network

MDPI

*Article*

# An Early-Exit Explainable Intrusion Prevention System with Federated Learning

**Saba Alaa Abdulwahhab [1], Qutaiba Alasad [2,*], Shahad Alahmed [1], Omer T. Khattab [3], Meaad Ahmed [1] and Jiann-Shuin Yuan [4]**

[1] Cybersecurity Department, Tikrit University, Tikrit 34001, Iraq; saba.programmer12@tu.edu.iq (S.A.A.); shahad.m.mustafa@tu.edu.iq (S.A.); myadahmdhsyn@gmail.com (M.A.)
[2] Departments of Cybersecurity and Petroleum Systems Control Engineering, Tikrit University, Tikrit 34001, Iraq
[3] College of Health and Medical Technologies-Al-Dour, Northern Technical University, Al-Dour 34002, Iraq; omar.kh@ntu.edu.iq
[4] Department of Electrical and Computer Engineering, University of Central Florida, Orlando, FL 32816-2450, USA; jiann-shiun.yuan@ucf.edu
* Correspondence: qutaibaeng@tu.edu.iq

## Abstract

Intrusion Prevention Systems (IPSs) are designed in order to detect and block malicious traffic with high accuracy, low latency, and limited resource use. Unfortunately, most learning-based solutions often depend on a single deep model or a hybrid cascade that can process every flow through the full pipeline, which is computationally expensive and offers little insight into why a decision was made. To overcome these issues, we propose EENet-IPS, an Early-Exit Explainable Intrusion Prevention System that combines multi-exit deep neural networks, federated training, and exit-specific explainability in a single architecture. A single backbone network exposes three exits (CNN, BiLSTM, and Transformer); confidently classified, simple traffic flows exit at the first exit in order to save time and computation, while ambiguous or complex flows are carefully routed deeper for closer analysis. Each exit has been paired with an exit-appropriate explanation technique, including SHAP, Integrated Gradients, and constrained counterfactuals, respectively, whose output is attached to the decision of a rule-based mitigation policy. To preserve data privacy across heterogeneous networks, the presented backbone is trained with federated learning (FedAvg) on the simulated non-IID clients that send clipped, noise-perturbed, and top-$k$-sparsified updates. We evaluate the proposed EENet-IPS on three public intrusion datasets— CSE-CIC-IDS2018, UNSW-NB15, and CICIoT2023—under both centralized and federated training, utilizing a leakage-controlled protocol (identifier removal, duplicate removal before splitting, and preprocessing fitted on the training data only) and three repeated runs. The routed system accomplished 99.98%, 95.78%, and 98.79% accuracy on the three datasets, with an AUC of at least 0.991. It resolves 73.9–99.9% of the flows at the first exit and lowers the mean single-flow CPU latency from 2.06–2.28 ms at full depth to 0.55–0.83 ms. In a single federated run (one client partition), the federated training stays within 1.75 percentage points of the centralized training while sending 5.0 times less uplink data than plain FedAvg. Among the recently related IPS/IDS studies that we compare with, none combines early-exit inference, federated training, and per-exit explanations, and the accuracy of EENet-IPS remains competitive with the reported results.

**Keywords:** cybersecurity; early-exit architectures; explainable AI; FL; IPS; multi-exit deep neural networks

Check for updates

Academic Editor: Bin Han

Received: 6 September 2026
Revised: 28 October 2026
Accepted: 3 November 2026
Published: 27 November 2026

**Copyright:** © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

## 1. Introduction

Nowadays, even though the increasing usage of Internet connections has provided unprecedented opportunities for human activities, it has also become subjected to many cyberattacks [1]. More specifically, intrusion has been considered as one of the most malicious attacks in many computing devices [2]. Due to the globalization, direct or indirect utilization of computers, it is very important to prevent different harmful activities and attacks that can be used to compromise computing infrastructures [3,4]. Cyberattacks have become more complex and include more sophisticated attacks, such as Advanced Persistent Threats (APTs), polymorphic malware, and zero-day attacks [5].

To mitigate such attacks, Intrusion Detection Systems (IDSs) have been developed and implemented, but, unfortunately, these cyber threats have also evolved with the technological developments. Recent studies have shown that depending only on IDSs is insufficient to obtain a secure system. IDS techniques are designed mainly to monitor network traffic and generate specific alerts when suspicious behavior is detected. However, they cannot prevent malicious actions. Consequently, the responses are reactive rather than proactive, which enables the threats to break the targeted system [6].

Many machine learning (ML) and deep learning (DL) techniques have been integrated with different IDSs to improve cybersecurity [7,8]. However, since they function as "black boxes", security experts and users cannot reveal the decision-making processes, and therefore they cannot interpret the underlying processing. Interestingly, the Explainable AI (XAI) can be integrated with intrusion prevention models to improve interpretability without reducing the performance of the model. However, these combined techniques cannot be used to prevent sophisticated attacks and incur high overhead.

In order to address these shortcomings, Intrusion Prevention Systems (IPSs) have incorporated critical security layers that are applied at earlier stages to detect and prohibit malicious activities before they compromise the system. Such proactive capability is especially important to prevent zero-day and sophisticated threats that are bypassed by traditional signature-based detection techniques [9,10]. ML and DL have also been integrated with IPS techniques to increase the security level of the system [11,12]. However, they require high computational overhead, provide limited interpretability, and are subject to various adversarial attacks [13,14]. Also, they fail to mitigate dynamic and sophisticated cyber adversaries [15]. In fact, IPS models have been boosted by Federated Learning (FL) to avoid sharing raw data and keep it private [16]. However, current IPS approaches also suffer from integrating them with explainability since the ML and DL models obscure decision-making, rendering the system difficult to be analyzed. Unfortunately, FL requires high communication overhead to handle data heterogeneity [17]. Also, the network traffic has exponentially increased, which adds significant challenges to the current IPS techniques in terms of preventing intrusions in real-time. This happens because most existing IPS techniques have utilized only a single mechanism, which limits the defensive design to prevent complex and advanced attacks. Moreover, they lack explanations in the systems and fail to obtain a protected system with low overhead and latency.

The early-exit (multi-exit) networks decrease the inference latency by letting confidently classified inputs leave at the intermediate classifiers [18]. However, their decisions remain opaque, which, in turn, makes them difficult to audit. Incorporating explainable smart approaches, such as SHAP and LIME, into IPS techniques significantly enhances operational confidence [19,20]. Although high-throughput networks and complex IoT systems prevent most IPS techniques from completing the process in milliseconds, these multi-exit models offer early-exit stages to reduce latency. The models preserve high performance, but they still cannot effectively protect sensitive data privacy [20,21]. To keep these sensitive data secure with high detection accuracy, FL can be employed [22,23]. To overcome these

challenges, we propose a secure design that is robust, generalizable, and resilient. Our main contributions are as follows:

1. We present a jointly latency-, privacy-, and explainability-aware multi-exit architecture. Unlike previous multi-exit IPS work, which can reduce the latency but leaves the decision process opaque, and previous FL-IDS work, which protects privacy but processes every flow through the full model, our proposed EENet-IPS combines a three-exit backbone (CNN, BiLSTM, and Transformer) with exit-native explanations and federated training in a single pipeline, so that the three properties are addressed together and their trade-offs are carefully measured.
2. We embed the exit-matched explainability rather than one-size-fits-all XAI. Instead of attaching a single and uniformly expensive explainer after the fact, we pair each exit with an explanation technique suited to its model class: KernelSHAP at the shallow CNN exit, Integrated Gradients at the recurrent BiLSTM exit, and constrained counterfactuals at the Transformer exit, and we measure the cost and quality of each explainer.
3. We have attached the explanations to the mitigation decisions. Each of the flagged flows is explained by the explainer of the exit that classified it, and the explanation is logged with the resulting action (block, quarantine, or escalation) for the analyst review and rule refinement.
4. We implement the Federated Collaborative Training (FCT) of the shared backbone across non-IID clients with partial participation. The clients send clipped, Gaussian-perturbed, and top-$k$-sparsified updates, so that the raw traffic never leaves its node of origin and the uplink traffic is decreased.
5. We design a confidence-based exit controller with temperature-calibrated thresholds and a mitigation feedback loop. The controller decides at the runtime whether a flow should be stopped early or be forwarded deeper, and a feedback loop records mitigation outcomes for use in subsequent federated training rounds.
6. The proposed framework has been evaluated under both centralized and federated training on three heterogeneous public datasets, with leakage controls, three repeated runs, ablations, classical and federated baseline techniques, and reported accuracy, calibration, exit-wise latency, explanation cost, and throughput under increasing flow volume. The evaluation in this paper is offline and dataset-based. It carefully measures detection, routing, and latency, whereas the mitigation layer is specified as a policy.

The remainder of the paper has been structured as follows: Previous related studies are explained in Section 2, the proposed EENet-IPS technique is illustrated in Section 3, the experimental setup and results are described in Section 4, the limitations and the plan for future work are discussed in Section 5, and finally, in Section 6, our conclusion is presented.

## 2. Related Works

### *2.1. Intrusion Prevention Systems (IPSs)*

In [9], the authors incorporated a single ML model into the Network Intrusion Detection Systems (NIDSs) to prevent serious intrusions, but this approach targets a narrow set of attacks. In [24], robust NIDS-based multiple ML algorithms were proposed to detect and prevent real-time attacks in high-speed 5G systems. This approach offers a high detection rate for complex traffic. However, it depends on a single CNN that can mainly capture spatial patterns, and every flow traverses the entire network, which elevates the processing cost on the resource-limited systems. Similarly, an online incremental learning IPS that integrated self-organizing incremental NNs with SVM classifiers has been developed to prevent new attacks in real time and during the normal operation, but at the penalty of requiring long latency [25]. The assessment was achieved on the outdated NSL-KDD dataset,

which does not include different sophisticated threats, and this method is not scalable to complex traffic flows in large-scale networks [25].

Recently, the ML techniques, including decision trees, random forests, and gradient boosting, have been integrated with an NIDS [26]. The feature selection approach and hyperparameter tuning were augmented to improve the detection rate, where the evaluation was done using the CSE-CIC-IDS2018 dataset. However, this technique targets detection only and depends mainly on the centralized training, which raises privacy concerns. Kumar et al. [27] optimized a support vector machine on fused IoT security datasets in order to separate normal from abnormal traffic, which clearly shows that the well-tuned classical models remain competitive, although they have been trained centrally and do not explain their decisions. In [28], the DL and ML models are hybridized to effectively detect anomalies in IDS for the IoT network. However, the latency reduction and privacy-preserving training have not been addressed.

### *2.2. DNN in Cybersecurity*

In [29], CNN, LSTM, and ANN models were used to expose intrusions in different static datasets, and the results indicated that the used models achieved high accuracy, approximately 96%. Such high accuracy ensured that DL models are good at capturing traffic patterns. However, since the experiments have been conducted on static datasets, the presented work cannot be generalized to dynamic and real-time network systems. Also, since no early decision-making techniques have been included, a full network propagation was used for every input sample, which in turn makes the execution time long. The design did not address data privacy since centralized data access was considered. The authors in [30] proposed a DL-based IDS tailored specifically for the IoT network to handle heterogeneous and unstructured IoT traffic flows by leveraging autoencoders in order to detect anomalies. However, each flow can pass through the full model, and the false-alarm rate is sensitive to the class imbalance in the training data.

### *2.3. FL in Information Security*

FL is a technique used to train ML models in a distributed way that is different from traditional centralized training. The end-users can take the advantage of the FL by not sharing their private and sensitive data with a central server [31]. It can protect data privacy by collaboratively training the intrusion detection model across distributed devices. In [32], Fed-MSE, a semi-supervised FL approach, was proposed by integrating autoencoders with one-class classifiers. The test was implemented using the N-BaIoT dataset, and the results indicated that the Fed-MSE achieved accuracy above 97%. However, every sample traverses the entire model, and the behavior under strongly non-IID data has not been examined. In [33], the NIDS-FGPA was presented leveraging FL with secure gradient and homomorphic encryption techniques. The 2D-CNN and BiGRU models have been combined to keep sensitive data confidential with high detection rates. This design offered a higher security level. However, it requires many global training rounds and does not really provide local blocking decisions or explanations. High data privacy and efficiency in the IoT and vehicular networks have been observed when augmenting FL with the intrusion detection technique. In [34], an optimal FL-based IDS was introduced for IoT applications. The proposal gained high accuracy by sharing the employed parameters, but it was tested only on limited datasets and does not utilize early exits. Alomari [35] combined only FL with differential-privacy-inspired noise for intrusion detection, which is the same type of the update perturbation adopted in EENet-IPS. FORT-IDS [36] targeted the federated, optimized, robust, and trustworthy intrusion detection for IIoT, considering the non-IID data and class imbalance, while FedTinyX-IDS [37] brings the federated, explainable

TinyML models to the intrusion detection in wireless sensor networks. Fatema et al. [38] combined FedAvg with SHAP explanations leveraging the CICIoT2023 dataset. EENet-IPS differs from these works by adding early-exit inference with exit-specific explanations to the federated training.

### *2.4. XAI in Cybersecurity*

Due to recent advances in AI systems employed in cybersecurity models, intrusions and attacks have become more complex and adaptive. Therefore, incorporating XAI into a cybersecurity system makes the system explainable, which enables the designers and defenders to understand and control the proposed defenses [39]. The XAI-IDS technique, including LIME and SHAP, was proposed to interpret the decision-making of multiple black-box ML models and then carefully detect the intrusions, which made the system trustworthy and transparent [19]. This work explained the contributing features and helped analyze the generated results, which improved the transparency of the alerts. However, the generated explanations required additional processing time, and they become available only after the full model has been evaluated. Similarly, XAI-XGBoost has been introduced to build an explainable IDS for medical IoT networks [40]. Their system utilized the XGBoost algorithm and considered only the most affected and important features with fixed imbalanced data to refine the model. In order to make the model's decisions clear, SHAP and LIME models have been augmented. However, the explanations are produced after the full model is assessed, and the model is trained centrally. In [41], the hybrid XAI-DL model was proposed to resolve real-time explainability, but augmenting the FL has not been taken into account. Separately, the authors in [42] proposed an XAI framework to resist serious cyber threats in the IoT system. While it achieved transparent anomaly detection, multi-exit mechanisms have not been incorporated to reduce system latency.

