International Journal of Safety and Security Engineering
Vol. 16, No. 2, February, 2026, pp. 297-305
Journal homepage: http://iieta.org/journals/ijsse

Fast Entropy Quantile Forest for Federated Distributed Denial of Service Detection in Software-Defined Networking Environments

Noble T N Pillai[1*], Meena Mathivanan[2]

Department of Electronics and Communication Engineering, Vels Institute of Science, Technology & Advanced Studies (VISTAS), Chennai 600 117, India

Corresponding Author Email: nobletnpillai@gmail.com

https://doi.org/10.18280/ijsse.160205

Received: 1 December 2025
Revised: 13 February 2026
Accepted: 20 February 2026
Available online: 28 February 2026

Keywords:
federated Distributed Denial of Service detection, Fast Entropy Quantile Forest, Software-Defined Networking, quantile-based feature splitting, network security, Machine Learning for Distributed Denial of Service detection, dynamic traffic patterns

ABSTRACT

Distributed Denial of Service (DDoS) detection in Software-Defined Networking (SDN) faces challenges related to scalability, privacy, and the bursty traffic patterns typical in these environments. This paper proposes the Fast Entropy Quantile Forest (FEQF), a novel federated tree construction model designed to enhance the detection of DDoS attacks. By replacing traditional impurity measures with a fast entropy metric and utilizing quantile-based feature splitting, the FEQF improves resilience and performance under challenging conditions. The model was evaluated using the Mendeley DDoS attack SDN dataset in a federated learning framework. Compared to federated Random Forest (RF) and Extra Trees Classifier (ETC), the FEQF outperforms them with a global accuracy of 97%, significantly lower aggregation risk, and an F1-score exceeding 0.95 under a severe class imbalance (1:100). Furthermore, in bursty client simulations, FEQF maintained stability, with the weakest client achieving an F1-score of 0.93. These results demonstrate the robustness of the FEQF in dynamic SDN environments and its potential as a scalable solution for DDoS detection.

1. INTRODUCTION

Software-Defined Networking (SDN) separates the network into three distinct planes. These are the control plane, the data plane, and the management plane. Each plane has its own responsibilities and operates independently from the others. Because of this, network management becomes more centralised and easier to control in a fine-grained way. This flexibility is the main reason why SDN has been widely adopted. However, this same centralisation and programmability also introduce security challenges, as attacks targeting control logic or exploiting dynamic flow management have network-wide consequences.

Distributed Denial of Service (DDoS) attacks are one of the most critical threats in SDN environments. The goal of threat actors behind a DDoS attack is to disrupt normal network operation. This is done by overwhelming network resources with large volumes of malicious traffic, generated simultaneously from multiple sources. In SDN, this type of attack is especially damaging. It triggers excessive flow setup requests, which ultimately puts the availability of the entire network at risk.

Centralised detection introduces two well-known problems: privacy concerns and scalability limitations. Federated learning (FL) was introduced to tackle both. In a federated SDN framework, learning happens locally at each edge site or branch. No raw traffic data or packet captures (PCAPs) leave the local site. Each node shares only its learned model updates. Despite this, implementing effective federated DDoS detection is not straightforward. Several technical challenges remain open, and this research works to address them.

A major obstacle in current FL research is the tree aggregation problem. Most existing federated frameworks aggregate model parameters and hyperparameters. This works well for many model types, but not for tree-based models. Decision trees have non-linear structures. When trained across multiple clients, each client produces its own distinct tree. These trees cannot be mathematically averaged. Because of this, conventional aggregation methods cannot produce a coherent global model.

Furthermore, real-world SDN traffic is inherently non-Independent and Identically Distributed (non-IID). Different regional sites observe disparate behaviors; some sites may be naturally "noisy" due to high activity, while others remain "calm". SDN traffic is also characterized by sudden, attack-induced surges known as bursty traffic, which can cause "client collapse" where the detection performance of individual nodes drops significantly. Additionally, these environments often suffer from severe class imbalance, where malicious flows are rare but critical minorities.

Another key limitation of existing federated DDoS detection systems is the risk to global stability, characterized by inconsistent convergence across clients and communication rounds. It is important to note that high accuracy alone is not sufficient. The model must also demonstrate stable convergence for deployment. If convergence is unstable, the aggregation process becomes ineffective, leading to unreliable global model performance.

While models like Random Forest (RF) and Extra Tree Classifier (ETC) often perform well in general settings, they frequently fail to deliver the desired performance under these specific, adverse conditions.

To overcome these challenges, this work proposes the Fast Entropy Quantile Forest (FEQF). FEQF is a novel federated tree construction model. It replaces traditional impurity measures with a fast entropy metric [1] to reduce computational overhead. It also uses quantile-based feature splitting to better handle the skewed and bursty traffic patterns that are typical in SDN environments.

There are mainly five contributions in this paper, which are listed below:

1. Identification of Scenario-Based Failures: We identify specific SDN scenarios, such as severe class imbalance, attack-induced traffic bursts and stability of global model convergence where existing general-purpose tree models, specifically RF and ETC, fail to provide stable and reliable detection performance.
2. Layered Federated Architecture: We present a three-level hierarchical framework adapts federated learning within the SDN framework.
3. The FEQF Algorithm: We introduce FEQF, a novel federated tree construction model. It is designed for stable aggregation and robustness. It achieves this by fast entropy and quantile-based split decisions to enhance resilience.
4. Customized Framework for Tree Pooling: We address the limitation of conventional FL aggregation (parameter and hyperparameter averaging) by extending an existing framework [2] that utilizes a pooling function to combine trained trees into a global model. We have extensively customized this framework to adapt to the specific architectural requirements of our proposed FEQF.
5. Empirical Validation under Stress: We conducted a comprehensive evaluation comparing FEQF with baseline models. The baseline models in use are the RF and ETC, focusing on performance under extreme class imbalance, bursty client scenarios, and convergence stability. In addition, we assess client-wise normal conditions, demonstrating the fact that our approach outperforms the baseline models in our environment.

