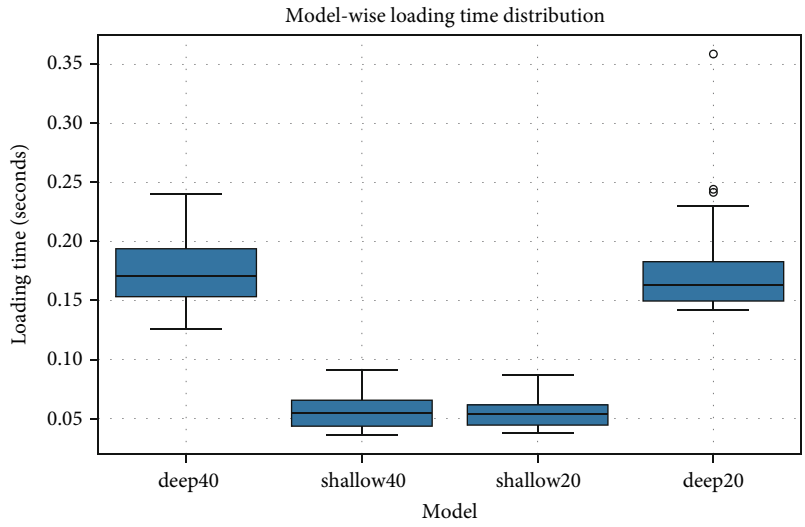

Figure 3: On-demand model-wise loading time variability.

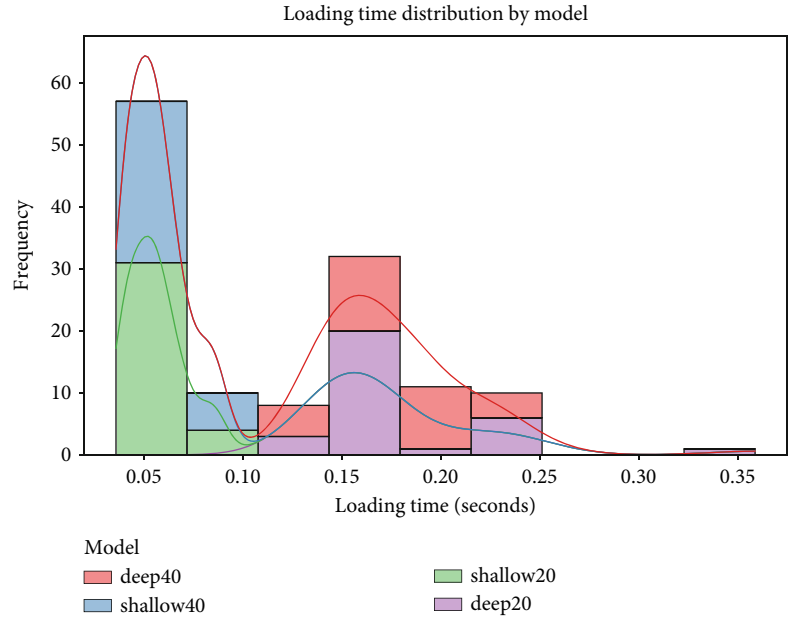

Figure 4: On-demand model-wise loading time distribution.

TABLE 5: Memory used by models.

Model	Memory used (MB)
deep40	17.872
shallow40	0.023
deep20	1.141
shallow20	0.001
Total memory used	19.037

4.4. Classification Metrics. Table 9 summarizes the classification metrics for both on-demand and preloading approaches.

The preloading approach offers a slight performance improvement over on-demand loading, achieving higher accuracy and an improved $F1$ score, indicating a marginally better balance between precision and recall. Additionally, preloading enhances precision, reducing false positives, while recall remains nearly identical. Furthermore, it eliminates model switching delays (0.0 s), a significant advantage for latency-sensitive applications. Although the overall metrics are similar, these slight improvements and reduced latency make preloading a more attractive option when both performance and speed are priorities. The adaptive model selection strategy optimizes both speed and accuracy based on available resources, making it a versatile solution for various attack detection use cases in sensor networks using an NGFW.

The comparison of intrusion detection techniques across five studies shown in Table 10 reveals key performance advantages of the proposed work, "adaptive NGFW using

TABLE 6: Comparison of preloading versus on-demand approach.

Approach	Model loading time (s)	Model switch time (s)	CPU usage (%)
On-demand	Varies (0.036–0.359 s)	Significant	0.33
Preloading	Fixed times (deep models ~0.13–0.197 s, shallow models ~0.039–0.047 s)	0.0 s (instant)	0.32

TABLE 7: Comparison of preloaded versus on-demand model prediction time.