### *2.5. Challenges in Previous Work*

The works above fall into related but different groups. More specifically, IDSs detect suspicious traffic and raise alerts, whereas IPSs also map detections to the blocking actions; FL-based IDSs address data privacy, XAI-based IDSs address transparency, and early-exit networks [18] address the inference cost. Most of the current existing works address one or two of these aspects, and they often incur high computational and communication overhead, are difficult to interpret, or have not been evaluated on distributed nodes. To protect the private data, the FL allows the model to be trained locally and shares only the updated information rather than the entire raw data. Even though this technique is effective in protecting data privacy, it depends on complex deep learning models that operate inherently as black boxes and give limited interpretability. Also, the communication cost and long delay in FL can limit the model from being implemented in real-time applications [34,38,43–47]. In fact, XAI techniques help address the "black box" problem by explaining why the model makes such a decision, enabling the designer to build a trust system. However, only a few studies have combined the XAI with IDS and FL [38], and they do not reduce the per-flow inference cost [38,48]. Powerful DL models are great for quickly detecting complex attacks with high performance, but they are usually trained on centralized data and provide clearly limited explanations [49–52].

The proposed EENet-IPS can effectively address these challenges through implementing a three-exit neural network architecture. The early-implemented exit is used to prevent low-latency intrusions by making fast decisions on less complex traffic flows, and therefore decreases the computational costs and response times. Then, the FL technique is integrated to maintain data privacy during the distributed training with a smaller possible number of communication rounds. Finally, an exit-specific explainer is attached to each exit, providing

attribution-based explanations for the flows resolved early and counterfactual explanations for complex cases. This design targets low latency, privacy-aware training, and interpretable decisions within one pipeline.

Table 1 illustrates how our proposed EENet-IPS can address these aforementioned limitations. For each limitation, we have identified the representative studies that exhibit it and highlighted the corresponding EENet-IPS framework specifically designed in order to overcome it.

**Table 1.** Mapping the recurring limitations in previous IPS/IDS work to the corresponding EENet-IPS design elements.

| Recurring Limitation | Representative Studies | How EENet-IPS Addresses It |
|---|---|---|
| Full-pipeline inference for every flow, causing high latency/overhead | [9,24,29,30,32,34] | Three-exit backbone lets 73.9–99.9% of the flows exit after the shallow CNN stage (Section 4.3.1), so only ambiguous flows reach the deeper, costlier exits |
| Centralized data collection, compromising privacy and cross-site generalization | [9,28,29] | Federated Collaborative Training (FedAvg with clipped, Gaussian-perturbed, and top-*k*-sparsified updates) keeps raw traffic on the originating node (Section 3.3) |
| Explanations computed only post hoc, after the decision, or not at all | [9,24,25,28–30] | Exit-native SHAP/Integrated Gradients/counterfactual explanations are generated for each flagged flow by the explainer of its exit, and their cost is measured (Sections 3.2 and 4.3.2) |
| Explanations used only for offline analyst review, not tied to system behavior | [19,41,42,48] | Explanations are attached to each mitigation decision for analyst review and rule refinement (Section 3.4) |
| FL and explainability treated as separate, non-interacting concerns | [32–34,41] | FCT and exit-aware explainability are trained and evaluated within the same pipeline (Sections 3.3 and 4.3.3) |
| Evaluation limited to a single dataset or attack family | [25,32,34,48,49] | Evaluated on three heterogeneous datasets (enterprise, general-purpose, and IoT traffic) under both centralized and federated training, with leakage controls and repeated runs (Section 4) |

## 3. Our Proposed EENet-IPS Design

The framework of our proposed EENet-IPS is explained in Figure 1. The proposed system starts by processing the network traffic flows that are obtained from three different datasets, including CSE-CIC-IDS2018, UNSW-NB15, and CICIoT2023, in order to mimic different real intrusions. The system can operate on the flow records. In this study, we utilize the flow-level features released with each dataset, which the dataset providers computed from the raw packets with their flow meters. The packet-to-flow extraction is therefore not part of our measurements. These vectors contain sensitive information about the traffic flow, such as the statistical features (entropy measures, byte rate, total packets exchanged, and flow duration), the timing features (statistics of the inter-arrival times between the packets and of the burst patterns), and the network protocol features (HTTP, TCP/IP, flags, and DNS query types). These valuable features are extracted and applied to ensure that the model can effectively capture both flow behaviors and summaries of the packet-level information, which are important for distinguishing benign traffic from malicious traffic. The preprocessed data is then fed into the EENet multi-exit deep network system, which mainly comprises three exits designed to minimize both the prediction latency and the computational resources used. Note that the early exits make quick decisions on the known

or simple inputs, while the complex and sophisticated inputs are carefully performed at the next incoming exits, in which the execution is hierarchically processed, as follows:

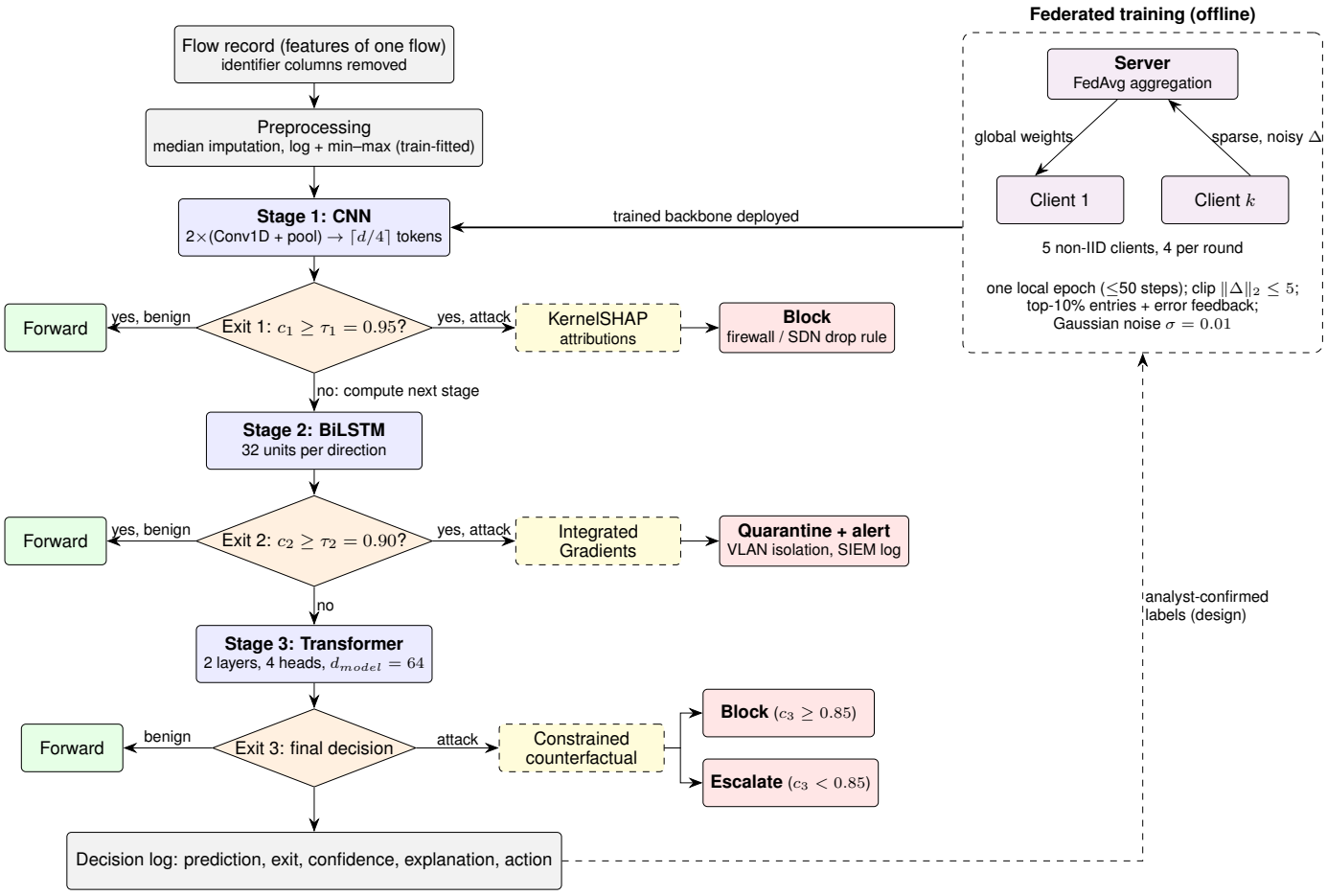



**Figure 1.** The proposed EENet-IPS architecture and per-flow decision path. A flow record passes through the CNN stage; Exit 1 accepts the decision if its calibrated confidence $c_1 = \max \text{softmax}(z_1/T_1)$ reaches $\tau_1$. Otherwise, the BiLSTM stage is computed, and so on. The flow flagged as malicious is explained by the explainer of the exit that classified it, and the policy maps the exit and confidence to an action. The backbone is trained offline with the federated learning.

The first exit is a simple CNN model that is used to quickly detect clear and high-confidence threats. It extracts spatial patterns from the statistical and protocol features and allows the extracted traffic flow patterns to exit early, which helps reduce latency and save resources used. If the confidence at the first exit is low, the traffic is forwarded to the second exit, which uses a Bidirectional Long Short-Term Memory (BiLSTM) model. This exit models the dependencies across the ordered feature tokens of a flow and is more suitable for detecting the stealthy or low-confidence attacks that are harder to identify. The third exit is the deepest layer, which includes a Transformer model with multi-head attention techniques. It has been designed specifically to analyze the complex or ambiguous traffic patterns that the earlier exits are not sufficiently confident in. This exit is mainly responsible for handling advanced and difficult attack types. Finally, an exit controller is implemented to dynamically decide whether the traffic should stop at the early exit or continue deeper to the next layers, and this depends on the temperature-calibrated confidence score of the current exit, as given in Section 3.1, where each exit has an associated explainer. The explanations are produced for the flows flagged as malicious and are attached to the mitigation decision for audit and rule refinement. We also incorporate the FL to train the EENet-IPS models

on multiple distributed locations, such as the edge devices and the cloud nodes, without moving or changing the raw traffic data. Each node trains the model leveraging its own local data and sends only the model updates to the given central server, where they are combined. This approach is important to keep the sensitive data private while allowing the system to learn from different network environments. Once the input traffic flow is classified at an exit, the EENet-IPS applies the mitigation action based on the confidence level and the explanation provided by the model. The clear and high-confidence threats are handled automatically, while for the uncertain traffic cases, the system will either trigger the alerts or send them for deeper analysis. The outcomes of these actions are recorded and then later used in order to refine the model during the future training, which in turn helps the system to be improved over the time. The implementation parts of our proposal are given in detail as in the following.

### *3.1. Backbone Architecture and Early-Exit Routing*

The CNN, BiLSTM, and Transformer form one sequential backbone, not three separate models, as reported in Table 2. The $d$ scaled features of a flow are read as a one-channel sequence of length $d$. The CNN stage reduces it to $L = \lceil d/4 \rceil$ tokens with 32 channels, which the BiLSTM stage and then the Transformer stage refine. This sequence is the ordered feature vector of a single flow record, not a temporal sequence of packets or flows. The BiLSTM and Transformer stages therefore model the dependencies among the feature tokens (i.e., among groups of flow features); they do not capture the temporal dynamics of the traffic. Each stage is followed by an exit, which mean-pools the tokens and applies a two-layer classifier. The three exits have been trained jointly with the average of their cross-entropy losses,

$$\mathcal{L} = \frac{1}{3}\sum_{k=1}^{3} \mathrm{CE}(z_k, y), \tag{1}$$

where $z_k$ are the logits of exit $k$ and $y$ is the label. After the training, a temperature $T_k$ is fitted for each exit on the validation split by minimizing the negative log-likelihood [53], and the confidence of exit $k$ is $c_k = \max \mathrm{softmax}(z_k/T_k)$. At the inference, a flow leaves at Exit 1 if $c_1 \geq \tau_1$, otherwise at Exit 2 if $c_2 \geq \tau_2$, and otherwise at Exit 3; the next stage is computed only for the flows that have not exited. We further utilize $\tau_1 = 0.95$ and $\tau_2 = 0.90$, i.e., the strict values that keep the early decisions conservative, and we report the validation sweep that supports this choice in Section 4.3.1. The value 0.85 at Exit 3 is not a routing threshold; the mitigation policy employs it in order to decide between blocking and escalation, as explained in Section 3.4.

**Table 2.** EENet-IPS backbone configuration (parameter counts for CSE-CIC-IDS2018, $d = 50$, $L = 13$).

| Block | Layers | Output | Parameters |
|---|---|---|---|
| Input | scaled flow features as a 1-channel feature sequence (not a time series) | $d \times 1$ | – |
| Stage 1 (CNN) | 2×[Conv1D(3), ReLU, MaxPool(2)], 32 ch., LayerNorm | $L \times 32$ | 3296 |
| Exit 1 head | mean pooling, FC(32)–ReLU–FC(2) | 2 | 1122 |
| Stage 2 (BiLSTM) | BiLSTM, 32 units per direction | $L \times 64$ | 16,896 |
| Exit 2 head | mean pooling, FC(32)–ReLU–FC(2) | 2 | 2146 |
| Stage 3 (Transformer) | learned positions + 2 encoder layers (4 heads, FFN 128, dropout 0.1) | $L \times 64$ | 67,776 |
| Exit 3 head | mean pooling, FC(32)–ReLU–FC(2) | 2 | 2146 |
| Total | | | 93,382 |

### *3.2. Exit-Aware Explainability*

The exit-aware explainability (EAE) is the core of our framework, in which each of the three exits generates specific explanations that can be easily interpreted by the humans. The explanations have been computed only for the flows flagged as malicious, by the explainer

of the exit that classified the flow, and they run alongside enforcement rather than on the routing path. Their measured cost is reported in Section 4.3.2.