The rest of the article is organized and structured as follows: Section 2 describes existing methods and previous works that contribute to improving security; Section 3 presents the methodology; Section 4 discusses experimental analysis; Section 5 concludes our work; and Section 6 provides future directions.

2. RELATED WORK

DDoS detection in SDN has evolved significantly. Early systems relied on traditional signature-based detection. Over time, these gave way to more adaptive, Machine Learning (ML)-driven approaches. This section reviews these advancements across four core categories: conventional ML and DL detection, statistical and information-theoretic methods, domain-specific federated approaches, and dataset development.

2.1 Conventional Deep Learning and Deep Learning detection

In conventional ML and DL architectures, most research has chased high accuracy using complex neural networks. Hnamte et al. [3] are a good example of this; they proposed a Deep Convolutional Neural Network that removes the pooling layer to reduce controller overhead. This achieved an accuracy of 99.99%.

Similarly, Al-Dunainawi et al. [4] utilized a 1D-CNN model optimized via the NSGA-II genetic algorithm to balance training time and precision. Hybridization is another dominant trend; Elshewey et al. [5] introduced a sequential CNN-GRU model for spatial-temporal feature extraction, while Wang et al. [6] proposed a two-stage detection approach combining switch port statistics with a Multi-Dimensional Deep Convolutional Classifier. Other researchers have explored automated feature learning, such as Ahuja et al. [7] Stacked Auto-Encoder MLP (SAE-MLP).

For real-time adaptability, Alashhab et al. [8] developed an ensemble online learning model to identify zero-day and low-rate DDoS attacks. Standard supervised classifiers also remain relevant; Alubaidan et al. [9] concluded that RF achieves 99% accuracy, while Karthika and Arockiasamy [10] found MLP superior for detecting TCP-SYN floods.

These models offer high accuracy. However, they rely on centralised architectures that expose raw traffic data. This raises privacy concerns and creates a single point of failure. Our work addresses both of these gaps by adopting a federated framework that keeps data local.

2.2 Statistical and information theoretical methods

In statistical and information theoretical methods, to minimize CPU utilization, researchers have introduced lightweight statistical measures. Guesmi et al. [11] extended the Fast Entropy computation method that drastically reduces the cost of calculating information entropy to secure cloud server environments.

Other innovations include the DeMi solution, which utilizes sample entropy with adaptive dynamic thresholds [12], and an Interquartile Range (IQR) approach proposed by Swami et al. [13] to detect outliers in "packet_in" messages. Swami et al. [13] specifically noted that such statistical measures are often more suitable than Deep Learning (DL) for real-time scenarios where minimizing CPU utilization is paramount.

Statistical methods are efficient but may struggle with complex, evolving attack patterns that mimic normal traffic. Our proposed FEQF model integrates the Fast Entropy metric directly into a tree-based ensemble, combining the speed of statistical measures with the predictive depth of ML.

2.3 Domain specific approaches

Several domain-specific approaches have been proposed in the literature. Khedr et al. [14] presented the FMDADM framework, designed specifically for stateful IoT networks. Garba et al. [15] focused on smart home security. Oleiwi et al. [16] took a different approach and proposed a privacy-preserving consensus system for the East-West interface.

Mortatha et al. [17] took a different direction and proposed a hybrid pipeline for the Internet of Medical Things (IoMT),

using the Whale Optimisation Algorithm.

In multi-controller settings, Gebremeskel et al. [18] used a hybrid entropy-LSTM model to share traffic information between neighbouring controllers. In distributed learning, Ma et al. [19] proposed EDRFS. This placed optimised RF models directly on SDN edge switches. Federated learning has also been explored in this context. Hauschild et al. [20] demonstrated that Federated Random Forests can improve the local predictive performance of models across distributed nodes.

A major bottleneck is the tree aggregation problem. Most FL frameworks focus on averaging parameters. This is not feasible for the non-linear structure of trees. In addition, standard tree models like RF and ETC are prone to client collapse during traffic bursts. To overcome both of these limitations, we customised a tree-pooling framework and introduced quantile-based splitting to protect weak clients during bursts.

2.4 Dataset development

In the dataset development section, the validity of ML models depends on the relevance of training data. Bahashwan et al. [21] introduced the HLD-DDoSDN dataset to reflect actual SDN traffic, while Yungaicela-Naula et al. [22] contributed the SDN-SlowRate-DDoS dataset captured from a physical testbed. Chahal et al. [23] provided a taxonomy of defense strategies specifically for SDN-enabled clouds, and Alasali [24] and Dakkak further categorized approximately 70 mechanisms to identify future research avenues.

Systematic reviews have identified ongoing gaps; Ali et al. [25] highlighted the prevalence of skewed datasets and the lack of multiclass classification. Karnani et al. [26] surveyed low-rate and high-rate DDoS, noting that many solutions fail to address the "single point of failure" in single-controller topologies. Finally, Ahuja et al. [27] also provided a widely used Mendeley DDoS SDN dataset.

Most existing studies evaluate models under idealized IID conditions. This research addresses this by federating the Mendeley dataset across ten clients, simulating it and testing client wise accuracy, convergence stability, class imbalance and bursty client scenarios.

3. METHODOLOGY

Figure 1 presents the layered SDN federated architecture used in this work. The architecture is designed with privacy-preserving network security in mind. Raw flow records are never transmitted beyond the originating edge compute node. These records may contain sensitive network intelligence such as source IP and destination IP addresses, port numbers and payload-derived features. Only serialised decision tree structures travel between tiers. These structures contain feature indices, threshold values, and leaf class predictions. They carry no direct information about individual flows. Because of this, membership inference attacks become substantially more difficult to carry out.