Model	Approach	Count	Mean (s)	Std (s)	Min (s)	Max (s)
deep20	Preloaded	103	0.090	0.016	0.061	0.146
	On-demand	87	0.093	0.035	0.053	0.226
deep40	Preloaded	103	0.087	0.018	0.052	0.187
	On-demand	92	0.095	0.032	0.049	0.187
shallow20	Preloaded	82	0.087	0.015	0.053	0.149
	On-demand	98	0.081	0.021	0.052	0.163
shallow40	Preloaded	82	0.085	0.013	0.059	0.112
	On-demand	93	0.083	0.025	0.053	0.212
Overall	Preloaded	370	0.087	0.016	0.052	0.149
	On-demand	370	0.088	0.028	0.052	0.197

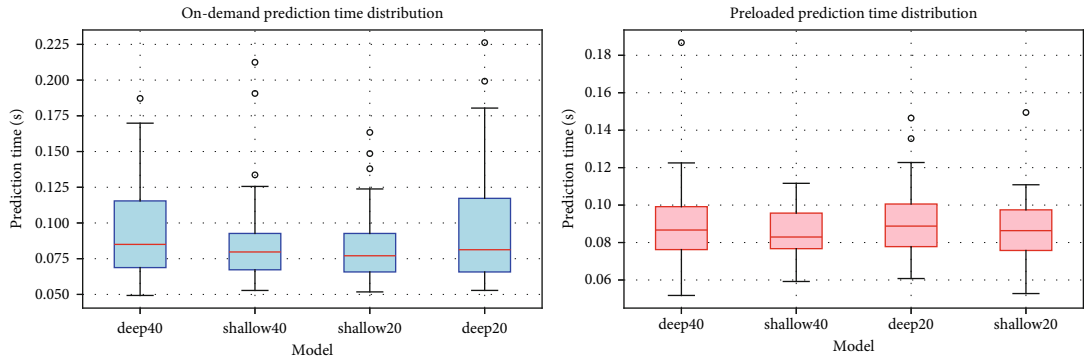

FIGURE 5: Prediction time variability of models in two approaches.

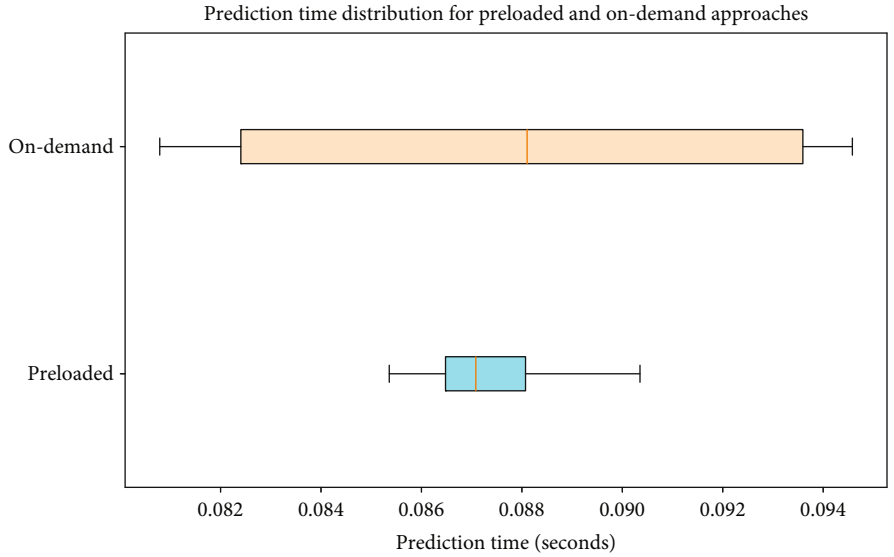

FIGURE 6: Distribution of mean prediction times for approaches.

TABLE 8: Time complexity analysis.