The generated explanations state clearly why a flow has been flagged and stored with the mitigation decision, so that the analysts can verify the automatic actions and refine feature-based rules. In the evaluated configuration, routing utilizes the calibrated confidence only.

#### 3.2.1. Exit 1: SHapley Additive Explanations (SHAP)

We have applied the SHAP [54] at the shallow CNN exit, which is based on the cooperative game theory. The SHAP values, $\phi_i$ for the feature $i$, are computed as the average marginal contribution on all of the possible feature coalitions, and can be mathematically expressed as follows:

$$\phi_i = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!} [f(S \cup \{i\}) - f(S)] \quad (2)$$

where N is the feature set, S the subsets excluding i, and f the model's output. For efficiency, we use KernelSHAP, a model-agnostic approximation with 100 coalition samples, a background of 50 $k$-means centroids of the training data, and the ten most relevant features retained per explanation, focusing on global statistics, such as flow entropy and packet rates and protocol features, e.g., TCP flags. This lightweight variant highlights dominant contributors to high-confidence detections like volumetric DoS, enabling rapid analyst verification without full model retraining.

#### 3.2.2. Exit 2: Integrated Gradients

For the BiLSTM exit, which models the dependencies across the feature tokens of a flow, we utilize the Integrated Gradients (IGs) to be the explanation method [55]. The IG attributes are a gradient-based and differentiable approach that works well with the recurrent architectures, such as BiLSTM. In fact, it explains the prediction $F(x)$ by measuring how each input feature can be used to contribute to the output as the input changes from the baseline $x'$ to the real input $x$. The baseline $x'$ has been selected to be as a neutral benign flow, which is represented by a vector of zeros. The attribution has been calculated by integrating the gradients along a straight-line path between the $x'$ and the $x$, which captures the contribution of each input feature, including the timing-related flow statistics. The integrated gradient value for feature $i$ is given in Equation (3).

$$IG_i(x) = (x_i - x_i') \times \int_{\alpha=0}^{1} \frac{\partial F(x' + \alpha(x - x'))}{\partial x_i} d\alpha \quad (3)$$

Practically, such integration has been estimated by leveraging the Riemann summation with 256 steps (trapezoidal rule). The IG is employed to reveal which flow features drive a decision, such as the inter-arrival-time and burst statistics and the forward/backward asymmetry, which can indicate the stealthy reconnaissance. IG satisfies the completeness property: the sum of all feature attributions is equal to the difference between the model's output for the input and for the baseline technique. We use the deviation from this property, together with the measured cost, in order to assess the explainer in Section 4.3.2.

#### 3.2.3. Exit 3: Constrained Counterfactual Explanations

At the Transformer exit, we utilize the counterfactual explanations in order to create minimized perturbations, $x'$, that can be used to flip the prediction of the model, e.g., from the benign to the attack. These changes have been constrained so that they stay valid for the real network behavior, such as validating the protocol states and the packet sizes [56]. This has been solved leveraging the following optimization equation:

$$x' = \arg\min_{x'} d(x, x') \quad \text{s.t.} \quad F(x') \neq F(x), \quad x' \in C \tag{4}$$

where the $d$ is the distance metric, e.g., L1 with the sparsity promotion, and the $C$ is used to make sure that the generated input stays valid according to the domain rules. In our implementation, $C$ keeps every feature inside its training range ($[0, 1]$ after scaling) and freezes the one-hot protocol, service, and state indicators. The problem is solved by utilizing an L1-regularized gradient search on the logit margin with at most 100 Adam steps, which stops once the predicted class flips, followed by a greedy pass that undoes the smallest changes as long as the class remains flipped. These counterfactuals provide more information about the ambiguous flows that are hard to be classified.

These generated explanations refine the operational trust by exposing the decision rationales, such as providing the summary plots and the heatmaps, and they support the analyst review of the escalated flows. Note that none of the explainers modifies the classifier, so the explanations do not really change the predictions or the accuracy reported in Section 4; their cost is reported next to the inference latency in Section 4.3.2.

### *3.3. FL Incorporation*

The EENet-IPS uses the federated collaborative training to allow the system to refine its model on different locations without the need for sharing the sensitive and private data. The basic process works as follows: a central server, such as a secure cloud setup, is first initialized as the base model. The individual nodes, e.g., the company branch offices, edge routers, and cloud servers, then perform the local training on this model by using their own confidential network traffic data. For this local training, they run Adam on the multi-exit loss in Equation (1) for one local epoch per round, capped at 50 steps (learning rate $\eta = 2 \times 10^{-3}$, batch size 512). These nodes only send the updated model parameters (the adjusted weights or calculated gradients) back to the main server. The original raw data will never leave its location. At the central server, all of the incoming updates are combined using Federated Averaging (FedAvg) [57] as shown in the following equation:

$$w^{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} w_k^{t+1} \tag{5}$$

where the $w^{(t+1)}$ is the updated global model at the round $t+1$, $K$ is the number of clients participating in the round, $n_k$ represents the local dataset size for the client $k$, and $n = \sum_{k=1}^{K} n_k$. In practice, each client sends its update $\Delta_k = w_k^{t+1} - w^t$, which the server averages with the same weights. In order to limit what an individual update reveals, each client clips $\Delta_k$ to an $\ell_2$ norm of $C = 5$, keeps only the 10% of its entries with the largest magnitude (accumulating the rest locally for later rounds, i.e., error feedback [58]), and adds the Gaussian noise $\mathcal{N}(0, \sigma^2)$ with $\sigma = 0.01$ to the transmitted values [35]. In fact, this perturbation is inspired by differential privacy, but we do not claim a formal $(\varepsilon, \delta)$ guarantee. The secure aggregation, homomorphic encryption, and robustness to malicious clients have not been implemented in this study and are left for future work.

We simulate $K_{\text{tot}} = 5$ clients. The training split has been divided among them with a Dirichlet ($\alpha = 0.5$) label split to obtain non-IID local data, and four of the five clients are sampled in each round to emulate intermittent availability. On the UNSW-NB15 dataset, each client applies the SMOTE-Tomek to its own data. On the CSE-CIC-IDS2018 and CICIoT2023 datasets, where some clients hold almost only benign or almost only malicious flows, each client instead adds the logarithm of its local class prior to the logits during training (logit adjustment [59]). In both cases, the server incorporates the logarithm of the class prior of the validation split into the exit logits of the aggregated model. The training runs for 15 rounds on the CSE-CIC-IDS2018 and CICIoT2023 datasets, and for 60 rounds on

the UNSW-NB15, whose validation accuracy rises more slowly; as in the centralized training, the global model with the lowest validation loss is kept. The uplink cost per participating client and round is $4P$ bytes for dense updates and $8\lfloor 0.1P \rfloor$ bytes for top-$k$ updates (a 32-bit value and a 32-bit index per entry), where $P \approx 93$ k is the number of model parameters. We compare this configuration (denoted EENet-FL) with the plain FedAvg (dense updates without clipping or noise) and with the FedProx [60] ($\mu = 10^{-3}$).

### *3.4. Mitigation Policy*

When the classification decision is performed at an exit, the proposed EENet-IPS will map the exit level, the confidence score, and the associated explanation to the predefined mitigation actions in order to ensure the right proactive threat response. This is achieved by leveraging the rule-based policy that applies the confidence thresholds (calibrated with the temperature scaling to decrease the calibration error significantly); the explanation is attached to each decision for analyst review but does not change the action. Below are practical cases:

At exit 1, when the traffic is classified as malicious with very high confidence (>95%), the system will immediately block the infected flow. The SHAP explanations will expose the main malicious features, e.g., the entropy spikes, and the block is enforced by the mitigation techniques, such as the IP blacklisting or the packet dropping through using the firewalls or the SDN controllers. This exit is designed to target the simple and obvious threats, including the DoS floods, and it is used as a high confidence threshold to limit the FPs.

At exit 2, when a flow that did not reach $\tau_1$ is classified as malicious with a confidence of at least $\tau_2 = 0.90$, the system raises an alert and temporarily quarantines the suspicious IPs or sessions. The IG explanations will reveal the features that drive the decision, e.g., the irregular inter-arrival-time statistics, and are logged with the alert to the SIEM systems, e.g., Splunk, while the quarantine is enforced by the temporary (short-term) isolation, e.g., VLAN segregation. This exit is used to balance the caution for stealthy attacks with the lower possibility of the benign traffic.

At exit 3, a flow classified as malicious with a confidence of at least 0.85 is blocked, whereas a malicious decision with a lower confidence is escalated for manual review. In fact, the counterfactual explanations will provide hypothetical changes, e.g., a lower packet rate that will lead to a benign classification, which will offer assistance for the manual analysis via use of the dashboards. This exit is designed to handle complex, ambiguous, and unseen attacks.

Finally, the feedback loop will record the analyzed corrections, e.g., the label refinements, and incorporate them into the future FL round in order to fine-tune the model via utilization of the semi-supervised updates. All of the mitigation actions are logged and traceable, with the explanations that are stored in the tamper-evident logs to boost the compliance, the investigation, and the post-incident analysis.

It is worth mentioning that the intended deployment is an inline gateway or SDN controller at the network edge, where blocking and quarantine map to the firewall rules or to the installation of SDN flow rules. In this study, the policy decisions have been computed and logged for each test flow. However, the actions are not enforced on live traffic, and the feedback loop is a design element that is not evaluated. Therefore, the mitigation delay (rule installation time) is not measured, and we only report the detection and decision latency in Section 4.

## 4. Results

### *4.1. Experimental Setup*

All of the experiments have been run on a 2-vCPU x86-64 Linux virtual machine with 32GB of RAM, utilizing Python 3.13 and PyTorch 2.14. Each training job leveraged one CPU thread, and all of the timings have been taken with one thread on an otherwise idle machine, which approximates a single-core edge gateway. The training employed Adam ($\eta = 2 \times 10^{-3}$, batch size 512), the multi-exit loss in Equation (1), a ReduceLROnPlateau scheduler (factor 0.5, patience 1), and early stopping on the validation loss with a patience of 4 epochs and at most 25 epochs for the CSE-CIC-IDS2018 and CICIoT2023 datasets. The UNSW-NB15 dataset has about ten times fewer training flows and a small, noisier validation split, so it utilizes a patience of 8 and at most 60 epochs. On the UNSW-NB15, the batch size and learning rate have been selected on the validation split from {64, 128, 256, 512} and $\{1, 2\}\times10^{-3}$ (batch size 128, $\eta = 2 \times 10^{-3}$), and each ablation model leverages whichever of the batch sizes 128 and 512 gives the higher validation accuracy. Every centralized EENet-IPS configuration and every classical baseline technique has been trained three times (seeds 1–3) on the same data split, and we report the mean ± standard deviation over these runs. The federated configurations and the ablation models are run once (seed 1); for the federated runs, this seed fixes both the partition of the training data among the clients and the training randomness. The code and the run logs utilized to generate each table and figure are publicly available at https://github.com/QutaibaEng/EENet-IPS (accessed on 23 October 2026).

The performance of the proposal has been evaluated using the Expected Calibration Error (ECE), AUC, Precision, F1-score, Recall, False Positive Rate (FPR), Accuracy, the confusion matrix, the average inference latency, and the FL uplink communication cost, with malicious flows as the positive class. The single-flow latency is the time of one forward pass with batch size 1, averaged over 1000 random test flows after 100 warm-up runs. It is reported cumulatively for each exit path (Exit 1: CNN stage and head; Exit 2: plus BiLSTM stage and head; Exit 3: full depth) and for the routed system, which also includes the confidence tests and the policy decision. It excludes the flow-feature extraction, which the dataset flow meters performed, and explanation generation, which has been reported separately. The batch throughput is measured by routing the whole test split in batches of 1024 flows.

### *4.2. Datasets Preprocessing*

We employ the Wednesday 14-02-2018 capture of the CSE-CIC-IDS2018 dataset [61] (benign, FTP-BruteForce, and SSH-Bruteforce flows), the official testing partition of the UNSW-NB15 dataset [62], and the file Merged01 of the current CICIoT2023 dataset release [63] (33 attack classes and benign traffic, 39 features). All tasks are binary (benign vs. malicious); the original attack labels are kept only for the per-family analysis. Each of the three datasets has been standardized and preprocessed as follows.

First, the identifier-like columns have been removed: the Timestamp and Dst Port in CSE-CIC-IDS2018, and the record id in the UNSW-NB15, whose attack-category column is utilized only as the label. None of the three files contains IP addresses or flow identifiers, so IP addresses are not model inputs; the 5-tuple is leveraged only by the dataset providers to form the flows. Second, the exact duplicate flows have been removed before splitting, together with the few feature vectors that occur with both labels, so that no identical flow appears in both of the training and the test data. This removes 41%, 34%, and 27% of the records, respectively. Third, each dataset has been split once into stratified 80% training and 20% test partitions (seed 42), and 10% of the training partition is held out for validation purposes.

All of the following statistics are fitted on the training partition only and then applied unchanged to the validation and test partitions. The missing and infinite values have been replaced by the training median. The constant features and one feature of each pair with a correlation above 0.99 are removed. Non-negative features are log-transformed, and all of the features are min–max scaled to $[0, 1]$, with values outside the training range clipped. The categorical proto, service, and state fields of UNSW-NB15 and the protocol field of CSE-CIC-IDS2018 are one-hot encoded, and categories that cover less than 0.5% of the training flows are pooled into an "other" level. The PCA technique is not really utilized, so the models and the explanations operate on the named features. Finally, the SMOTE-Tomek technique has been applied to the training partition of UNSW-NB15 only, and never to the validation or test data. Because it balances the classes, the logarithm of the ratio between the class priors of the validation split and of the resampled training data is added to the exit logits after training, which restores the class prior of the classified traffic [64]. Table 3 explains in detail the three datasets before and after applying the cleaning, where all have two classes (benign and malicious).