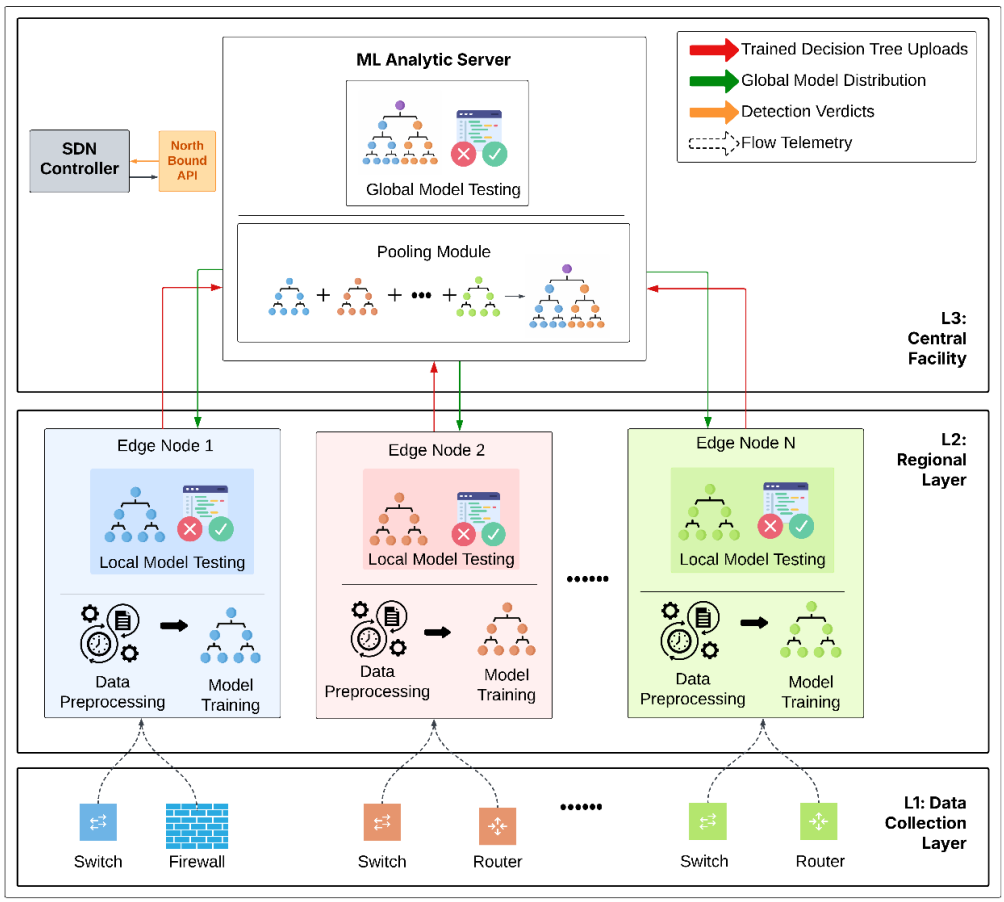

Figure 1. Proposed three-layer hierarchical Federated learning (FL) architecture for Distributed Denial of Service (DDoS) attack detection in Software-Defined Networking (SDN) environments

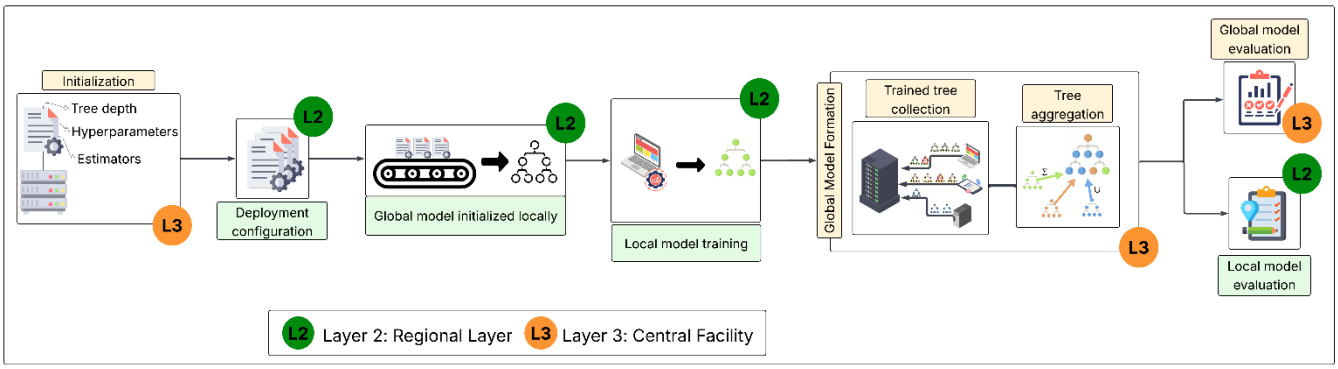

Figure 2. Federated learning workflow for the proposed Distributed Denial of Service (DDoS) detection framework, spanning Layer 2 (Regional Layer) and Layer 3 (Central Facility)

Communication volume is further reduced by the compactness of tree structures relative to raw flow telemetry. A tree of depth 5 has at most 63 nodes, each requiring a feature index, a threshold value, and a class prediction, a total of approximately 1.5 KB per tree in 32-bit floating-point representation. With 100 trees per edge compute node, each FL communication round involves under 150 KB per client, orders of magnitude smaller than the raw flow dataset accumulated during the same training window, which typically spans tens of megabytes. Additionally, a failure or compromise of the central facility does not immediately disable detection: edge compute nodes continue local inference with the last received global model until connectivity is restored.

In the architecture there are 3 logical layers, each with its own responsibilities and interfaces. The description and function of each layer is provided below.

3.1 Level 1 (L1): Data collection layer

Layer 1 is the data collection layer in our SDN federated learning architecture. At L1, many distributed sites work together. Each site runs its own SDN components making sure every corner of the network is covered.

SDN components are OpenFlow-capable switches, routers, and firewalls that forward traffic and export flow telemetry. In a federated learning environment, they act as data collectors for regional edge processing.