Operation	On-demand complexity ngfw_multi_model2.py	Preloading complexity ngfw_multi_model1.py
Model loading	$O(m)$ (loads model dynamically)	$O(1)$ (preloads all models)
Data preprocessing	$O(n \times f)$ (for n samples and f features)	$O(n \times f)$ (same)
Model switching	$O(m)$ (loads model on-demand)	$O(1)$ (switch preloaded models)
Inference	$O(n \times f \times w \times l)$ (for n samples, f features, w neurons per layer, and l layers)	$O(n \times f \times w \times l)$ (same)
Logging	$O(1)$ per log entry	$O(1)$ per log entry
Total complexity	$O(m) + O(n \times f \times w \times l)$	$O(n \times f \times w \times l)$

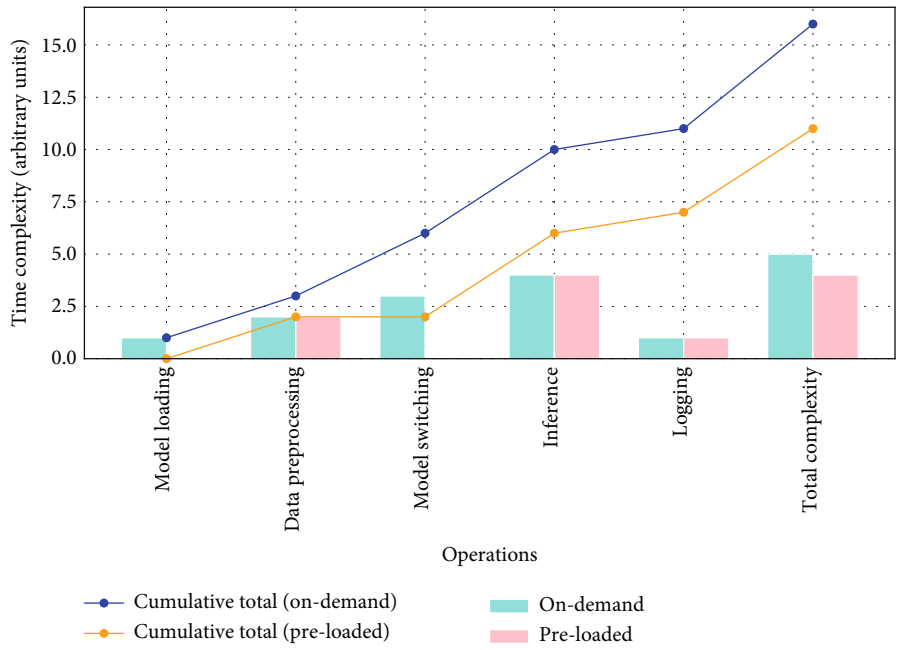

FIGURE 7: Comparison of time complexities between the on-demand model loading approach.

TABLE 9: Comparison of classification metrics of two approaches.

Approach	Accuracy	Precision	Recall	*F*1 score
On-demand loading	0.9576	0.9687	0.9458	0.9571
Preloading	0.9589	0.9759	0.9453	0.9584

4 DL models." Unlike the others, this work offers a well-rounded performance profile with 96% accuracy, 98% precision, 95% recall, 96% *F*1 score, and memory use of 19 MB. It outperforms or matches the best metrics across prior studies. For instance, although [21] reports a slightly higher accuracy at 99%, it lacks transparency on precision, recall, and memory usage. Dhakal et al. [22] offer decent performance with federated learning, yet it peaks at 88% recall and lacks full metric coverage. Lee et al. [28] and Kheddar et al. [23] excel in niche areas but either omit key statistics or face scalability concerns in constrained environments. In contrast, this work not only maintains top-tier detection performance but also demonstrates practical deployment readiness with a lightweight 19 MB memory footprint, making it ideal for real-world, resource-sensitive environments. Thus, its balanced excellence across accuracy, efficiency, and transparency justifies it as a superior, production-ready IDS approach.

4.5. Real-World Challenges. Deploying the proposed solution in low-resource environments presents several challenges. The shallow20 model, offering a balance between latency, accuracy, and resource consumption, is preferable where deep models are impractical. NGFWs must efficiently process real-time sensor traffic while minimizing latency spikes that could delay attack detection. AI models remain susceptible to evasion attacks, requiring continuous adversarial training using dropped packets to enhance resilience. Seamless NGFW integration with MQTT, HTTP, and WebSocket protocols across heterogeneous devices is critical for long-term adaptability. Furthermore, edge-based AI deployments must incorporate load balancing to avoid centralized points of failure. Compliance with stringent privacy regulations in healthcare, smart cities, and industrial applications demands privacy-preserving AI techniques. Mitigation measures include applying homomorphic encryption to secure data

TABLE 10: Comparison of performance metrics with similar work.

Paper	Technique	Accuracy (%)	Precision (%)	Recall (%)	*F*1 score (%)	Memory use
Lee et al. [28]	KNN, neural network, SVM	98	NA	NA	96	NA
Dhakal et al. [22]	Federated learning (FedAvg, FedAvg+)	NA	85	88	87	Edge-optimized
Patel et al. [21]	Binary decision tree, XGBoost, SVM	99	NA	NA	NA	NA
Kheddar et al. [23]	Deep reinforcement learning (DRL), Q-learning	Varies	98.5	99.3	98.8	Constrained
This work	Adaptive NGFW with 4 DL models	96	98	95	96	19 MB

during AI inference and implementing differential privacy through noise-based anonymization techniques to protect sensitive user data in smart city and healthcare contexts [33].