**Table 3.** Datasets before and after cleaning (binary labels). Columns: raw columns including the label/input features after preprocessing.

| Dataset | Records | After De-Duplication | Columns/Features | Train/Validation/Test | Malicious (%) |
|---|---|---|---|---|---|
| CSE-CIC-IDS2018 | 1,048,575 | 619,798 | 80/50 | 446,254/49,584/123,960 | 15.2 |
| UNSW-NB15 | 82,332 | 53,940 | 45/53 | 38,836/4316/10,788 | 36.6 |
| CICIoT2023 | 712,311 | 518,091 | 40/33 | 373,024/41,448/103,619 | 96.8 |

### *4.3. Experimental Results*

In this subsection, we have analyzed and evaluated our proposed EENet-IPS based on the security and performance metrics using the three preprocessed datasets and compared the results with different related works. The main goal is to show how the multi-exit approach can process the straightforward flows quickly at the early exit and forward the complex ones to upper levels without losing accuracy. All of the tables and figures in this section have been generated from the same runs.

#### 4.3.1. Experiment I: Centralized Training and Early-Exit Routing

Table 4 illustrates the routed system on the test partition, and Table 5 gives the confusion matrices of the first run. The routed EENet-IPS accomplishes 99.98%, 95.78%, and 98.79% accuracy, with FPRs of 0.00%, 2.50%, and 14.21%, and the ECE stays at or below 0.005 after temperature scaling. The standard deviations over the three runs are small. After the de-duplication, the brute-force flows of the CSE-CIC-IDS2018 dataset are still almost perfectly separable, while the UNSW-NB15 dataset is the hardest dataset. Since the CICIoT2023 dataset is 96.8% malicious, its FPR on the small benign class (14.21%) is more informative than its accuracy. This FPR corresponds to the default operating point (attack probability $\geq 0.5$); an operator can lower it by raising the decision threshold at the cost of Recall, as the AUC of 0.9964 indicates.

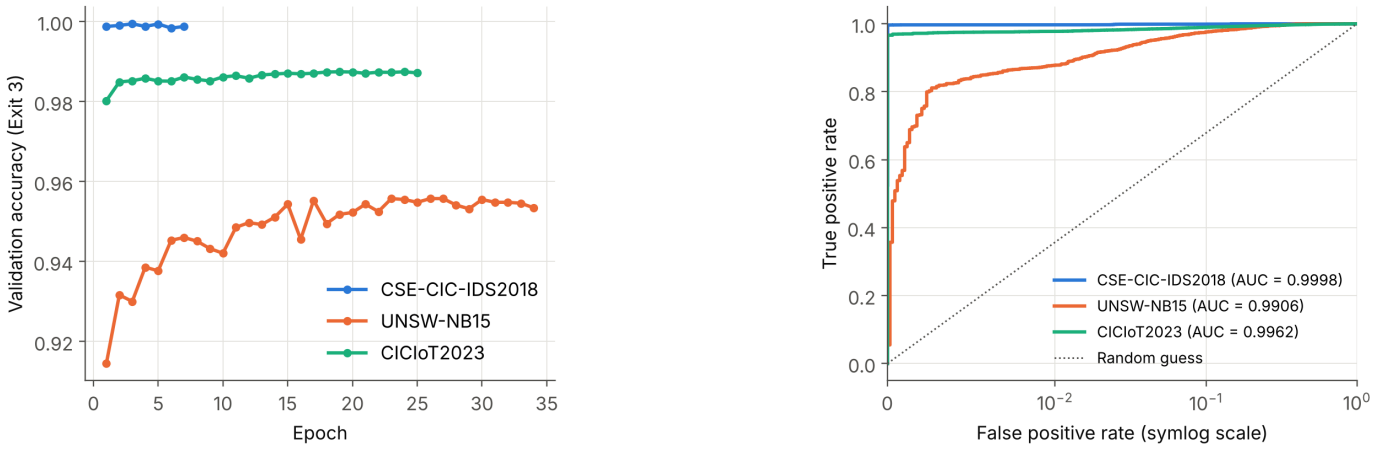
**Table 4.** Performance of the routed EENet-IPS on the test partitions (centralized training, mean ± standard deviation over three runs).

| Dataset | Accuracy (%) | Precision | Recall | F1-Score | AUC | FPR (%) | ECE |
|---|---|---|---|---|---|---|---|
| CSE-CIC-IDS2018 | 99.98 ± 0.03 | 0.9999 ± 0.0001 | 0.9986 ± 0.0019 | 0.9993 ± 0.0010 | 0.9999 ± 0.0001 | 0.00 ± 0.00 | 0.0000 ± 0.0000 |
| UNSW-NB15 | 95.78 ± 0.45 | 0.9553 ± 0.0051 | 0.9281 ± 0.0104 | 0.9415 ± 0.0064 | 0.9912 ± 0.0017 | 2.50 ± 0.29 | 0.0044 ± 0.0006 |
| CICIoT2023 | 98.79 ± 0.06 | 0.9953 ± 0.0004 | 0.9922 ± 0.0009 | 0.9937 ± 0.0003 | 0.9964 ± 0.0002 | 14.21 ± 1.22 | 0.0009 ± 0.0003 |

**Table 5.** The confusion matrices of the routed EENet-IPS on the test partitions (first run).

| Dataset | TN | FP | FN | TP | FPR (%) | FNR (%) |
|---|---|---|---|---|---|---|
| CSE-CIC-IDS2018 | 105,139 | 2 | 67 | 18,752 | 0.00 | 0.36 |
| UNSW-NB15 | 6646 | 194 | 281 | 3667 | 2.84 | 7.12 |
| CICIoT2023 | 2805 | 505 | 775 | 99,534 | 15.26 | 0.77 |

Figure 2 demonstrates the validation accuracy per epoch and the ROC curves, and Figure 3 shows that the training and validation losses converge without a widening gap. Table 6 explains the routing in detail. On average, 99.9%, 73.9%, and 94.5% of the test flows leave at Exit 1, and the accuracy on these flows is high because only flows with $c_1 \geq 0.95$ exit there. The flows that reach Exit 3 are the hardest ones, which really explains the lower accuracy on that row (on CSE-CIC-IDS2018, only about five test flows per run reach it). The malicious decisions escalated to an analyst ($c_3 < 0.85$) are 0.00%, 3.53%, and 1.44% of the test flows. Figure 4a shows the calibrated confidence of each exit for the flows that reach it, which is consistent with the Exit-1 shares in Table 6. Note that since most of the flows stop after the CNN stage, the mean single-flow latency of the routed system (0.554, 0.832, and 0.615 ms) stays well below that of the full-depth path (2.278, 2.057, and 2.104 ms). With batches of 1024 flows, the routed model processes 56,195, 21,013, and 49,194 flows per second on one thread. Scaling an already extracted flow record adds about 25.6 µs.

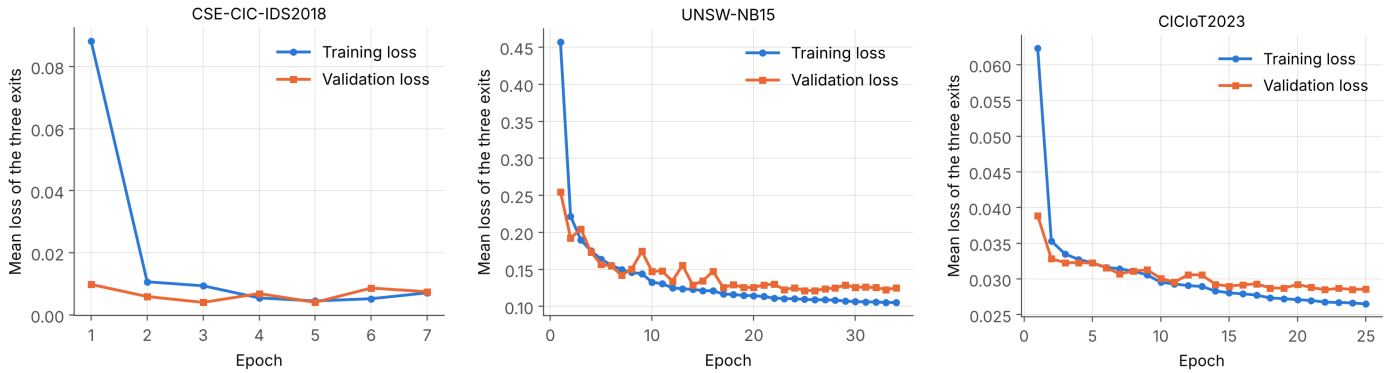



(**a**) Validation accuracy (Exit 3) per epoch

(**b**) ROC curves of the routed system

**Figure 2.** Performance of EENet-IPS (first run): (**a**) validation accuracy of Exit 3 during training and (**b**) ROC curves of the routed system on the test partitions, with a logarithmic false-positive-rate axis; the AUC values in the legend are those of this run, whereas Table 4 gives the mean over the three runs.

The thresholds have been checked on the validation partition (first run) for all of the pairs of $\tau_1$ and $\tau_2$ from 0.80, 0.85, 0.90, 0.95, and 0.99. On UNSW-NB15, with $\tau_2 = 0.90$, the validation accuracy/Exit-1 share is $\tau_1 = 0.80$: 95.53%/88.5%; $\tau_1 = 0.85$: 95.67%/85.9%; $\tau_1 = 0.90$: 95.81%/81.3%; $\tau_1 = 0.95$: 95.85%/72.3%; $\tau_1 = 0.99$: 95.90%/55.0%. Looser thresholds therefore move more flows to Exit 1 at the cost of accuracy: with $\tau_1 = \tau_2 = 0.80$, the accuracy falls 0.44 percentage points below full depth, against 0.12 points for the chosen $\tau_1 = 0.95$ and $\tau_2 = 0.90$. On the CSE-CIC-IDS2018 and CICIoT2023 datasets, each threshold pair of the sweep stays within 0.01 percentage points of full depth, so the strict values cost almost nothing there.

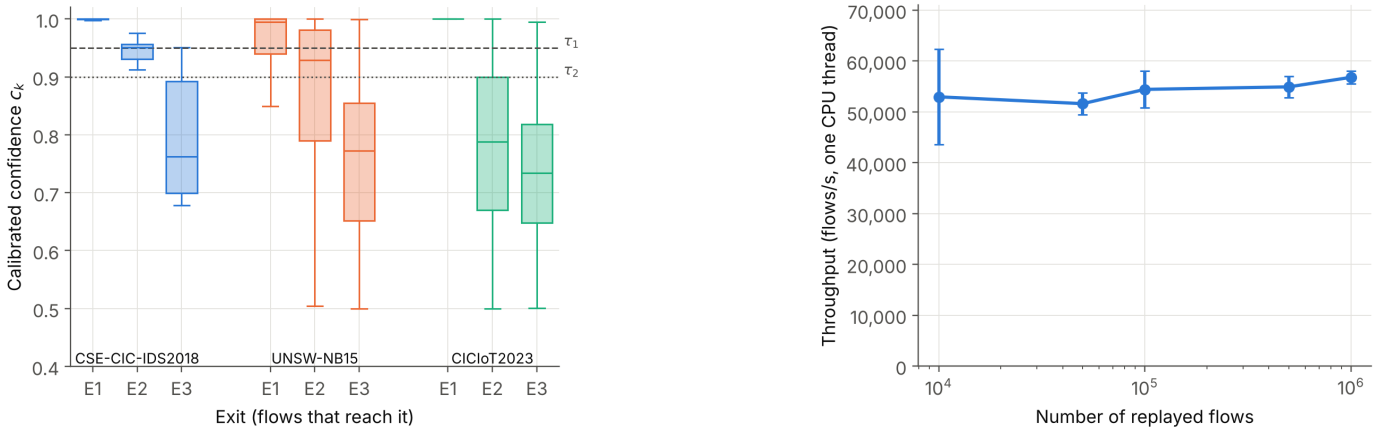



(**a**) CSE-CIC-IDS2018 (**b**) UNSW-NB15 (**c**) CICIoT2023

**Figure 3.** Training and validation loss (mean loss of the three exits) per epoch for (**a**) CSE-CIC-IDS2018, (**b**) UNSW-NB15, and (**c**) CICIoT2023 (first run).

**Table 6.** The early-exit routing on the test partitions: share of the flows leaving at each exit, accuracy on those flows (mean ± standard deviation over three runs), and single-flow CPU latency of each exit path (first run, one thread).

| Dataset | Exit | Flows (%) | Accuracy on These Flows (%) | Latency (ms/Flow) |
|---|---|---|---|---|
| CSE-CIC-IDS2018 | Exit 1 | 99.9 ± 0.0 | 99.98 ± 0.02 | 0.384 |
| | Exit 2 | 0.1 ± 0.0 | 96.54 ± 5.15 | 1.201 |
| | Exit 3 | 0.0 ± 0.0 | 79.63 ± 26.25 | 2.278 |
| | Routed (all flows) | 100 | 99.98 ± 0.03 | 0.554 |
| UNSW-NB15 | Exit 1 | 73.9 ± 1.7 | 99.32 ± 0.22 | 0.343 |
| | Exit 2 | 14.3 ± 1.7 | 96.02 ± 0.22 | 1.072 |
| | Exit 3 | 11.7 ± 0.2 | 73.20 ± 2.10 | 2.057 |
| | Routed (all flows) | 100 | 95.78 ± 0.45 | 0.832 |
| CICIoT2023 | Exit 1 | 94.5 ± 0.1 | 99.98 ± 0.01 | 0.358 |
| | Exit 2 | 1.4 ± 0.1 | 94.04 ± 0.68 | 1.109 |
| | Exit 3 | 4.1 ± 0.1 | 73.17 ± 1.75 | 2.104 |
| | Routed (all flows) | 100 | 98.79 ± 0.06 | 0.615 |




(**a**) Calibrated confidence per exit (**b**) Throughput under increasing flow volume

**Figure 4.** (**a**) Calibrated confidence $c_k$ of each exit for the test flows that reach it (first run; the dashed and dotted lines mark $\tau_1$ and $\tau_2$). (**b**) Throughput of the routed model when $10^4$–$10^6$ CSE-CIC-IDS2018 test flows are replayed (mean ± standard deviation over three repetitions).