These devices use sFlow hardware sampling or IPFIX/NetFlow export to stream flow records to local collectors.

No ML inference or training runs on data-plane hardware; flow export adds negligible overhead to switch CPUs and is supported natively by modern network ASICs.

3.2 Level 2 (L2): Regional layer

Layer 2 consists of multiple edge compute nodes, each associated with a cluster of local network devices. Depending on the deployment context, these nodes can take several physical forms. A rack-mounted server in a branch office or campus equipment room, co-located with access switches, is one option. Another is an NFV appliance running virtualised network functions alongside the switch fabric. A next-generation firewall platform with sufficient general-purpose CPU and memory to host a containerised ML workload is also a viable form. Finally, an embedded compute module integrated into a modular switch chassis can serve the same purpose.

Each edge compute node receives flow records from its associated switches via sFlow or IPFIX, maintains a local dataset, trains a local ML model periodically, and communicates trained trees to the central ML analytics server. Each site also evaluates its model performance locally.

The ML analytics server is a command center that orchestrates training and evaluation. In cases where regional sites may not finish training at the same time due to various factors like resource availability, latency in communication and environmental factors, or a site failture, the analytics server waits for a minimum number of client responses to begin with next stage.

The ML analytics server also makes sure that only after training and evaluation instructions are executed, a response is sent from the edge compute node. This maintains the robustness of the framework.

3.3 Level 3 (L3): Central facility

Level 3 is the central facility. It hosts the ML analytics server, which works closely with the SDN control plane. This facility is typically located at the organisation's primary data centre or network operations centre. It contains both the SDN controller and the ML analytics server as distinct, co-located systems.

The SDN controller running platforms such as ONOS, OpenDaylight, or Ryu is responsible exclusively for its core function: maintaining the global network view, computing forwarding paths and installing flow rules via the OpenFlow southbound interface.

The ML analytics server is a separate system integrated with the controller via the northbound API. It receives trained tree models from edge compute nodes, performs global aggregation by setting up connection with model pooling modules for global model construction. It also maintains a global test dataset for model validation. The developed model is also sent back to L2 nodes for local evaluation.

It can also deliver detection verdicts and mitigation recommendations to the controller, which translates them into flow rule installations.

Figure 2 describes a detailed low-level workflow of the ML analytics server.

3.4 Rationale for architectural separation

Separating the ML analytics server from the SDN controller is not merely an implementation convenience. It is in fact an

architecturally necessary decision for production viability. During a DDoS event, the controller is under maximum stress. It must process PacketIn events at a high rate, recompute forwarding tables, and install mitigation rules with sub-second latency. Running tree aggregation across hundreds of received estimators or evaluating a global forest on incoming telemetry at the same time would compete for CPU and memory on the same host. This could delay the very mitigation actions the detection system is designed to trigger. Separation solves this. It allows the ML server to be independently scaled, updated without controller downtime, and monitored with ML-specific tools, all without any risk to network stability.

The decision to run edge FL clients on associated compute infrastructure rather than on switch operating systems reflects a fundamental hardware constraint. Network switch ASICs are optimised for wire-speed packet forwarding. Their general-purpose CPUs are reserved for control-plane tasks such as spanning tree, LLDP, and OpenFlow agent operation. A tree model with 100 trees of depth 5 is computationally modest by server standards. But it would be entirely impractical on most switch CPUs. Edge compute nodes are a different story. Even modest ones with four CPU cores and 8 GB RAM can execute training on thousands of flow records in seconds. Switch hardware remains completely unaffected.

3.5 Fast Entropy Quantile Forest

In addition to the federated architecture, our work proposes a novel tree-based learning model designed explicitly for federated DDoS detection in SDN environments, inspired by Ahuja et al. [7] and quantile-based split selection. Algorithm 1 and 2 describes our novel tree constructions.

Algorithm 1. Fast Entropy Quantile Forest (FEQF)

Input: Training data X, class labels y, number of estimators E
Output: FEQF model F
Initialise an empty ensemble F
for $e = 1$ to E **do**
Draw a bootstrap sample from (X, y) if bootstrapping is enabled
Train a Fast Entropy Quantile Decision Tree using Algorithm 2
Add the trained tree to ensemble F
end for
for each test instance x **do**
Obtain predictions from all trees in F
Assign the final class label using majority voting
end for
return the trained ensemble model F

Algorithm 2. Fast Entropy Quantile Forest (FEQF) - Decision Tree Construction

Input: Training data X, class labels y, maximum depth D, minimum samples per split S
Output: Trained decision tree T
if depth $\geq D$ **or** $|y| < S$ or all samples belong to one class then
Create a leaf node with the majority class label
return
end if
Select a subset of candidate features using the predefined feature selection strategy
for each selected feature f **do**
Compute a fixed set of quantile-based thresholds from the distribution of f
for each threshold q **do**
Partition the data into left and right subsets
Compute fast entropy for the parent node
Compute fast entropy for the left and right child nodes
Compute entropy gain using the fast entropy criterion
end for
end for
Select the feature–threshold pair ($f*$, $q*$) that maximises entropy gain
Create an internal decision node using ($f*$, $q*$)
Recursively apply this procedure to the left and right subsets
return the constructed decision tree T

Furthermore, split thresholds are derived from percentile statistics of the feature distribution, where each threshold corresponds to a value below which a specified proportion of the data resides. Specifically, the model evaluates only four candidate split points for every feature: the 20th, 40th, 60th, and 80th percentiles. This "Fast Split Search" approach significantly reduces computation compared to evaluating every unique value as a potential threshold.

The global model is formed through ensemble aggregation, where local decision trees are merged into a single combined model. A subset of decision trees is randomly sampled from each client's local model and aggregated to form a unified global forest. Specifically, each client contributes a fixed number of 10 trees, which are randomly selected from its locally trained forest. The total number of trees in the global model is maintained at approximately 100. There is a total of 10 clients participating.