4.6. Deployment Limitations and Practical Considerations. While the adaptive NGFW has been rigorously validated using high-fidelity simulations and real-time emulation loops, the system has not yet been deployed in live sensor network environments. This represents a limitation in assessing long-term behavior, fault tolerance, and response to unpredictable traffic or node failure. However, the current architecture has been designed to be modular and lightweight enough for near-edge and embedded deployment, and our test scenarios closely mirror operational traffic patterns using the UNSW-NB15 dataset.

4.7. Scalability Considerations. Dynamically switching between low-resource shallow models and high-accuracy deep models based on real-time network conditions allows the adaptive NGFW to optimize performance. However, frequent model switching introduces performance overhead, necessitating caching preloaded models to minimize delays.

As shown in Figure 8, the NGFW system experiences a rise in CPU usage as the model switching frequency increases. While low-frequency switching, less than 5/min, incurs minimal overhead, frequent switching, more than 20/min, leads to a noticeable increase in CPU load. This cumulative computational cost, if left unoptimized, can affect large-scale deployments where hundreds of nodes operate concurrently. This trade-off underscores the need for intelligent switch threshold management and batching logic. Current work focuses on using moving average inference rates and predictive rate-limiting to reduce unnecessary switching while maintaining adaptability, throughput, and resource efficiency at scale. To address this, future work will explore dynamic batching, inference rate throttling, and switch threshold tuning to maintain responsiveness without overburdening the system.

Moreover, a distributed approach using federated learning [33] enhances scalability and resilience. Local attack detection at edge nodes reduces reliance on central servers, mitigating network congestion. Preloading models significantly reduces switching delays, making them ideal for real-time applications. However, this increases memory consumption, requiring careful resource allocation strategies to balance efficiency and performance. Furthermore, the NGFW framework should implement incremental learning to adapt to evolving attack patterns. Federated learning enables AI models to be trained collaboratively across multiple sensor networks, ensuring continuous improvement while preserving data privacy.

Overall, these results confirm the NGFW's adaptability, maintaining high detection accuracy and efficient real-time traffic classification. By integrating adaptive security mechanisms, the proposed NGFW offers a scalable and robust attack detection solution for modern sensor networks. The optimal choice between on-demand and preloaded switching ultimately depends on resource availability and inference load. In summary, the NGFW leverages adaptive AI models to balance accuracy and response time, optimizing performance based on network conditions.

4.8. Novelty of This Study. This study introduces a novel adaptive AI-driven NGFW specifically designed for securing resource-constrained sensor networks—a domain where current security frameworks remain inadequate. Its key innovation lies in a real-time model-switching mechanism that dynamically selects from four neural networks (two shallow and two deep), enabling intelligent adaptation to varying traffic loads and hardware limitations. This is the first implementation to offer dual runtime strategies: (i) on-demand model loading for memory-constrained deployments and (ii) preloaded model switching for latency-sensitive environments. This flexibility allows seamless operation across diverse infrastructure—from lightweight edge nodes to high-capacity gateways. Validated using the UNSW-NB15 dataset, the system achieves 96% accuracy, 98% precision, and 95% recall, with 4-ms latency and a compact 19-MB memory footprint. These performance metrics, coupled with adaptive deployment strategies, make the system highly effective for real-time protection in IoT, healthcare, smart grid, and industrial automation contexts.

In contrast to existing NGFWs, which rely on static AI models or lack adaptability under fluctuating resource constraints, this work introduces modular model-switching logic, resource-aware optimization, and lightweight implementation, addressing major limitations in the literature. It defines a new benchmark for scalable, responsive, and efficient AI-based firewalls. Moreover, this framework lays the groundwork for future enhancements, including federated learning, adversarial training, and explainable AI (XAI), to improve robustness, transparency, and privacy in distributed environments. As such, it presents a practical and forward-looking solution to the evolving cybersecurity challenges in next-generation sensor networks.