In order to examine the behavior under increasing load, we have replayed $N \in \{10^4, \ldots, 10^6\}$ test flows in the CSE-CIC-IDS2018 dataset, sampled with replacement, through the routed model. A single worker process (one CPU thread, no concurrency) fed them in

micro-batches of 1024, and each run has been repeated three times, as shown in Table 7 and Figure 4b. The throughput stays between 42,119 and 59,164 flows per second, the memory is still bounded, and the accuracy and the Exit-1 share do not change with $N$, since the replay resamples the same test distribution. These numbers characterize the computational capacity of one core. The queueing and live-traffic effects are not modeled and are discussed in Section 5.

**Table 7.** Replay of increasing numbers of CSE-CIC-IDS2018 test flows through the routed model (one CPU thread, micro-batches of 1024, mean ± standard deviation over three repetitions).

| Flows | Throughput (Flows/s) | Latency (ms/Flow) | Peak Memory (MB) | Accuracy (%) | Exit-1 Share (%) |
|---|---|---|---|---|---|
| 10,000 | 52,944 ± 9409 | 0.0193 ± 0.0038 | 816 ± 10 | 99.91 ± 0.02 | 99.9 ± 0.0 |
| 50,000 | 51,600 ± 2186 | 0.0194 ± 0.0008 | 816 ± 10 | 99.94 ± 0.01 | 99.9 ± 0.0 |
| 100,000 | 54,406 ± 3635 | 0.0184 ± 0.0012 | 819 ± 6 | 99.95 ± 0.01 | 99.9 ± 0.0 |
| 500,000 | 54,904 ± 2083 | 0.0182 ± 0.0007 | 820 ± 3 | 99.95 ± 0.00 | 99.9 ± 0.0 |
| 1,000,000 | 56,776 ± 1244 | 0.0176 ± 0.0004 | 822 ± 0 | 99.95 ± 0.00 | 99.9 ± 0.0 |

#### 4.3.2. Experiment II: Exit Contributions, Ablation Studies, and Explanations

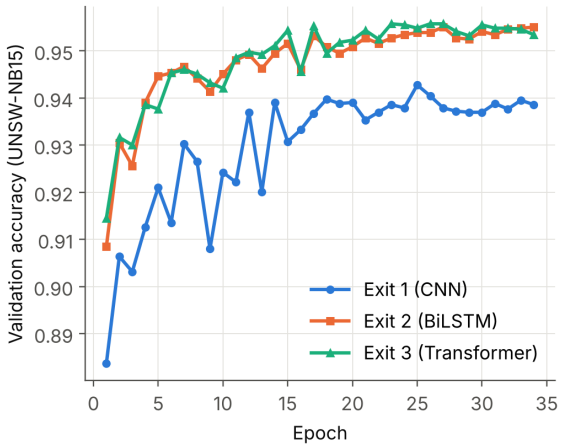
In this experiment, we have analyzed the contribution of each early-exit classifier in the proposed architecture. Unlike Table 6, Table 8 shows the evaluation on each exit on all of the test flows. On the UNSW-NB15, the accuracy increases from Exit 1 (94.23%) to Exits 2 and 3 (95.76% and 95.76%). On the CSE-CIC-IDS2018 and CICIoT2023, all of the three exits are close to saturation, which is why most flows can safely leave at Exit 1. Figure 5a shows that the deeper exits reach a higher validation accuracy within a few epochs, whereas Exit 1 stays lower and fluctuates more, and Figure 5b compares the exits across datasets.

**Table 8.** Each exit evaluated on all of the test flows (centralized training, mean ± standard deviation over three runs).

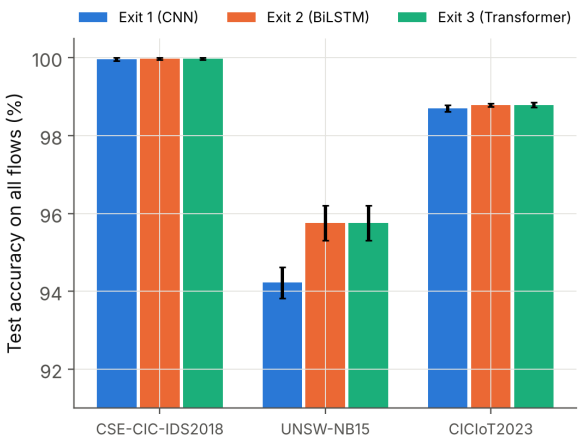
| Dataset | Exit | Accuracy (%) | Precision | Recall | F1-Score | AUC |
|---|---|---|---|---|---|---|
| CSE-CIC-IDS2018 | Exit 1 (CNN) | 99.97 ± 0.04 | 0.9993 ± 0.0010 | 0.9985 ± 0.0016 | 0.9989 ± 0.0013 | 0.9999 ± 0.0001 |
| | Exit 2 (BiLSTM) | 99.98 ± 0.03 | 0.9999 ± 0.0000 | 0.9986 ± 0.0019 | 0.9993 ± 0.0010 | 1.0000 ± 0.0001 |
| | Exit 3 (Transformer) | 99.98 ± 0.03 | 0.9999 ± 0.0001 | 0.9987 ± 0.0019 | 0.9993 ± 0.0010 | 0.9999 ± 0.0001 |
| UNSW-NB15 | Exit 1 (CNN) | 94.23 ± 0.40 | 0.9264 ± 0.0051 | 0.9149 ± 0.0068 | 0.9206 ± 0.0056 | 0.9866 ± 0.0022 |
| | Exit 2 (BiLSTM) | 95.76 ± 0.45 | 0.9552 ± 0.0021 | 0.9276 ± 0.0111 | 0.9412 ± 0.0065 | 0.9917 ± 0.0015 |
| | Exit 3 (Transformer) | 95.76 ± 0.45 | 0.9551 ± 0.0045 | 0.9276 ± 0.0110 | 0.9412 ± 0.0065 | 0.9920 ± 0.0011 |
| CICIoT2023 | Exit 1 (CNN) | 98.70 ± 0.08 | 0.9946 ± 0.0003 | 0.9920 ± 0.0006 | 0.9933 ± 0.0004 | 0.9958 ± 0.0004 |
| | Exit 2 (BiLSTM) | 98.78 ± 0.04 | 0.9952 ± 0.0004 | 0.9922 ± 0.0006 | 0.9937 ± 0.0002 | 0.9964 ± 0.0002 |
| | Exit 3 (Transformer) | 98.79 ± 0.06 | 0.9953 ± 0.0004 | 0.9922 ± 0.0009 | 0.9937 ± 0.0003 | 0.9964 ± 0.0002 |

Table 9 compares EENet-IPS with classical models trained under the same protocol and with deep ablations: each stage utilized alone, and the same backbone trained and used without early exits. The backbone without exits is the full-depth reference. The routed EENet-IPS is at least as accurate as this reference on all of the three datasets while computing the BiLSTM and Transformer stages for only a minority of the flows. The variant without FL corresponds to the centralized rows, and FL without the privacy mechanisms corresponds to plain FedAvg in Table 10. The traditional and classical models are strong on these tabular features, as previous work has also reported [27], and gradient boosting is the most accurate model on all of the three datasets, 0.02–0.86 percentage points above EENet-IPS. On the CICIoT2023 dataset, the false-alarm rate of EENet-IPS (14.21 ± 1.22%) is similar to that of gradient boosting (14.76 ± 0.73%), which detects slightly more attacks (99.41% against 99.22%), and lower than that of the random forest (16.83 ± 0.15%). This rate is high for each model; only 3.2% of the CICIoT2023 test flows are benign. On the UNSW-NB15 dataset, the tree ensembles raise fewer false alarms (1.82–1.89% against 2.50%).

The graded policy in Section 3.4 limits the cost of a false alarm: only 0.40% (UNSW-NB15) and 0.98% (CICIoT2023) of the benign flows are blocked automatically, whereas the other flagged benign flows are quarantined or escalated to an analyst.




(**a**) Per-exit validation accuracy (UNSW-NB15)




(**b**) Per-exit test accuracy

**Figure 5.** Accuracy analysis: (**a**) validation accuracy of each exit per epoch on UNSW-NB15 (first run); (**b**) test accuracy of each exit on all flows (mean ± standard deviation over three runs).

**Table 9.** Ablation studies and baseline techniques on the test partitions (mean over three runs for EENet-IPS and the classical models; seed 1 for the deep ablations).

| Dataset | Model | Accuracy (%) | F1-Score | AUC | FPR (%) |
|---|---|---|---|---|---|
| CSE-CIC-IDS2018 | Logistic regression | 99.89 | 0.9962 | 1.0000 | 0.09 |
| | Random forest | 100.00 | 0.9999 | 1.0000 | 0.00 |
| | Gradient boosting | 100.00 | 1.0000 | 1.0000 | 0.00 |
| | CNN only | 99.98 | 0.9993 | 1.0000 | 0.01 |
| | BiLSTM only | 99.99 | 0.9997 | 1.0000 | 0.00 |
| | Transformer only | 99.99 | 0.9995 | 1.0000 | 0.00 |
| | Backbone without early exits | 99.96 | 0.9986 | 1.0000 | 0.05 |
| | EENet-IPS (routed, early exits) | 99.98 | 0.9993 | 0.9999 | 0.00 |
| UNSW-NB15 | Logistic regression | 92.36 | 0.8958 | 0.9726 | 6.12 |
| | Random forest | 96.09 | 0.9454 | 0.9935 | 1.82 |
| | Gradient boosting | 96.64 | 0.9534 | 0.9949 | 1.89 |
| | CNN only | 94.25 | 0.9212 | 0.9869 | 4.31 |
| | BiLSTM only | 94.59 | 0.9250 | 0.9882 | 3.45 |
| | Transformer only | 95.12 | 0.9320 | 0.9898 | 2.69 |
| | Backbone without early exits | 95.11 | 0.9324 | 0.9903 | 3.11 |
| | EENet-IPS (routed, early exits) | 95.78 | 0.9415 | 0.9912 | 2.50 |
| CICIoT2023 | Logistic regression | 98.22 | 0.9908 | 0.9929 | 25.65 |
| | Random forest | 98.95 | 0.9946 | 0.9969 | 16.83 |
| | Gradient boosting | 98.96 | 0.9946 | 0.9971 | 14.76 |
| | CNN only | 98.64 | 0.9930 | 0.9956 | 19.85 |
| | BiLSTM only | 98.71 | 0.9933 | 0.9959 | 15.20 |
| | Transformer only | 98.62 | 0.9929 | 0.9954 | 23.75 |
| | Backbone without early exits | 98.77 | 0.9936 | 0.9961 | 14.26 |
| | EENet-IPS (routed, early exits) | 98.79 | 0.9937 | 0.9964 | 14.21 |

Gradient boosting nevertheless really lacks three main properties that EENet-IPS needs. First, the tree ensembles cannot be merged by averaging their parameters, so the federated tree learning needs dedicated protocols [65,66]. The histogram-based federated gradient boosting can be used to rebuild the centralized trees. However, the clients must exchange

the gradient histograms at each level of each tree [66]. For the gradient-boosting models in Table 9, this means at least 1137–2536 synchronous exchanges and about 0.38–0.96 GB of uplink traffic (five clients, float32 gradient and Hessian sums per histogram bin), against 15–60 rounds and 4.5–17.9 MB for the proposed EENet-FL; see Table 10. The one-shot ensembles of locally trained trees avoid these exchanges and can be accurate. However, the server then receives the complete, unperturbed model of each client, and the inference cost grows with the number of clients. Second, the backbone is differentiable. So, the tree ensembles admit the exact TreeSHAP attributions [67], but they have no gradients. Therefore, the Integrated Gradients, the constrained counterfactual search of Exit 3, and the gradient-based adversarial training do not apply to them directly, where the counterfactuals for tree ensembles require mixed-integer programming [68]. Third, the depth of the EENet-IPS adapts to each flow: 73.9–99.9% of the flows are decided after the first stage, the routed system matches or exceeds every deep alternative in accuracy and false-alarm rate on UNSW-NB15 and CICIoT2023, and its calibrated exit confidences drive the graded actions.