Key model hyperparameters are optimized using Particle Swarm Optimization (PSO). The maximum tree depth is selected within the range of 5-20 (optimized value: 16), while the minimum split size is determined within 2-20 (optimized value: 4). The number of trees is tuned within 5-30, with the final model utilizing 30 estimators. Feature selection follows a square-root strategy based on the total number of features and max-features ratio ranging from 0.3 to 1.0 (optimized value: 0.74).

Two rounds of communication are required per model update. First is a collection where each client locally trains its own estimators; these local models are collected by a central server. Second is redistribution, where the server aggregates these estimators into a combined forest and sends the global model back to the clients for evaluation or use.

4. EXPERIMENTAL RESULTS AND ANALYSIS

This section details a comprehensive evaluation of the proposed model within a federated learning framework. The experiments are designed to analyse accuracy, stability, robustness under class imbalance, and resilience to bursty clients. The proposed approach is compared against two strong tree-based baselines: RF and ETC. Experiments were carried out using the Mendeley DDoS attack SDN Dataset [27]. About the dataset, network simulation is carried out on ten different topologies and includes benign TCP, UDP, and ICMP traffic. This also includes malicious traffic such as TCP SYN flood, UDP flood, and ICMP attack. It contains 104,345 records and

23 features.

We have performed preprocessing and anomaly detection steps on the dataset before using it. In the dataset we have found that 506 rows contain some empty data out of 104,345 rows and 23 columns; hence, we removed those irrelevant 506 rows. This was done to ensure algorithmic stability and prevent errors.

The remaining data in the dataset are 103,839 rows and 23 columns. After removing some unwanted features and the same features, the remaining dataset contains 18 features. Also, some features have many zero values, so we removed them as well.

The initial 23 features were reduced to 15. Features with many zero values or those that were redundant provided no information gain. In tree-based models, such features do not contribute to meaningful splits and only increase the computational overhead of the fast entropy metric.

A total of 5222 duplicated rows were removed. Redundant records can lead to over-representation bias, where a model overfits specific, repeated traffic patterns.

We also used an Isolation Forest anomaly detection algorithm. We removed 9861 rows that were marked as outliers. Isolation Forest was employed to remove extreme anomalies that are likely to represent measurement errors or transient network noise rather than the core characteristics of a DDoS signature. This step is critical for defining meaningful decision boundaries that are not skewed by a few extreme outliers.

After performing the above, the dataset finally contained around 88765 rows and 15 columns. This dataset was converted into a federated dataset for 10 clients. The distribution made sure that it was able to mimic 10 independent clients. Each partition was assigned to individual clients. No two clients received the same data partition. Each partition has 6212 rows for training and 2662 for testing.

To ensure the objectivity of the evaluation, the preprocessing did not aim to clean the attack patterns into an idealized state. Instead, it focused on technical data quality.

Table 1. Evaluation metrics: Definitions and detection relevance

Metrics	Description
Accuracy = $\frac{TP + TN}{TP + TN + FP + FN}$	This metric shows how often the model is correct.
F1-score (Attack Class) = $2 \times \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$	In DDoS detection, the attack traffic is a rare but crucial class, so excelling in it makes this metric especially valuable.
AUC (Area Under ROC Curve)= $\int_0^1 \text{TPR(FPR)}\ d(\text{FPR})$	Independent of any fixed threshold, it evaluates ranking performance and remains robust under class imbalance.
Stability Risk (Standard Deviation) $\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$	Quantifies convergence consistency across clients and communication rounds.

4.1 Evaluation metrics

To ensure a fair and informative evaluation in highly imbalanced, non-IID settings, multiple complementary metrics are used, as described in Table 1. The experiments were repeated multiple times, consistently yielding similar results with no significant variation.

4.2 Client-wise global accuracy analysis

The aim here is to understand how evenly each client supports the global model during training. As shown in Figure 3, our approach maintains a stable accuracy of about 97% across all clients, with very little variation. RF reaches a similar average accuracy, but its performance fluctuates slightly more among clients. Extra Trees (ET) performs far worse, with clear drops in accuracy.

The stable 97% accuracy across all clients comes down to one key design decision: quantile-based splitting. Standard decision trees evaluate every unique value as a potential split point. This makes them highly sensitive to local noise or biased data on any given client. FEQF avoids this problem. It limits the search to four fixed percentiles: the 20th, 40th, 60th, and 80th. Decision boundaries are therefore grounded in the broader data distribution. This is why the global model holds up well against individual client variations.

The simulations here represent distributed L2 layer. The results validate that L2 edge compute nodes can effectively train models on local traffic statistics to support a consistent global defense.

Figure 3. Client-wise global accuracy comparison

4.3 Stability risk trade-off analysis

In federated deployment, achieving high accuracy alone is not enough. Stable convergence is also very important. Here, the experiment focuses on the balance between average server accuracy and stability risk, which is measured using standard deviation. A desirable model is the one that has high accuracy and stability during the aggregation phase. From Figure 4, we can see that our model shows very stable convergence, with a low variance of 0.0035. This indicates low aggregation risk. ET exhibits much higher variance, around 0.0065, which suggests unstable convergence and higher risk during model aggregation. RF falls in between, with a variance close to 0.0040, reflecting moderate stability and moderate aggregation risk.

The low aggregation risk is a combined result of PSO, quantile thresholds, and fast entropy. While PSO ensures the forest has the right capacity (depth and size), quantile thresholds provide the structural rigidity needed to ignore local outliers, and fast entropy provides the mathematical precision to select the most stable features. Together, they prevent the unstable tails and high variance that characterize the baseline models during the aggregation phase.

The ML analytics server in L3 layer is responsible for coordinating learning rounds and utilizing the pooling function to combine trees. The simulation results provide quantitative proof of L3's efficacy.