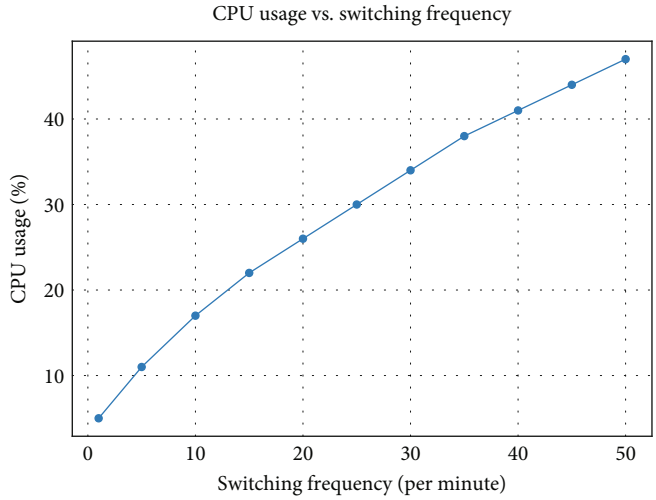

Figure 8: CPU usage versus model switching frequency.

4.9. Future Work. Building on the adaptive NGFW framework introduced in this study, future work will enhance system adaptability, resilience, and deployment efficiency. Advanced shallow–deep hybrid AI architectures will be developed, combining the low-latency response of shallow models with the deep feature extraction capabilities of deep networks to support dynamic traffic conditions and hardware limitations. Continuous learning techniques, including incremental and online learning, will be implemented to allow the NGFW to adapt to evolving threats without full retraining. Federated learning will also be integrated to enable decentralized, privacy-preserving model updates across distributed nodes [22]. Hardware-aware optimizations such as model quantization, pruning, and lightweight inference will ensure efficient operation on resource-constrained devices [34]. Neural processing unit (NPU) integration will be explored to accelerate AI inference while minimizing CPU and memory usage [35]. XAI techniques will be adopted to enhance system transparency, trust, and diagnostic capabilities in critical infrastructure deployments [36].

Initial validation will be conducted on two testbeds: a smart campus sensor network for environmental monitoring and an IoT-enabled agriculture setup, both employing Raspberry Pi–based nodes with Wi-Fi and ZigBee connectivity [37]. To further strengthen security, adversarial defense strategies such as model ensembling and adversarial training will be integrated [38]. RL and continuous policy adaptation mechanisms will also be employed to dynamically update security measures against evolving threats [39].

In addition, AI-powered Zero Trust frameworks will be explored to ensure continuous authentication of sensor nodes before granting network access [40]. AI-driven threat intelligence, leveraging real-time feeds and pattern recognition, will be utilized to detect and mitigate advanced persistent threats (APTs) in resource-constrained environments [41]. Future work will also enable real-time edge-to-cloud synchronization for NGFW models to enhance global threat mitigation [42]. Blockchain integration will be explored to provide tamper-proof security logging and preserve log integrity across decentralized networks [43], further enhancing system adaptability and resilience.

Finally, future NGFW iterations will incorporate privacy-by-design principles—including differential privacy, consent-based logging, and policy-driven model auditing—to ensure compliance with global regulations such as GDPR and HIPAA [6, 44]. These advancements are aimed at facilitating lawful, scalable, and secure AI-driven cybersecurity for next-generation sensor networks. These deployments will evaluate latency, model switching accuracy, and adaptability under varying network constraints. Long-term validation will extend to smart city grids, industrial IoT systems, and healthcare sensor networks to assess scalability and operational robustness.

5. Conclusion

This study presented the design and evaluation of an adaptive AI-driven NGFW specifically optimized for securing resource-constrained sensor network environments. By dynamically switching between shallow and deep neural network models in response to real-time network conditions, the proposed NGFW achieved a critical balance between detection accuracy, computational efficiency, and latency optimization. Empirical validation using the UNSW-NB15 dataset demonstrated that preloaded model switching substantially reduced inference latency while maintaining a high classification performance, achieving an accuracy of 96% and a precision of 98% and sustaining a minimal memory footprint of 19 MB. The dual runtime strategies—namely, on-demand model loading and preloaded model switching—offered significant flexibility, thereby enabling deployment across diverse hardware environments ranging from ultralow-resource edge devices to high-performance gateway nodes. Comparative analyses against contemporary AI-based IDSs highlighted the superior adaptability, scalability, and deployment readiness of the proposed framework, particularly in dynamic and heterogeneous sensor network ecosystems.

Furthermore, this research advances the state of the art by integrating lightweight AI architectures with real-time model-switching logic, thereby addressing the critical challenges of resource management, threat detection latency, and model adaptability. The findings also lay a robust foundation for future work in areas such as hybrid shallow–DL architectures, federated learning for decentralized model evolution, XAI for system transparency, and privacy-preserving security frameworks to ensure regulatory compliance in critical domains. In summary, the adaptive NGFW framework introduced in this study offers a practical, scalable, and resilient cybersecurity solution, bridging the persistent gap between high-performance AI-driven threat detection and the stringent operational constraints inherent to next-generation sensor networks.