**Table 10.** The federated training (five non-IID clients, four per round, seed 1) compared with the centralized training (mean over three runs). Uplink: the total data sent by the clients over all rounds. The federated values correspond to a single run and may vary with the client partition and the training randomness.

| Dataset | Training | Accuracy (%) | F1-Score | AUC | FPR (%) | Exit-1 Share (%) | Uplink (MB) |
|---|---|---|---|---|---|---|---|
| CSE-CIC-IDS2018 | Centralized (upper bound) | 99.98 | 0.9993 | 0.9999 | 0.00 | 99.9 | – |
| | FedAvg (dense) | 99.89 | 0.9963 | 0.9974 | 0.08 | 100.0 | 22.4 |
| | FedProx (dense) | 99.89 | 0.9963 | 0.9975 | 0.08 | 100.0 | 22.4 |
| | EENet-FL (clip + noise + top-$k$) | 99.89 | 0.9963 | 0.9969 | 0.08 | 100.0 | 4.5 |
| UNSW-NB15 | Centralized (upper bound) | 95.78 | 0.9415 | 0.9912 | 2.50 | 73.9 | – |
| | FedAvg (dense) | 93.70 | 0.9145 | 0.9811 | 5.41 | 62.2 | 89.7 |
| | FedProx (dense) | 93.71 | 0.9145 | 0.9815 | 5.32 | 59.7 | 89.7 |
| | EENet-FL (clip + noise + top-$k$) | 94.03 | 0.9179 | 0.9818 | 4.34 | 56.2 | 17.9 |
| CICIoT2023 | Centralized (upper bound) | 98.79 | 0.9937 | 0.9964 | 14.21 | 94.5 | – |
| | FedAvg (dense) | 98.12 | 0.9903 | 0.9923 | 25.65 | 93.9 | 22.4 |
| | FedProx (dense) | 98.21 | 0.9907 | 0.9932 | 21.78 | 94.0 | 22.4 |
| | EENet-FL (clip + noise + top-$k$) | 98.01 | 0.9897 | 0.9925 | 18.07 | 93.9 | 4.5 |

The binary accuracy can hide the weak performance at the rare attack types. Table 11 reports for each attack family the share of the test flows flagged as malicious (detection rate), and for benign traffic, the false-alarm rate. The CICIoT2023 classes have been grouped into the seven families of [63]. After the de-duplication, only 54 unique FTP-BruteForce flows remain in the CSE-CIC-IDS2018 dataset, and only 66.7% of the 15 in the test partition have been flagged as malicious, although the overall accuracy is 99.98%. In the CICIoT2023, the flood attacks (DDoS, DoS, and Mirai) have been detected at rates of at least 99.99%, whereas the brute-force, reconnaissance, spoofing, and web-based attacks are detected at only 74.6–83.1%. These weaker families are where future improvements, e.g., class-balanced training or additional features, are most needed.

Table 12 illustrates the cost of each explainer next to the inference latency of the same exit (single flow, one CPU thread), together with a quality measure. One explanation costs 72–117 ms for KernelSHAP, 39.2 ms for Integrated Gradients, and 100.4 ms for a counterfactual using the UNSW-NB15 dataset. This is 32 to 342 times the inference time of the same exit, which is why the explanations are generated only for the flagged flows and outside the routing path. KernelSHAP satisfies the local accuracy almost exactly. The median completeness error of the Integrated Gradients is at most 2.70%. On the UNSW-NB15 and CICIoT2023 datasets, the counterfactuals are valid for 100.0% and 100.0% of the explained flows, change on average 5.5 and 5.2 features, and never violate the constraints. On the CSE-CIC-IDS2018 dataset, only about five test flows per run reach Exit 3, so the explained set is completed with the least confident flows, and only 51.0% of them can be flipped within the feasible range and the step budget. As the explainers do not modify the

classifier, all of the accuracies in this section are the same with or without explanations; their only cost is the latency in Table 12.

**Table 11.** Detection rate per attack family and false-alarm rate on benign traffic (routed EENet-IPS, test partitions, mean over three runs; $n$ is the number of test flows).

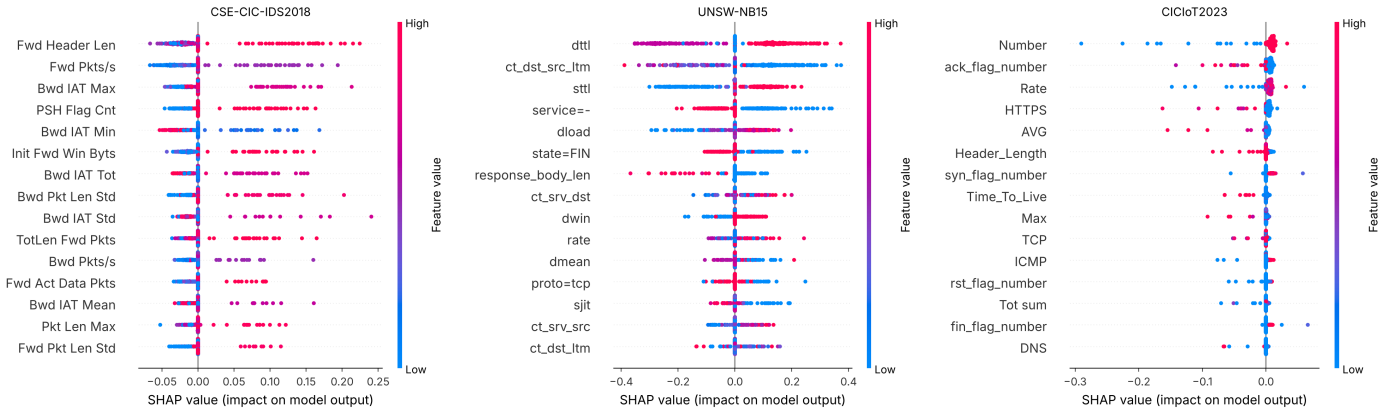
| Dataset | Family | $n$ | Flagged as Malicious (%) |
|---|---|---|---|
| CSE-CIC-IDS2018 | FTP-BruteForce | 15 | 66.67 |
| | SSH-Bruteforce | 18,804 | 99.89 |
| | Benign (false alarms) | 105,141 | 0.00 |
| UNSW-NB15 | Analysis | 61 | 100.00 |
| | Backdoor | 11 | 84.85 |
| | DoS | 274 | 92.34 |
| | Exploits | 1442 | 96.07 |
| | Fuzzers | 848 | 80.58 |
| | Generic | 741 | 98.83 |
| | Reconnaissance | 486 | 95.27 |
| | Shellcode | 74 | 91.44 |
| | Worms | 11 | 84.85 |
| | Benign (false alarms) | 6840 | 2.50 |
| CICIoT2023 | Brute force | 42 | 74.60 |
| | DDoS | 70,327 | 100.00 |
| | DoS | 18,363 | 100.00 |
| | Mirai | 7986 | 100.00 |
| | Reconnaissance | 2097 | 75.39 |
| | Spoofing | 1429 | 83.11 |
| | Web-based | 65 | 74.87 |
| | Benign (false alarms) | 3310 | 14.21 |

Moreover, we evaluate the SHAP based on the analysis and the heatmaps, computed at Exit 1 for 300 random test flows of each dataset. The SHAP has been utilized to quantify the impact of each feature on the predictions of the model to identify the critical network characteristics, such as the flags, the packet rates, or the flow metrics that can be used to distinguish the malicious traffic from the normal one. The heatmaps have been used to achieve this by visualizing the correlations between the SHAP values of the most influential features. Figures 6 and 7 illustrate how these tools differentiate benign and attack traffic across datasets.

In CSE-CIC-IDS2018, the most impactful features are the Fwd Header Len, Fwd Pkts/s, and Bwd IAT Max. In UNSW-NB15, they are dttl, ct_dst_src_ltm, sttl, and the record id no longer appears because it has been removed from the inputs. In CICIoT2023, they are Number, ack_flag_number, and Rate. The red and blue colors in Figure 7 represent the strength and direction of the correlation, where red implies a positive correlation and blue represents a negative one. This analysis helps us to understand the internal relationships between the features that have the most impact on the model decisions for each dataset, and the variation in the correlation patterns reflects the different attack types.

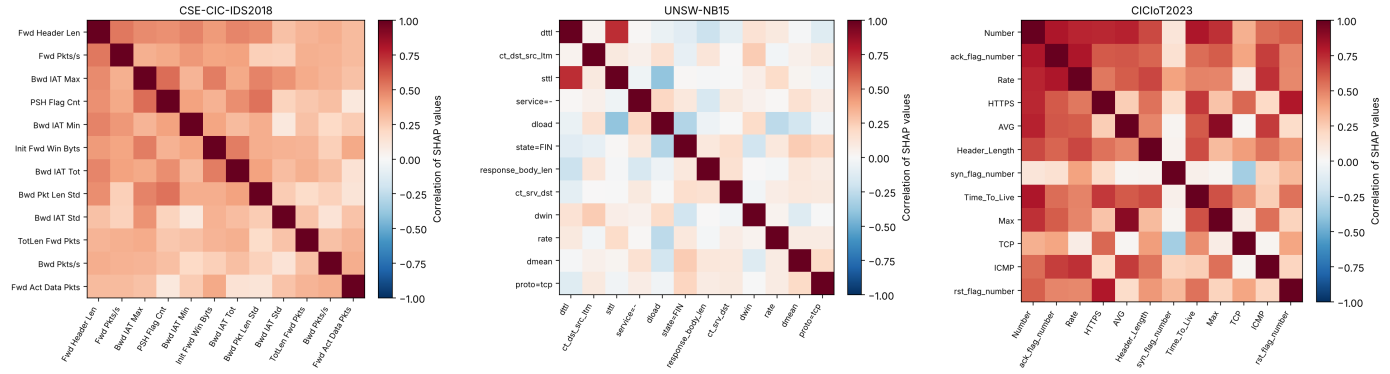
**Table 12.** Cost and quality of the exit-specific explainers (first run). Inference: the single-flow latency of the exit path; explanation: time to explain one flow (mean ± standard deviation over 100 flows; for the counterfactuals, 200 flows that reach Exit 3 or, on CSE-CIC-IDS2018, have the lowest Exit-3 confidence).

| Dataset | Explainer | Inference (ms) | Explanation (ms) | Quality |
|---|---|---|---|---|
| CSE-CIC-IDS2018 | Exit 1: KernelSHAP | 0.384 | 97.3 ± 15.1 | local-accuracy gap $1.1\times10^{-9}$ |
| | Exit 2: Integrated Gradients | 1.201 | 39.0 ± 10.3 | completeness error 0.58% |
| | Exit 3: counterfactual | 2.278 | 565.8 ± 469.9 | validity 51.0%, 4.3 features, 0 violations |
| UNSW-NB15 | Exit 1: KernelSHAP | 0.343 | 117.1 ± 23.7 | local-accuracy gap $4.1\times10^{-8}$ |
| | Exit 2: Integrated Gradients | 1.072 | 39.2 ± 8.3 | completeness error 2.70% |
| | Exit 3: counterfactual | 2.057 | 100.4 ± 138.1 | validity 100.0%, 5.5 features, 0 violations |
| CICIoT2023 | Exit 1: KernelSHAP | 0.358 | 71.8 ± 11.5 | local-accuracy gap $1.7\times10^{-9}$ |
| | Exit 2: Integrated Gradients | 1.109 | 38.1 ± 12.0 | completeness error 0.35% |
| | Exit 3: counterfactual | 2.104 | 90.6 ± 153.7 | validity 100.0%, 5.2 features, 0 violations |




(**a**) CSE-CIC-IDS2018 (**b**) UNSW-NB15 (**c**) CICIoT2023

**Figure 6.** SHAP summary at Exit 1 in three different datasets.

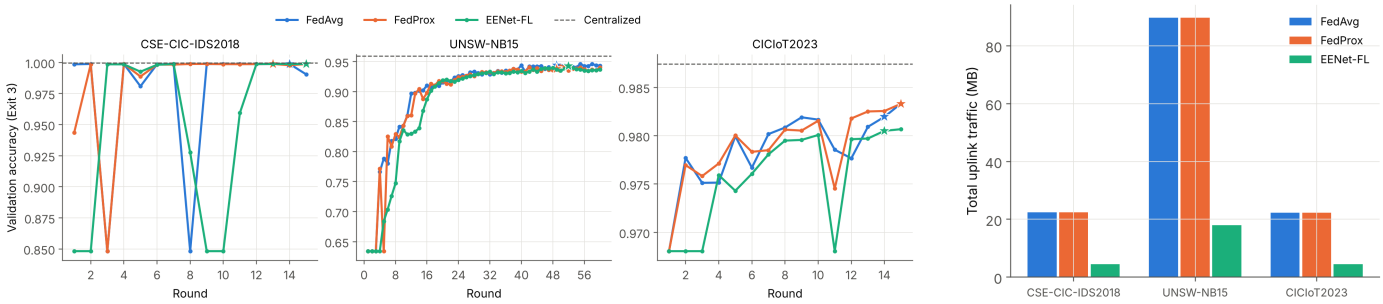



(**a**) CSE-CIC-IDS2018 (**b**) UNSW-NB15 (**c**) CICIoT2023

**Figure 7.** The correlation between the SHAP values of the twelve most influential features in three different datasets.

#### 4.3.3. Experiment III: Federated Early-Exit Training

The federated configuration is described in Section 3.3: five non-IID clients, four of which take part in each round, one local epoch (at most 50 steps) per round, and 15 rounds (60 for UNSW-NB15). Table 10 compares the centralized training with the plain FedAvg, FedProx, and EENet-FL (clipping, Gaussian noise, and top-10% sparsification with error feedback), all with the same backbone, calibration, and routing. EENet-FL reaches 99.89%, 94.03%, and 98.01% accuracy, i.e., at most 1.75 percentage points below the centralized training, while each client uploads about 5.0 times less data than with FedAvg (last column of Table 10). The saving comes from the top-$k$ sparsification; early exits act only at inference time and do not really change the size of the exchanged updates. Figure 8 shows the convergence and the uplink traffic. Under the strong label skew of CSE-CIC-IDS2018, the validation accuracy of all three configurations is dropped in some rounds, which is why the global model with the lowest validation loss has been maintained. The federated results correspond to a single realization (seed 1), whereas the centralized results are averaged over three runs. The seed determines the Dirichlet partition of the data among the clients as well as the initialization, client sampling, and local data order, so the values in Table 10 may vary with a different client partition or initialization. For this seed, the three federated configurations use the same client partition, and their accuracies differ by at most 0.33 percentage points, which is less than the run-to-run standard deviation of the centralized training on UNSW-NB15 (0.45 percentage points, Table 4); these differences should therefore not be read as a ranking of the configurations.




(**a**) Validation accuracy (Exit 3) per round (**b**) Total uplink traffic

**Figure 8.** The federated EENet-IPS: (**a**) validation accuracy of Exit 3 after each round for FedAvg, FedProx, and EENet-FL (stars: selected round with the lowest validation loss; dashed line: centralized accuracy) and (**b**) total uplink traffic per configuration.