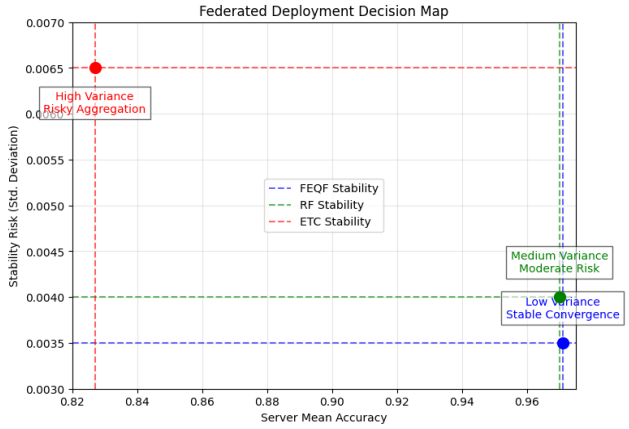

Figure 4. Federated deployment decision map

4.4 Performance under class imbalance (1:100)

This experiment evaluates local client performance under a severe 1:100 attack-to-benign imbalance. These simulations represent a quantitative evaluation of nodes present at L2 layer.

From Figure 5, we see that:

- Proposed model achieves consistently high F1-scores greater than 0.87 across all clients, with a peak value at 0.95, as shown in Figure 3.
- RF shows unstable behaviour among clients, with a peak value of 0.80.
- ET performs inconsistently and fails to generalise with a peak value at 0.71.

The proposed FEQF model is susceptible to minority-class decision boundaries, enabling more effective learning from scarce attack samples. Consequently, the model achieves superior attack detection performance even under extreme class imbalance.

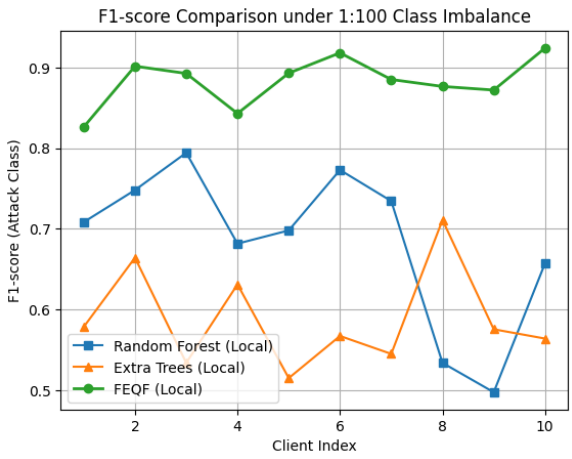

Figure 5. F1-score comparison under 1:100 class imbalance

From Figure 6 we see that Area Under the ROC Curve (AUC) performance of different models. FEQF has the highest score, 0.85. This shows strong classification ability. It can clearly separate attack and benign traffic. In comparison, RF performs at a lower level. Its highest AUC reaches around 0.72, showing limited ranking capability across clients. ET performed even weaker, with AUC values often close to random guessing 0.5, and only occasional peaks approaching 0.64. It shows unreliable classification performance.

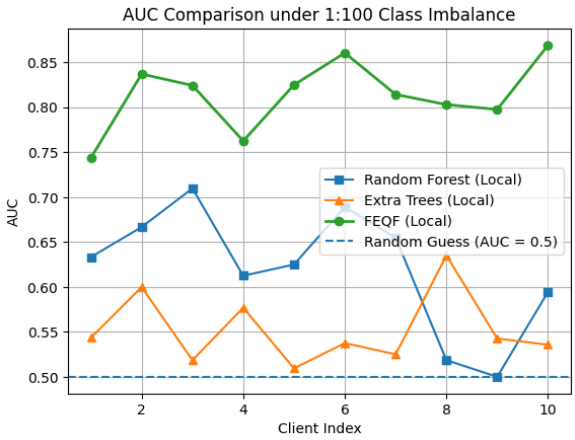

Figure 6. Area Under the ROC Curve (AUC) comparison under 1:100 class imbalance

FEQF superior performance under extreme imbalance is driven by the Fast Entropy adaptation. This entropy metric is specifically designed for DDoS and network traffic scenarios to be susceptible to minority-class signals. While standard RFs may focus on the majority class (benign traffic) to reduce overall error, the Fast Entropy calculation effectively identifies the high-information gain provided by scarce attack samples. Additionally, the Quantile Thresholds ensure that candidate splits are tested across the entire distribution, preventing the minority samples from being washed out by majority class split points.

4.5 Burst client analysis

A minority of nodes present at L2 layer may experience attack-induced traffic surges without overwhelming the entire federation. To evaluate under bursty conditions we used 30% bursty clients to enable meaningful robustness evaluation while preserving sufficient stable clients for effective global learning. From Figure 7, ET experiences client collapse, with the weakest client dropping to an F1-score of 0.71. RF exhibits unstable tails, that is, performance degradation in worst-case clients.

Our model maintains performance for the weakest clients, achieving 0.93 F1-score. This shows even the weakest client using our proposed model is stable compared to baseline models.

The reason FEQF prevents client collapse during traffic surges is the inherent robustness of Percentile-based thresholds. In a bursty environment, 30% of clients experience extreme outliers in traffic volume.

Unlike standard trees that may split on specific outlier values during a traffic surge, FEQF uses percentile-based splitting. Because these thresholds are rank-dependent, extreme spikes in the top percentages of traffic do not drastically shift the decision boundaries, preventing the model from overfitting to the burst noise.

Also, fast entropy used in FEQF allows the model to prioritize information gain from sparse attack signatures, even amidst the high-volume noise of a traffic surge. This ensures that the model continues to identify threats effectively,

regardless of the fluctuating traffic levels on the weakest nodes.

Furthermore, the PSO-tuned max-features ratio (0.74) and 30 estimators ensure the ensemble has enough diversity to maintain performance even when a minority of nodes experience high-stress conditions.