Data Availability Statement

The data supporting this study's findings are openly available at https://github.com/niranjanmeegammana/SD_NGFW_JP.

Conflicts of Interest

The authors declare no conflicts of interest.

Funding

No funding was received for this manuscript.

Acknowledgments

In accordance with the AI Policy, we acknowledge the use of ChatGPT 4 for copyediting purposes. The tool was used solely for language refinement, grammar correction, and readability improvements, with no influence on the research content, analysis, or conclusions presented in this manuscript.

References

[1] K. Pandey, S. Jain, D. Pandey, and O. Kaiwartya, eds., *The Future of Agriculture: IoT, AI and Blockchain Technology for Sustainable Farming* (Bentham Science Publishers, 2024), https://doi.org/10.2174/9789815274349124010l.

[2] W. Lei, H. Wen, J. Wu, and W. Hou, "MADDPG-Based Security Situational Awareness for Smart Grid With Intelligent Edge," *Applied Sciences* 11, no. 7 (2021): 3101, https://doi.org/10.3390/app11073101.

[3] A. Alwarafy, K. A. Al-Thelaya, M. Abdallah, J. Schneider, and M. Hamdi, "A Survey on Security and Privacy Issues in Edge Computing-Assisted Internet of Things," *IEEE Internet of Things Journal* (p. 1, https://doi.org/10.1109/jiot.2020.3015432.

[4] M. Majid, S. Habib, A. R. Javed, et al., "Applications of Wireless Sensor Networks and Internet of Things Frameworks in the Industry Revolution 4.0: A Systematic Literature Review," *Sensors* 22, no. 6 (2022): 2087, https://doi.org/10.3390/s22062087.

[5] H. S. Fernando and J. H. Abawajy, "A Security Framework for Networked RFID," in *Internet and Distributed Computing Advancements* (IGI Global Scientific Publishing, 2012), 85–114, https://doi.org/10.4018/978-1-4666-0161-1.ch004.

[6] S. Ahmadi, "Next Generation AI-Based Firewalls: A Comparative Study" 2023, ResearchGate. Available at: https://www.researchgate.net/publication/377060591_Next_Generation_AI-Based_Firewalls_A_Comparative_Study [Accessed 18 Feb. 2025].

[7] N. Moustafa, M. Abdel-Basset, and R. Mohamed, *Deep Learning Approaches for Security Threats in IoT Environments* (Wiley-IEEE Press, 2022).

[8] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, 2016).

[9] N. W. Meegammana and H. Fernando, "Detection of Network Attacks on Application Servers Using Deep Learning in IoT Environments," in *Proceedings of the 2023 5th International Conference on Advancements in Computing (ICAC)* (IEEE, 2023), https://doi.org/10.1109/icac60630.2023.10417159.

[10] S. T. Haque, M. Debnath, A. Yasmin, T. Mahmud, and A. H. H. Ngu, "Experimental Study of Long Short-Term Memory and Transformer Models for Fall Detection on Smartwatches," *Sensors* 24, no. 19 (2024): 6235–6235, https://doi.org/10.3390/s24196235.

[11] J. Liang and Y. Kim, "Evolution of Firewalls: Toward Securer Network Using Next Generation Firewall," in *Proceedings of the 2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC)* (IEEE, 2022), https://doi.org/10.1109/CCWC54503.2022.9720435.

[12] Y. A. Abid, J. Wu, M. Farhan, and T. Ahmad, "ECMT Framework for Internet of Things: An Integrative Approach Employing In-Memory Attribute Examination and Sophisticated Neural Network Architectures in Conjunction With Hybridized Machine Learning Methodologies," *IEEE Internet of Things Journal* 11, no. 4 (2023): 5867–5886, https://doi.org/10.1109/JIOT.2023.3312152.

[13] E. Schiller, A. Aidoo, J. Fuhrer, J. Stahl, M. Ziörjen, and B. Stiller, "Landscape of IoT Security," *Computer Science Review* 44, no. 44 (2022): 100467, https://doi.org/10.1016/j.cosrev.2022.100467.

[14] O. Jullian, B. Otero, E. Rodriguez, N. Gutierrez, H. Antona, and R. Canal, "Deep-Learning Based Detection for Cyber-Attacks in IoT Networks: A Distributed Attack Detection Framework," *Journal of Network and Systems Management* 31, no. 2 (2023): https://doi.org/10.1007/s10922-023-09722-7.

