Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

FL-SNNIDS: Federated Spiking Neural Networks for Energy-Efficient Intrusion Detection in Agricultural SDN-IoT Networks

Selma Amrani
University Ferhat Abbas of Setif
Chirihane Gherbi
University Ferhat Abbas of Setif
Khedidja Medani
University Ferhat Abbas of Setif
Hakim Mabed
hmabed@femto-st.fr

Université Marie et Louis Pasteur

Research Article

Keywords: Software-Defined Networking, SNNs, Federated Learning (FL), IDS, Smart Agriculture, Farm-Flow

Posted Date: July 30th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10204905/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

FL-SNNIDS: Federated Spiking Neural Networks for Energy-Efficient Intrusion Detection in Agricultural SDN-IoT Networks

Selma Amrani
LRSD Laboratory, Faculty of Sciences
Ferhat Abbas University of Setif 1
Setif, Algeria
selma.amrani@univ-setif.dz

Chirihane Gherbi
LRSD Laboratory, Faculty of Sciences
Ferhat Abbas University of Setif 1
Setif, Algeria
chirihane.gherbi@univ-setif.dz

Khedidja Medani
LRSD Laboratory, Faculty of Sciences
Arabic literature and language department
Ferhat Abbas University of Setif 1
Mouhamed Lamine Debaghine Setif 2 University
Setif, Algeria
k.medani@univ-setif2.dz

Hakim Mabed (corresponding author)
DISC/FEMTO-ST
Université Marie et Louis Pasteur
Montbéliard, France.
hmabed@femto-st.fr

June 30, 2026

Abstract

The modernization of smart agriculture increasingly relies on Software-Defined Networking (SDN)-enabled IoT infrastructures to support distributed and data-driven agricultural operations. However, the deployment of heterogeneous and resource-constrained edge devices expands the attack surface while introducing strict requirements related to inference latency, energy efficiency, scalability, and data locality. Existing intrusion detection systems often rely on computationally intensive deep learning models, which may limit their deployment in distributed agricultural IoT environments.

This paper focuses on designing a unified IDS for SDN-based IoT networks in smart agriculture, capable of addressing the extreme demands of an open environment—namely, exposure to external attacks and energy autonomy constraints. To address these challenges, this paper presents FL-SNNIDS, a federated Spiking Neural Network-based intrusion detection framework. The proposed architecture deploys SNN-based intrusion detection models at local agricultural SDN controllers and uses Federated Learning to collaboratively train a global model without exchanging raw network traffic data. By exploiting sparse spike activity and event-driven computation, the local SNN aims to provide accurate and lightweight intrusion detection with reduced inference latency and operation-level computational cost.

The proposed approach was evaluated on the SDN-IoT and Farm-Flow datasets, achieving accuracies of 99.93% and 99.68%, respectively. The federated framework preserved competitive detection performance across distributed clients while maintaining low inference latency and reduced operation-level analytical energy estimates.

An external cross-dataset evaluation was additionally conducted using the ASEADOS-SDN-IoT dataset under few-shot adaptation settings. The results demonstrated effective adaptation for dominant traffic categories. Overall, the findings indicate that our proposed **FL-SNNIDS offers optimal trade-off between detection performance, inference latency, energy consumption and operation-level computational complexity.**

Keywords:
Software-Defined Networking, SNNs, Federated Learning (FL), IDS, Smart Agriculture, Farm-Flow.

1 Introduction

Smart Agriculture, also known as Smart Farming, represents a transformative shift in food production by leveraging the Internet of Things (IoT) to enable real-time monitoring, predictive analytics, and automation of agricultural operations. This technological evolution has significantly enhanced efficiency, resource optimization, and crop yields, and reshaped traditional farming practices and supply chains [1–3]. Central to this transformation is the integration of Software-Defined Networking (SDN), which provides dynamic, centralized, and flexible management of complex and heterogeneous networks underlying modern agricultural environments [1].

Despite these advancements, several challenges remain in securing SDN-based agricultural IoT systems. A recent systematic literature review [4] extensively analyzed Machine Learning applications in this domain, identifying critical research gaps, particularly in energy efficiency and decentralized security. These limitations directly motivate the present work, which aims to bridge these gaps through a neuromorphic and federated approach.