#### 4.3.4. Comparison with Related Studies

In order to situate the effectiveness of our EENet-IPS, we compare it with the recently published studies in terms of accuracy, technique, dataset, and whether FL, XAI, and low-latency operation are really supported, as illustrated in Table 13. In fact, two caveats have been applied to this comparison and stated explicitly rather than left implicit. First, the listed accuracies are as reported in each study's own paper, on that study's own dataset, split, and hardware, so this is a survey-style comparison of the published outcomes rather than a controlled head-to-head reproduction under the given identical conditions. Second, the "accuracy" alone is an incomplete basis metric for the comparison because it does not really reveal whether a technique also preserves privacy, remains interpretable, or stays within a real-time limited latency budget.

**Table 13.** Comparison with other related work.

| No. | Research | Year | Dataset | Algorithms | Accuracy | FL | XAI | Latency |
|---|---|---|---|---|---|---|---|---|
| 1 | [43] | 2025 | N-BaIoT | DL-models | 90.39 | Y | N | N |
| 2 | [44] | 2025 | CICIoT2023 | DNN+CNN+ hybrid CNN+BiLSTM | ≈99 | Y | N | Y |
| 3 | [45] | 2024 | Edge-IIoTset, CIC-IDS2017 | CNN | 95.8 | Y | N | Y |
| 4 | [34] | 2025 | MQTT | DNN | 95.59 | Y | N | Y |
| 5 | [38] | 2025 | CICIoT2023 | ANN + SHAP | 88.2 | Y | Y | N |
| 6 | [46] | 2023 | TON-IoT | DBN+DNN | 70 | Y | N | N |
| 7 | [47] | 2025 | Real-world non-synthetic data / benchmark datasets | SVM (federated setup) and other ML models | 99.2 | Y | N | Y |
| 8 | [48] | 2025 | TON-IoT | CNN | 99.5 | N | Y | N |
| 9 | [49] | 2024 | UNSW-NB15 | NN (FPGA) | 90.91 | N | N | Y |
| 10 | [50] | 2025 | CICIDS 2017, TON-IoT, UNSW NB15, AWS CloudTrail, CTU-13, KDD99 | GNN (GNN-AID), QI-VAE, Transformers | GNN-AID: 98.7 HT-SDM 99.1 | Y | N | Y |
| 11 | [51] | 2025 | CICIoT2023 | DNN+MLP | 85% | N | N | N |
| 12 | [52] | 2022 | WSN-DS | CNN | 97 | N | N | N |
| 13 | EENet | 2026 | CSE-CIC-IDS2018 UNSW-NB15 CICIoT2023 | CNN–BiLSTM–Transformer early exits + FedAvg | 99.98 95.78 98.79 | Y | Y | Y |

After carefully re-checking each study, we have corrected the attributions of rows 5, 6, 7, and 9, which had been shifted, and we now mark the federated XAI-IDS of Fatema et al. [38] as supporting XAI. Table 13 therefore shows that FL and XAI have already been combined [38], and that federated or explainable IDSs reach accuracies comparable to ours on their own data. Among the compared studies, what distinguishes EENet-IPS is the combination of early-exit inference, exit-specific explanations, and federated training.

Its accuracies are competitive, and since they are measured on different data, splits, and leakage controls, they do not establish superiority over the compared techniques.

## 5. Limitations and Future Work

Evaluation scope. The evaluation has replayed the labeled flow records offline. The latencies are CPU timings of the model and the policy decision; they exclude the flow-feature extraction and the enforcement of the actions, and the replay test does not really model concurrency or queueing. We will therefore deploy the proposed EENet-IPS inline on representative edge hardware (Jetson- and Raspberry-Pi-class devices) next to an SDN controller. There, we will measure the end-to-end delay from the packet capture through flow export, decision, and flow-rule installation under the live and concurrent traffic.

Adversarial robustness. An adaptive attacker may craft the flows that reach high confidence at Exit 1 with benign-looking features, or may poison the federated updates. We plan to (i) utilize the constrained counterfactual generator of Exit 3 in order to produce feasible evasion flows for adversarial training and robustness evaluation, (ii) monitor the shifts in the per-exit confidence distributions and in the Exit-1 share as an early warning, and (iii) replace the FedAvg by the robust aggregation, e.g., trimmed mean or Krum, with client anomaly scoring.

Model degradation over time. The traffic and attacks drift, and the accuracy of a fixed model degrades. The drift will be monitored through the per-exit confidence distributions, the Exit-1 share, and the escalation rate. When these leave their reference ranges, the new federated rounds will start from the current global model with the analyst-confirmed labels of the decision log, instead of retraining from scratch, followed by re-calibration of the temperatures and re-checking of the thresholds. How many rounds and labeled flows this requires will be determined experimentally.

Privacy. The update perturbation is inspired by differential privacy but is not accounted, and FedAvg alone does not prevent inference attacks. Future work will calibrate the noise to a target $(\varepsilon, \delta)$ with a privacy accountant, add secure aggregation, and evaluate membership-inference and gradient-inversion attacks.

Explainability and data. Routing currently uses the calibrated confidence only, and the explainers cost far more than the inference. We will compare XAI-assisted routing with confidence-only routing and use faster amortized explainers. Finally, each dataset is represented by one capture or partition, so temporal and cross-dataset generalization will be evaluated in future work. EENet-IPS classifies each flow record independently from its tabular features; modelling actual temporal behavior, e.g., sequences of consecutive flows of the same host or packet-level sequences, would require a different input representation and is left for future work. Finally, the federated experiments have been run with one seed; repeating them over several client partitions and initializations in order to quantify their variance is also part of our future work.

## 6. Conclusions

In this work, a novel EENet-IPS has been designed and implemented in order to effectively detect different network intrusions and map them to prevention actions. Multi-exit DNNs, federated learning, and exit-aware explainability have been combined to address the main limitations of existing previous IPS techniques, including high computational overhead, data privacy issues, and limited interpretability. Simply, the high-confidence traffic exits early through a shallow CNN stage, while more ambiguous or complex flows are routed to a BiLSTM and, if needed, a Transformer exit for deeper analysis. Each exit has been paired with an explainer (SHAP, Integrated Gradients, or constrained counterfactuals), whose explanation is logged with the decision of a rule-based mitigation policy.

Under the leakage-controlled protocol with three repeated runs, the routed system accomplishes 99.98%, 95.78%, and 98.79% accuracy on the CSE-CIC-IDS2018, UNSW-NB15, and CICIoT2023 datasets, respectively. It resolves 73.9–99.9% of the flows at the first exit, which keeps the mean single-flow CPU latency at 0.55–0.83 ms instead of 2.06–2.28 ms at full depth. In a single federated run, the federated training with clipped, noise-perturbed, and sparsified updates stays within 1.75 percentage points of the centralized training and decreases the uplink traffic by about 5.0-fold. Among the twelve compared studies, the proposed EENet-IPS is the design that combines early-exit inference, federated training, and per-exit explanations, with competitive accuracy. The gradient-boosted trees are still slightly more accurate on these tabular benchmarks, but the EENet-IPS offers what they lack in this setting: training by federated averaging in 15–60 rounds rather than more than a thousand histogram exchanges, gradients for explanations and counterfactual search, and per-flow adaptive depth.

**Author Contributions:** Conceptualization, Q.A.; methodology, Q.A. and S.A.; proposal improvement and implementation, Q.A.; software running, S.A.A., Q.A. and S.A.; validation, Q.A., M.A. and O.T.K.; formal analysis, Q.A. and S.A.A.; investigation, M.A. and S.A.; data curation, S.A.A., O.T.K. and M.A.; writing—original draft preparation, Q.A. and S.A.; writing—review and editing, Q.A., S.A. and J.-S.Y.; supervision, J.-S.Y. and Q.A. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research has received no external funding.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The three utilized datasets analyzed in the current study are publicly available. The CSE-CIC-IDS2018 is available from the Canadian Institute for Cybersecurity at https://www.unb.ca/cic/datasets/ids-2018.html (accessed on 4 January 2026) (the file Wednesday-14-02-2018_TrafficForML_CICFlowMeter.csv). The UNSW-NB15 is available from the University of New South Wales at https://research.unsw.edu.au/projects/unsw-nb15-dataset (https://doi.org/10.26190/5d7ac5b1e8485 (accessed on 4 January 2026)) (the file UNSW_NB15_testing-set.csv). Finally, the CICIoT2023 is available from the Canadian Institute for Cybersecurity at https://www.unb.ca/cic/datasets/iotdataset-2023.html (accessed on 4 January 2026) (the file Merged01.csv of the merged CSV release). The code, the run logs, and the scripts that can produce all of the results are publicly available at https://github.com/QutaibaEng/EENet-IPS (accessed on 23 October 2026).

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

## References

1. Bijone, M. A Survey on Secure Network: Intrusion Detection & Prevention Approaches. *Am. J. Inf. Syst.* **2016**, *4*, 69–88. Available online: http://pubs.sciepub.com/ajis/4/3/2 (accessed on 28 November 2025).
2. Alasad, Q.; Ahmed, M.; Alahmed, S.; Khattab, O.T.; Abdulwahhab, S.A.; Yuan, J.S. A Comprehensive Review: The Evolving Cat-and-Mouse Game in Network Intrusion Detection Systems Leveraging Machine Learning. *J. Cybersecur. Priv.* **2026**, *6*, 13 . [CrossRef]
3. Chakraborty, N. Intrusion Detection System and Intrusion Prevention System: A Comparative Study. *Int. J. Comput. Bus. Res. (IJCBR)* **2013**, *4*, 1–8. Available online: http://researchmanuscripts.com/May2013/1.pdf (accessed on 17 December 2025). [CrossRef]
4. Ahmed, M.; Alasad, Q.; Yuan, J.S.; Alawad, M. Re-Evaluating Deep Learning Attacks and Defenses in Cybersecurity Systems. *Big Data Cogn. Comput.* **2024**, *8*, 191. [CrossRef]
5. Bilge, L.; Dumitraş, T. Before we knew it: An empirical study of zero-day attacks in the real world. In *Proceedings of the CCS'12: The ACM Conference on Computer and Communications Security*; ACM: Raleigh, NC, USA, 2012; pp. 833–844.
6. Abbas, S.; Naser, W.; Kadhim, A. Subject review: Intrusion detection system (IDS) and intrusion prevention system (IPS). *Glob. J. Eng. Technol. Adv.* **2023**, *2*, 155–158. [CrossRef]

7. Salih, A.A.; Ameen, S.Y.; Zeebaree, S.R.; Sadeeq, M.A.; Kak, S.F.; Omar, N.; Ibrahim, I.M.; Yasin, H.M.; Rashid, Z.N.; Ageed, Z.S. Deep learning approaches for intrusion detection. *Asian J. Res. Comput. Sci.* **2021**, *9*, 50–64. [CrossRef]
8. Alasad, Q.; Hammood, M.M.; Alahmed, S. Performance and Complexity Tradeoffs of Feature Selection on Intrusion Detection System on FPGAs: An Algorithm-Architecture Interaction. In *Proceedings of the 2nd International Conference on Emerging Technologies and Intelligent Systems (ICETIS 2022)*; Al-Sharafi, M.A., Al-Emran, M., Al-Kabi, M.N., Shaalan, K., Eds.; Lecture Notes in Networks and Systems; Springer: Cham, Switzerland, 2023; Volume 584, pp. 533–542. [CrossRef]
9. Araujo-Filho, P.F.D.; Pinheiro, A.J.; Kaddoum, G.; Campelo, D.R.; Soares, F.L. An efficient intrusion prevention system for CAN: Hindering cyber-attacks with a low-cost platform. *IEEE Access* **2021**, *9*, 166855–166869. [CrossRef]
10. Quincozes, S.E.; Albuquerque, C.; Passos, D.; Mossé, D. A survey on intrusion detection and prevention systems in digital substations. *Comput. Netw.* **2021**, *184*, 107679. [CrossRef]
11. Jayalaxmi, P.L.S.; Saha, R.; Kumar, G.; Conti, M.; Kim, T.H. Machine and Deep Learning Solutions for Intrusion Detection and Prevention in IoTs: A Survey. *IEEE Access* **2022**, *10*, 121173–121192. [CrossRef]
12. Alahmed, S.; Alasad, Q.; Yuan, J.S.; Alawad, M. Impacting Robustness in Deep Learning-Based NIDS through Poisoning Attacks. *Algorithms* **2024**, *17*, 155. [CrossRef]
13. Alahmed, S.; Alasad, Q.; Hammood, M.M.; Yuan, J.S.; Alawad, M. Mitigation of black-box attacks on intrusion detection systems-based ml. *Computers* **2022**, *11*, 115. [CrossRef]
14. Fatehi, N.; Alasad, Q.; Alawad, M. Towards Adversarial Attacks for Clinical Document Classification. *Electronics* **2023**, *12*, 129. [CrossRef]
15. Seo, W.; Pak, W. Real-time network intrusion prevention system based on hybrid machine learning. *IEEE Access* **2021**, *9*, 46386–46397. [CrossRef]
16. Wen, J.; Zhang, Z.; Lan, Y.; Cui, Z.; Cai, J.; Zhang, W. A survey on federated learning: Challenges and applications. *Int. J. Mach. Learn. Cybern.* **2023**, *14*, 513–535. [CrossRef] [PubMed]
17. Bharati, S.; Mondal, M.R.H.; Podder, P.; Prasath, V.B.S. Federated learning: Applications, challenges and future directions. *Int. J. Hybrid. Intell. Syst.* **2022**, *18*, 19–35. [CrossRef]
18. Teerapittayanon, S.; McDanel, B.; Kung, H.T. BranchyNet: Fast Inference via Early Exiting from Deep Neural Networks. In *Proceedings of the 2016 23rd International Conference on Pattern Recognition (ICPR), Cancun, Mexico, 4–8 December 2016*; IEEE: New York, NY, USA, 2016; pp. 2464–2469. [CrossRef]
19. Arreche, O.; Guntur, T.; Abdallah, M. XAI-IDS: Toward Proposing an Explainable Artificial Intelligence Framework for Enhancing Network Intrusion Detection Systems. *Appl. Sci.* **2024**, *14*, 4170. [CrossRef]
20. Muhammad, A.E.; Yow, K.C.; Bačanin-Džakula, N.; Khan, M.A. L-xaids: A LIME-based eXplainable AI framework for intrusion detection systems. *Clust. Comput.* **2025**, *28*, 654. [CrossRef]
21. Elhoseny, M.; Riad, K. RT-PPS: Real-time privacy-preserving scheme for cloud-hosted IoT data. *J. High Speed Netw.* **2025**, *31*, 71–89. (In English) [CrossRef]
22. Rahmati, M.; Pagano, A. Federated Learning-Driven Cybersecurity Framework for IoT Networks with Privacy Preserving and Real-Time Threat Detection Capabilities. *Informatics* **2025**, *12*, 62. [CrossRef]
23. Silva, H.; Moraes, R. Privacy-Preserving IoT Intrusion Detection: Challenges and Solutions in Implementing the CSAI-4-CPS Model. *Comput. Sci. Inf. Technol. (CS & IT)* **2024**, *14*, 107–125. [CrossRef]
24. Bocu, R.; Iavich, M. Real-time intrusion detection and prevention system for 5G and beyond software-defined networks. *Symmetry* **2022**, *15*, 110. [CrossRef]
25. Constantinides, C.; Shiaeles, S.; Ghita, B.; Kolokotronis, N. A novel online incremental learning intrusion prevention system. In *Proceedings of the 2019 10th International Conference on Information, Intelligence, Systems and Applications (IISA), Patras, Greece, 15–17 July 2019*; IEEE: New York, NY, USA, 2019; pp. 1–6. [CrossRef]
26. Songma, S.; Sathuphan, T.; Pamutha, T. Optimizing Intrusion Detection Systems in Three Phases on the CSE-CIC-IDS-2018 Dataset. *Computers* **2023**, *12*, 245. [CrossRef]
27. Kumar, D.; Pawar, P.P.; Ananthan, B.; Rajasekaran, S.; Prabhakaran, T.V. Optimized Support Vector Machine Based Fused IoT Network Security Management. In *Proceedings of the 2024 3rd International Conference on Artificial Intelligence for Internet of Things (AIIoT), Vellore, India, 3–4 May 2024*; IEEE: New York, NY, USA, 2024; pp. 1–5. [CrossRef]
28. Shahid, U.; Zunnurain Hussain, M.; Zulkifl Hasan, M.; Haider, A.; Ali, J.; Altaf, J. Hybrid Intrusion Detection System for RPL IoT Networks Using Machine Learning and Deep Learning. *IEEE Access* **2024**, *12*, 113099–113112. [CrossRef]
29. Chatterjee, S.; Chaudhary, S.; Cherukuri, A.K. Intrusion Detection System Using Deep Learning for Network Security. *arXiv* **2025**, arXiv:2505.05810.
30. Jablaoui, R.; Cheikhrouhou, O.; Hamdi, M.; Liouane, N. Deep learning enabled intrusion detection system for IoT security. *EURASIP J. Wirel. Commun. Netw.* **2025**, *2025*, 66. (In English) [CrossRef]
31. Li, T.; Sahu, A.K.; Talwalkar, A.; Smith, V. Federated learning: Challenges, methods, and future directions. *IEEE Signal Process. Mag.* **2020**, *37*, 50–60. [CrossRef]