[15] Y. Ali, H. U. Khan, and M. Khalid, "Engineering the Advances of the Artificial Neural Networks (ANNs) for the Security Requirements of Internet of Things: A Systematic Review," *Journal of Big Data* 10, no. 1 (2023): https://doi.org/10.1186/s40537-023-00805-5.

[16] I. H. Sarker, "AI-Based Modeling: Techniques, Applications and Research Issues Towards Automation, Intelligent and Smart Systems," *SN Computer Science* 3, no. 2 (2022): 158, https://doi.org/10.1007/s42979-022-01043-x.

[17] N. W. Khan, M. S. Alshehri, M. A. Khan, et al., "A Hybrid Deep Learning-Based Intrusion Detection System for IoT Networks," *Mathematical Biosciences and Engineering* 20, no. 8 (2023): 13491–13520, https://doi.org/10.3934/mbe.2023602.

[18] S. A. Bakhsh, M. A. Khan, F. Ahmed, M. S. Alshehri, H. Ali, and J. Ahmad, "Enhancing IoT Network Security Through Deep Learning-Powered Intrusion Detection System," *Internet of Things* 24 (2023): 100936, https://doi.org/10.1016/j.iot.2023.100936.

[19] K. C. Ravikumar, P. Chiranjeevi, N. M. Devarajan, C. Kaur, and A. I. Taloba, "Challenges in Internet of Things Towards the Security Using Deep Learning Techniques," *Measurement:*

Sensors 24 (2022): 100473, https://doi.org/10.1016/j.measen.2022.100473.

[20] J. Cook, S. U. Rehman, and M. A. Khan, "Security and Privacy for Low Power IoT Devices on 5G and Beyond Networks: Challenges and Future Directions," *IEEE Access* 11 (2023): 39295–39317, https://doi.org/10.1109/access.2023.3268064.

[21] M. Patel, P. P. Amritha, V. B. Sudheer, and M. Sethumadhavan, "DDoS Attack Detection Model Using Machine Learning Algorithm in Next Generation Firewall," *Procedia Computer Science* 233 (2024): 175–183, https://doi.org/10.1016/j.procs.2024.03.207.

[22] R. Dhakal, W. Raza, V. Tummala, and L. Niure Kandel, "Enhancing Intrusion Detection in IoT Networks Through Federated Learning," *IEEE Access* 12 (2024): 167168–167182, https://doi.org/10.1109/ACCESS.2024.3495702.

[23] H. Kheddar, D. W. Dawoud, A. I. Awad, Y. Himeur, and M. K. Khan, "Reinforcement-Learning-Based Intrusion Detection in Communication Networks: A Review," *IEEE Communications Surveys & Tutorials* (p. 1, https://doi.org/10.1109/COMST.2024.3484491.

[24] Y. L. Khaleel, M. A. Habeeb, A. S. Albahri, T. Al-Quraishi, O. S. Albahri, and A. H. Alamoodi, "Network and Cybersecurity Applications of Defense in Adversarial Attacks: A State-of-the-Art Using Machine Learning and Deep Learning Methods," *Journal of Intelligent Systems* 33, no. 1 (2024): https://doi.org/10.1109/COMST.2024.3484491.

[25] Solar Winds, "*What Is a Reverse Proxy?*" 2024, solarwinds.com. Available at: https://www.solarwinds.com/resources/it-glossary/reverse-proxy [Accessed 24 Jun. 2024].

[26] N. W. Meegammana and H. Fernando, "Securing IoT Servers: Shallow vs. Deep Neural Network Architectures," in *Proceedings of the ICTER 2024* (Zenodo, 2024), https://doi.org/10.5281/zenodo.14899236.

[27] J. Wang, S. Wu, H. Zhang, B. Yuan, C. Dai, and N. R. Pal, "Universal Approximation Abilities of a Modular Differentiable Neural Network," *IEEE Transactions on Neural Networks and Learning Systems* 36, no. 3 (2024): 5586–5600, https://doi.org/10.1109/tnnls.2024.3378697.

[28] J.-K. Lee, T. Hong, and G. Lee, "AI-Based Approach to Firewall Rule Refinement on High-Performance Computing Service Network," *Applied Sciences* 14, no. 11 (2024): 4373–4373, https://doi.org/10.3390/app14114373.