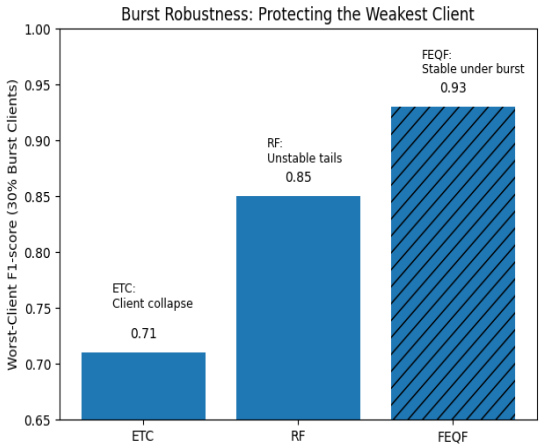

Figure 7. Burst robustness: Protecting weak clients

Table 2 summarizes the comparative peak performance of the proposed model against baseline models.

Table 2. Comparison of peak performance metrics across models

Metric	FEQF (Proposed)	Random Forest (RF)	Extra Trees (ET)
Client-Wise Global Accuracy	97% (Consistent)	97% (Fluctuating)	84%
Stability Risk	0.0035	0.0040	0.0065
F1 (Imbalance 1:100)	0.95	0.80	0.71
AUC Score	0.85	0.72	0.64
F1 (Burst Clients)	0.93	0.85	0.71
Key Mechanism	PSO + Fast Entropy + Quantiles	Standard Gini/Entropy	Random Splitting

Note: Fast Entropy Quantile Forest (FEQF).

5. CONCLUSION

In this study, we introduced the FEQF, a specialized tree-based model designed for federated DDoS detection in SDN. Our findings indicate that FEQF demonstrates specific performance advantages over the tested baselines, RF and ETC.

The FEQF model is primarily applicable to federated SDN environments where privacy requirements prevent the sharing of raw traffic data. By replacing traditional impurity measures with a fast entropy metric and utilizing quantile-based feature splitting, the model effectively addresses critical network challenges such as attack-induced bursts (F1 score 0.93) and severe class imbalance (F1 score 0.95).

Another technical finding of this work is the model's ability to achieve stable convergence during the global aggregation phase. While traditional tree-based federated models often suffer from high aggregation risk, the FEQF model demonstrates a low stability risk (0.0035).

While the results are promising, it is important to define the current boundaries of this validation.

This research focused exclusively on comparing FEQF against other tree-based federated learners (RF and ETC). It does not account for comparisons against other DL architectures, such as CNNs or GRUs that have different computational and privacy profiles.

The experiments were conducted by federating the Mendeley DDoS attack SDN dataset. While this dataset was partitioned to simulate ten clients, the model's performance on other emerging datasets or live, multi-source network traffic remains to be verified. Experiments were performed within a simulated federated environment rather than a live SDN testbed.

Also, the synchronization of training rounds across distributed L2 nodes remains sensitive to regional latency and resource availability, which was not simulated but visible in real time deployments.

6. FUTURE DIRECTIONS

Future directions aim to address several identified challenges, though not limited to them.

Firstly, the current version of FEQF does not include mechanisms to defend against Byzantine attacks or malicious model poisoning during the aggregation phase.

Secondly, FEQF relies on statistical feature snapshots; it does not yet integrate temporal learning mechanisms (like LSTM) necessary to detect low-rate or slowly evolving DDoS attacks.

Thirdly, the aggregation currently assumes a uniform tree construction across nodes. Exploring weighted tree aggregations would be beneficial for environments with highly heterogeneous client resources.

Fourthly, this study focused on DDoS flooding attacks. Future iterations must expand to include multi-class detection for other threats such as spoofing, probing, or insider attacks.

Fifthly, our experiments compared FEQF with standard scikit-learn RF and ETC. As part of future work, this comparison could be extended to include other ML models as well as optimized versions of RF and ETC.

Finally, future work should also focus on implementing model pruning and compression to further manage the complexity of pooled global models and minimize deployment delays. This ensures that the system remains both robust and scalable for large-scale network infrastructures.

REFERENCES

[1] No, G., Ra, I. (2009). An efficient and reliable DDoS attack detection using a fast entropy computation method. In 2009 9th International Symposium on Communications and Information Technology, Icheon, Korea (South), pp. 1223-1228. https://doi.org/10.1109/ISCIT.2009.5341118

[2] FLEXible-FL Team. (2023). Federated random forest with FLEX. GitHub repository. https://github.com/FLEXible-FL/flex-trees/blob/main/notebooks/Federated%20Random%20Forest%20with%20FLEX.ipynb, accessed on Jan. 3, 2026.

[3] Hnamte, V., Hussain, J. (2023). An efficient DDoS attack

detection mechanism in SDN environment. International Journal of Information Technology, 15(5): 2623-2636. https://doi.org/10.1007/s41870-023-01332-5

[4] Al-Dunainawi, Y., Al-Kaseem, B.R., Al-Raweshidy, H.S. (2023). Optimized artificial intelligence model for DDoS detection in SDN environment. IEEE Access, 11: 106733-106748. https://doi.org/10.1109/ACCESS.2023.3319214

[5] Elshewey, A.M., Abbas, S., Osman, A.M., Aldakheel, E.A., Fouad, Y. (2025). DDoS classification of network traffic in software defined networking SDN using a hybrid convolutional and gated recurrent neural network. Scientific Reports, 15(1): 29122. https://doi.org/10.1038/s41598-025-13754-1

[6] Wang, K., Fu, Y., Duan, X., Liu, T. (2024). Detection and mitigation of DDoS attacks based on multi-dimensional characteristics in SDN. Scientific Reports, 14(1): 16421. https://doi.org/10.1038/s41598-024-66907-z

[7] Ahuja, N., Mukhopadhyay, D., Singal, G. (2024). DDoS attack traffic classification in SDN using deep learning. Personal and Ubiquitous Computing, 28(2): 417-429. https://doi.org/10.1007/s00779-023-01785-2