32. Beuran, R. FedMSE: Semi-supervised federated learning approach for IoT network intrusion detection. *Comput. Secur.* **2025**, *151*, 104337. [CrossRef]
33. Wang, J.; Yang, K.; Li, M. NIDS-FGPA: A federated learning network intrusion detection algorithm based on secure aggregation of gradient similarity models. *PLoS ONE* **2024**, *19*, e0308639. [CrossRef]
34. Karunamurthy, A.; Vijayan, K.; Kshirsagar, P.R.; Tan, K.T.J.S.R. An optimal federated learning-based intrusion detection for IoT environment. *Sci. Rep.* **2025**, *15*, 8696. [CrossRef] [PubMed]
35. Alomari, Z. Privacy-Preserving Intrusion Detection Using Federated Learning and Differential-Privacy-Inspired Noise. In *Proceedings of the 2025 4th International Conference on Computing, Management and Telecommunications (ComManTel), Madrid, Spain, 14–17 December 2025*; IEEE: New York, NY, USA, 2025; pp. 106–112. [CrossRef]
36. Al Mazroa, A. FORT-IDS: A Federated, Optimized, Robust and Trustworthy Intrusion Detection System for IIoT Security. *Sci. Rep.* **2026**, *16*, 1483. [CrossRef] [PubMed]
37. Kulkarni, K.; Muthuvel, S.K. FedTinyX-IDS: Federated Explainable TinyML for Robust WSN Intrusion Detection. In *Proceedings of the 2026 International Conference on Intelligent and Sustainable Electronics & Computing Technologies (INSECT), Pune, India, 29–30 May 2026*; IEEE: New York, NY, USA, 2026; pp. 1–6. [CrossRef]
38. Fatema, K.; Dey, S.K.; Anannya, M.; Khan, R.T.; Rashid, M.M.; Su, C.; Mazumder, R. Federated XAI IDS: An Explainable and Safeguarding Privacy Approach to Detect Intrusion Combining Federated Learning and SHAP. *Future Internet* **2025**, *17*, 234. [CrossRef]
39. Rjoub, G.; Bentahar, J.; Wahab, O.A.; Mizouni, R.; Song, A.; Cohen, R.; Otrok, H.; Mourad, A. A survey on explainable artificial intelligence for cybersecurity. *IEEE Trans. Netw. Serv. Manag.* **2023**, *20*, 5115–5140. [CrossRef]
40. Hosain, Y.; Çakmak, M. XAI-XGBoost: An innovative explainable intrusion detection approach for securing internet of medical things systems. *Sci. Rep.* **2025**, *15*, 22278. [CrossRef] [PubMed]
41. Keshk, M.; Koroniotis, N.; Pham, N.; Moustafa, N.; Turnbull, B.; Zomaya, A.Y. An explainable deep learning-enabled intrusion detection framework in IoT networks. *Inf. Sci.* **2023**, *639*, 119000. [CrossRef]
42. Sadhwani, S.; Navare, A.; Mohan, A.; Muthalagu, R.; Pawar, P.M. IoT-based intrusion detection system using explainable multi-class deep learning approaches. *Comput. Electr. Eng.* **2025**, *123*, 110256. [CrossRef]
43. Olanrewaju-George, B.; Pranggono, B. Federated learning-based intrusion detection system for the internet of things using unsupervised and supervised deep learning models. *Cyber Secur. Appl.* **2025**, *3*, 100068. [CrossRef]
44. Albanbay, N.; Tursynbek, Y.; Graffi, K.; Uskenbayeva, R.; Kalpeyeva, Z.; Abilkaiyr, Z.; Ayapov, Y. Federated Learning-Based Intrusion Detection in IoT Networks: Performance Evaluation and Data Scaling Study. *J. Sens. Actuator Netw.* **2025**, *14*, 78. [CrossRef]
45. Rehman, T.; Tariq, N.; Khan, F.A.; Rehman, S.U. FFL-IDS: A Fog-Enabled Federated Learning-Based Intrusion Detection System to Counter Jamming and Spoofing Attacks for the Industrial Internet of Things. *Sensors* **2025**, *25*, 10. [CrossRef] [PubMed]
46. Belarbi, O.; Spyridopoulos, T.; Anthi, E.; Mavromatis, I.; Carnelli, P.; Khan, A. Federated deep learning for intrusion detection in IoT networks. In *Proceedings of the GLOBECOM 2023-2023 IEEE Global Communications Conference, Kuala Lumpur, Malaysia, 8–12 December 2023*; IEEE: New York, NY, USA, 2023; pp. 237–242.
47. Devine, M.; Ardakani, S.P.; Al-Khafajiy, M.; James, Y. Federated Machine Learning to Enable Intrusion Detection Systems in IoT Networks. *Electronics* **2025**, *14*, 1176. [CrossRef]
48. Ebrahimi, F.; Javidan, R.; Akbari, R.; Hosseini, Y. Intrusion detection in the internet of things using convolutional neural networks: An explainable AI approach. *Cybersecurity* **2025**, *8*, 66. [CrossRef]
49. Farooq, M.A.; Hassan, S.M.F.U.; Farooq, M.U.; Rafique, A. High Throughput Neural Network for Network Intrusion Detection on FPGAs: An Algorithm-Architecture Interaction. In *Proceedings of the ICISSP, Rome, Italy, 26–28 February 2024*; Curran Associates, Inc.: Northampton, MA, USA, 2024; pp. 423–429.
50. Nagarjun, A.V.; Rajkumar, S. Quantum deep learning-enhanced ethereum blockchain for cloud security: Intrusion detection, fraud prevention, and secure data migration. *Sci. Rep.* **2025**, *15*, 38711. [CrossRef] [PubMed]
51. Erskine, S.K. Real-Time Large-Scale Intrusion Detection and Prevention System (IDPS) CICIoT Dataset Traffic Assessment Based on Deep Learning. *Appl. Syst. Innov.* **2025**, *8*, 52. [CrossRef]
52. Chandre, P.R.; Mahalle, P.; Shinde, G. Intrusion prevention system using convolutional neural network for wireless sensor network. *IAES Int. J. Artif. Intell. (IJ-AI)* **2022**, *11*, 504–515. [CrossRef]
53. Guo, C.; Pleiss, G.; Sun, Y.; Weinberger, K.Q. On Calibration of Modern Neural Networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML), Sydney, Australia, 6–11 August 2017*; Proceedings of Machine Learning Research: Red Hook, NY, USA, 2017; Volume 70, pp. 1321–1330.
54. Lundberg, S.M.; Lee, S.I. A Unified Approach to Interpreting Model Predictions. In *Proceedings of the Advances in Neural Information Processing Systems 30 (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017*; Neural Information Processing Systems Foundation: La Jolla, CA, USA, 2027; pp. 4765–4774.

55. Sundararajan, M.; Taly, A.; Yan, Q. Axiomatic Attribution for Deep Networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML), Sydney, Australia, 6–11 August 2017*; Proceedings of Machine Learning Research: Cambridge, MA, USA, 2017; Volume 70, pp. 3319–3328.
56. Wachter, S.; Mittelstadt, B.; Russell, C. Counterfactual Explanations without Opening the Black Box: Automated Decisions and the GDPR. *Harv. J. Law Technol.* **2018**, *31*, 841–887.
57. McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; Agüera, Y.; Arcas, B. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS), Fort Lauderdale, FL, USA, 20–22 April 2017*; Proceedings of Machine Learning Research: Cambridge, MA, USA, 2017; Volume 54, pp. 1273–1282.
58. Stich, S.U.; Cordonnier, J.B.; Jaggi, M. Sparsified SGD with Memory. In *Proceedings of the Advances in Neural Information Processing Systems 31 (NeurIPS 2018), Montréal, QC, Canada, 3–8 December 2018*; Curran Associates, Inc.: Red Hook, NY, USA, 2018; pp. 4447–4458.
59. Menon, A.K.; Jayasumana, S.; Rawat, A.S.; Jain, H.; Veit, A.; Kumar, S. Long-Tail Learning via Logit Adjustment. In Proceedings of the 9th International Conference on Learning Representations (ICLR), Virtual Event, 3–7 May 2021.
60. Li, T.; Sahu, A.K.; Zaheer, M.; Sanjabi, M.; Talwalkar, A.; Smith, V. Federated Optimization in Heterogeneous Networks. *Proc. Mach. Learn. Syst. (MLSys)* **2020**, *2*, 429–450.
61. Sharafaldin, I.; Lashkari, A.H.; Ghorbani, A.A. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP), Madeira, Portugal, 22–24 January 2018*; SciTePress: Setúbal, Portugal, 2018; pp. 108–116. [CrossRef]
62. Moustafa, N.; Slay, J. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In *Proceedings of the 2015 Military Communications and Information Systems Conference (MilCIS), Canberra, Australia, 10–12 November 2015*; IEEE: New York, NY, USA, 2015; pp. 1–6.
63. Neto, E.C.P.; Dadkhah, S.; Ferreira, R.; Zohourian, A.; Lu, R.; Ghorbani, A.A. CICIoT2023: A Real-Time Dataset and Benchmark for Large-Scale Attacks in IoT Environment. *Sensors* **2023**, *23*, 5941. [CrossRef] [PubMed]
64. Saerens, M.; Latinne, P.; Decaestecker, C. Adjusting the Outputs of a Classifier to New a Priori Probabilities: A Simple Procedure. *Neural Comput.* **2002**, *14*, 21–41. [CrossRef] [PubMed]
65. Li, Q.; Wen, Z.; He, B. Practical Federated Gradient Boosting Decision Trees. *Proc. AAAI Conf. Artif. Intell.* **2020**, *34*, 4642–4649. [CrossRef]
66. Li, Q.; Wu, Z.; Cai, Y.; Han, Y.; Yung, C.M.; Fu, T.; He, B. FedTree: A Federated Learning System for Trees. *Proc. Mach. Learn. Syst. (MLSys 2023)* **2023**, *5*, 89–103.
67. Lundberg, S.M.; Erion, G.; Chen, H.; DeGrave, A.; Prutkin, J.M.; Nair, B.; Katz, R.; Himmelfarb, J.; Bansal, N.; Lee, S.I. From Local Explanations to Global Understanding with Explainable AI for Trees. *Nat. Mach. Intell.* **2020**, *2*, 56–67. [CrossRef] [PubMed]
68. Parmentier, A.; Vidal, T. Optimal Counterfactual Explanations in Tree Ensembles. In *Proceedings of the 38th International Conference on Machine Learning (ICML), Virtual Event, 18–24 July 2021*; Proceedings of Machine Learning Research: Cambridge, MA, USA, 2021; Volume 139, pp. 8422–8431.


**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.