[29] N. Sutaria, "Bias and Ethical Concerns in Machine Learning" 2022, ISACA. Available at: https://www.isaca.org/resources/isaca-journal/issues/2022/volume-4/bias-and-ethical-concerns-in-machine-learning [Accessed 10 Feb. 2025].

[30] Y. A. Abid, J. Wu, G. Xu, S. Fu, and M. Waqas, "Multilevel Deep Neural Network Approach for Enhanced Distributed Denial-of-Service Attack Detection and Classification in Software-Defined Internet of Things Networks," *IEEE Internet of Things Journal* 11, no. 14 (2024): 24715–24725, https://doi.org/10.1109/jiot.2024.3376578.

[31] W. David, "*UNSW_NB15 Dataset*." 2019, https://www.kaggle.com. Available at: https://www.kaggle.com/datasets/mrwellsdavid/unsw-nb15 [Accessed 25 Feb. 2024].

[32] F. Chollet, *Deep Learning With Python* (Manning, Cop, 2018).

[33] R. Aziz, S. Banerjee, S. Bouzefrane, and T. Le Vinh, "Exploring Homomorphic Encryption and Differential Privacy Techniques Towards Secure Federated Learning Paradigm," *Future Internet* 15, no. 9 (2023): 310, https://doi.org/10.3390/fi15090310.

[34] S. An, J. Shin, and J. Kim, "Quantization-Aware Training With Dynamic and Static Pruning," *Access* 13 (2025): 57476–57484, https://doi.org/10.1109/access.2025.3556629.

[35] A. Zhang, Y. Li, W. Li, Y. Xie, Z. Zhang, and Z. Yang, "Lightweight NPU Method Based on Hyper-Threading Technology," in *Proceedings of the 2024 IEEE International Conference on Signal, Information and Data Processing (ICSIDP)* (IEEE, 2024), 1–5, https://doi.org/10.1109/icsidp62679.2024.10869015.

[36] T. Hulsen, "Explainable Artificial Intelligence (XAI): Concepts and Challenges in Healthcare," *AI* 4, no. 3 (2023): 652–666, https://doi.org/10.3390/ai4030034.

[37] A. Alwarafy, K. A. Al-Thelaya, M. Abdallah, J. Schneider, and M. Hamdi, "A Survey on Security and Privacy Issues in Edge-Computing-Assisted Internet of Things," *IEEE Internet of Things Journal* 8, no. 6 (2020): 4004–4022, https://doi.org/10.1109/jiot.2020.3015432.

[38] M. Standen, J. Kim, and C. Szabo, "Adversarial Machine Learning Attacks and Defences in Multi-Agent Reinforcement Learning," *ACM Computing Surveys* 57, no. 5 (2024): 1–35, https://doi.org/10.1145/3708320.

[39] S. K. Jagatheesaperumal, M. Rahouti, M. Aledhari, et al., "Distributed Reinforcement Learning for IoT Security in Heterogeneous and Distributed Networks," *Computing&AI Connect* 1, no. 1 (2024): 1–10, https://doi.org/10.69709/CAIC.2024.100109.

[40] M. Hussain, S. Pal, Z. Jadidi, E. Foo, and S. Kanhere, "Federated Zero Trust Architecture Using Artificial Intelligence," *IEEE Wireless Communications* 31, no. 2 (2024): 30–35, https://doi.org/10.1109/MWC.001.2300405.

[41] S. M. Islam, M. S. Bari, A. Sarkar, A. J. M. O. R. Khan, and R. Paul, "AI-Driven Threat Intelligence: Transforming Cybersecurity for Proactive Risk Management in Critical Sectors," *International Journal of Computer Science and Information Technology* 16, no. 5 (2024): 125–131, https://doi.org/10.5121/ijcsit.2024.16510.

[42] F. C. Andriulo, M. Fiore, M. Mongiello, E. Traversa, and V. Zizzo, "Edge Computing and Cloud Computing for Internet of Things: A Review," *Informatics* 11, no. 4 (2024): 71, https://doi.org/10.3390/informatics11040071.

[43] S. Almarri and A. Aljughaiman, "Blockchain Technology for IoT Security and Trust: A Comprehensive SLR," *Sustainability* 16, no. 23 (2024): 10177–10177, https://doi.org/10.3390/su162310177.

[44] J. N. Chukwunweike, M. Yussuf, O. Okusi, and T. Oluwatobi, "The Role of Deep Learning in Ensuring Privacy Integrity and Security: Applications in AI-Driven Cybersecurity Solutions," *World Journal of Advanced Research and Reviews* 23, no. 2 (2024): 1778–1790, https://doi.org/10.30574/wjarr.2024.23.2.2550.