[8] Alashhab, A.A., Zahid, M.S., Isyaku, B., Elnour, A.A., Nagmeldin, W., Abdelmaboud, A., Abdullah, T.A.A., Maiwada, U.D. (2024). Enhancing DDoS attack detection and mitigation in SDN using an ensemble online machine learning model. IEEE Access, 12: 51630-51649. https://doi.org/10.1109/ACCESS.2024.3384398

[9] Alubaidan, H., Alzaher, R., AlQhatani, M., Mohammed, R. (2023). DDoS detection in Software-Defined Network (SDN) using machine learning. International Journal on Cybernetics & Informatics, 12(04): 93-104. https://doi.org/10.5121/ijci.2023.120408

[10] Karthika, P., Arockiasamy, K. (2023). Simulation of SDN in Mininet and detection of DDoS attack using machine learning. Bulletin of Electrical Engineering and Informatics, 12(3): 1797-1805. https://doi.org/10.11591/eei.v12i3.5232

[11] Guesmi, H., Saidane, L.A. (2017). Using SDN approach to secure cloud servers against flooding based DDoS attacks. In 2017 25th International Conference on Systems Engineering (ICSEng), Las Vegas, NV, USA, pp. 309-315. https://doi.org/10.1109/ICSEng.2017.31

[12] Eliyan, L.F., Di Pietro, R. (2023). Demi: A solution to detect and mitigate DoS attacks in SDN. IEEE Access, 11: 82477-82495. https://doi.org/10.1109/ACCESS.2023.3301994

[13] Swami, R., Dave, M., Ranga, V. (2023). IQR-based approach for DDoS detection and mitigation in SDN. Defence Technology, 25: 76-87. https://doi.org/10.1016/j.dt.2022.10.003

[14] Khedr, W.I., Gouda, A.E., Mohamed, E.R. (2023). FMDADM: A multi-layer DDoS attack detection and mitigation framework using machine learning for stateful SDN-based IoT networks. IEEE Access, 11: 28934-28954. https://doi.org/10.1109/ACCESS.2023.3260256

[15] Garba, U.H., Toosi, A.N., Pasha, M.F., Khan, S. (2024). SDN-based detection and mitigation of DDoS attacks on smart homes. Computer Communications, 221: 29-41. https://doi.org/10.1016/j.comcom.2024.04.001

[16] Oleiwi, W.K., Hussein, A.M., Gheni, H.Q., Al-Qurabat, A.K.M. (2025). Framework for enhanced privacy-preserving consensus system for distributed SDN: Redefining security for the east-west interface. International Journal of Safety and Security Engineering, 15(10): 2093-2102. https://doi.org/10.18280/ijsse.151012

[17] Mortatha, M.B., Abdulah, D.A. (2025). Optimizing IoT intrusion detection and attack classification via WOA-based feature reduction and boosting algorithms. International Journal of Safety and Security Engineering, 15(10): 2169-2176. https://doi.org/10.18280/ijsse.151019

[18] Gebremeskel, T.G., Gemeda, K.A., Krishna, T.G., Ramulu, P.J. (2023). DDoS attack detection and classification using hybrid model for multicontroller SDN. Wireless Communications and Mobile Computing, 2023(1): 9965945. https://doi.org/10.1155/2023/9965945

[19] Ma, R., Wang, Q., Bu, X., Chen, X. (2023). Real-time detection of DDoS attacks based on random forest in SDN. Applied Sciences, 13(13): 7872. https://doi.org/10.3390/app13137872

[20] Hauschild, A.C., Lemanczyk, M., Matschinske, J., Frisch, T., Zolotareva, O., Holzinger, A., Baumbach, J., Heider, D. (2022). Federated random forests can improve local performance of predictive models for various healthcare applications. Bioinformatics, 38(8): 2278-2286. https://doi.org/10.1093/bioinformatics/btac065

[21] Bahashwan, A.A., Anbar, M., Manickam, S., Issa, G., Aladaileh, M.A., Alabsi, B.A., Rihan, S.D.A. (2024). HLD-DDoSDN: High and low-rates dataset-based DDoS attacks against SDN. PLOS One, 19(2): e0297548. https://doi.org/10.1371/journal.pone.0297548

[22] Yungaicela-Naula, N.M., Vargas-Rosales, C., Perez-Diaz, J.A., Jacob, E., Martinez-Cagnazzo, C. (2023). Physical assessment of an SDN-based security framework for DDoS attack mitigation: Introducing the SDN-SlowRate-DDoS dataset. IEEE Access, 11: 46820-46831. https://doi.org/10.1109/ACCESS.2023.3274577

[23] Chahal, J.K., Bhandari, A., Behal, S. (2024). DDoS attacks & defense mechanisms in SDN-enabled cloud: Taxonomy, review and research challenges. Computer Science Review, 53: 100644. https://doi.org/10.1016/j.cosrev.2024.100644

[24] Alasali, T., Dakkak, O. (2023). Exploring the landscape of SDN-based DDoS defense: A holistic examination of detection and mitigation approaches, research gaps and promising avenues for future exploration. International Journal of Advanced Natural Sciences and Engineering Researches, 7(4): 327-349. https://doi.org/10.59287/ijanser.726

[25] Ali, T.E., Chong, Y.W., Manickam, S. (2023). Machine learning techniques to detect a DDoS attack in SDN: A systematic review. Applied Sciences, 13(5): 3183. https://doi.org/10.3390/app13053183

[26] Karnani, S., Agrawal, N., Kumar, R. (2024). A comprehensive survey on low-rate and high-rate DDoS defense approaches in SDN: Taxonomy, research challenges, and opportunities. Multimedia Tools and Applications, 83(12): 35253-35306. https://doi.org/10.1007/s11042-023-16781-0

[27] Ahuja, N., Singal, G., Mukhopadhyay, D. (2020). DDoS attack SDN dataset. Mendeley Data. https://doi.org/10.17632/jxpfjc64kr.1