In particular, the convergence of IoT and SDN introduces a broad and multifaceted attack surface, making network security a critical concern. The widespread deployment of resource-constrained IoT devices, the heterogeneity of data streams, and the central role of the SDN controller collectively render smart agriculture ecosystems highly targeted by cyber threats [1–3]. Typical vulnerabilities include weak authentication, lack of standardization, and outdated software, which can be exploited for malicious code injection, denial-of-service (DoS) attacks, data breaches, and reconnaissance activities [2]. Such attacks not only jeopardize the integrity and availability of agricultural data and operations but can also result in significant physical and economic consequences, including crop loss, supply chain disruption, and food insecurity [1, 2].

Deep Learning (DL)-based Intrusion Detection Systems (IDS) have been widely adopted to secure SDN-IoT agricultural networks due to their high accuracy in detecting sophisticated cyber threats. Nevertheless, deploying these models in agricultural environments faces critical challenges related to energy efficiency, data privacy, and network architecture. Traditional DL models, such as LSTM, GRU, and CNN, are computationally intensive, resulting in high latency and excessive energy consumption. makes them impractical for resource-constrained IoT devices, where power and computational resources are limited, and real-time detection is required [5].

Centralized training of DL-based IDS requires aggregating raw network traffic from distributed farm devices to a central server, raising significant privacy concerns. Sensitive agricultural data may be exposed during transmission or storage, increasing the risk of breaches and regulatory non-compliance. Moreover, in SDN architectures, centralizing IDS operations on a single controller introduces a computation and communication bottleneck. The controller becomes overloaded, impairing network scalability and creating a single point of failure, which can be exploited by attackers and limits the network's ability to respond to dynamic threats [5, 6].

To overcome these limitations, we introduce FL-SNNIDS (Federated Spiking Neural Network Intrusion Detection System), a decentralized security framework specifically engineered for resource-constrained SDN-IoT agricultural infrastructures. By integrating local Spiking Neural Networks (SNNs), the framework leverages asynchronous, event-driven processing to achieve high-fidelity inference with minimal power dissipation. Furthermore, the use of Federated Learning (FL), to aggregate the training results of the local SNNs, significantly alleviates the computational and communication overhead on the central SDN controller while enhancing the overall system's scalability and data privacy.

To summarize, the main contributions of this paper are as follows:

1. Federated SNN-based IDS architecture: We design FL-SNNIDS, a distributed intrusion detection architecture in which agricultural SDN controllers collaboratively train lightweight SNN models without exchanging raw network traffic data.

2. Resource-aware local intrusion detection: We develop an SNN-based multiclass classifier that exploits sparse spike activity and event-driven processing to provide low-latency intrusion detection at the agricultural network edge.

3. Unified experimental comparison: We evaluate the proposed approach against conventional machine-learning and deep-learning baselines using identical preprocessing, training, validation, and test partitions on the SDN-IoT and Farm-Flow datasets. The evaluation considers detection performance, inference latency, model complexity, and operation-level analytical energy estimates.

4. Federated and cross-domain evaluation: We assess the behaviour of the proposed framework under distributed training and examine its transferability to the ASEADOS-SDN-IoT dataset using limited target-domain adaptation. The cross-dataset results identify both the adaptation potential and the limitations of the model for previously unseen minority attack classes.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 details the architecture of our FL-SNNIDS model. Section 4 presents the methodology, and Section 5 describes the experimental setup. Section 6 presents and analyzes the experimental results. Finally, Section 7 concludes the paper and outlines directions for future research.

2 Related Work

The security of SDN-enabled IoT networks has been extensively studied, primarily through the lens of traditional Deep Learning (DL). However, the convergence of energy efficiency and privacy preservation remains a significant challenge. This section reviews existing literature categorized by detection methodology.

2.1 Deep Learning-based IDS for SDN-IoT

Recent advancements have focused on capturing temporal and spatial features of network traffic. Chaganti et al. [7] employed Long Short-Term Memory (LSTM) networks, achieving 97.1% accuracy, while Sri Vidhya et al. [8] utilized Bidirectional LSTMs (BiLSTM) to capture long-range dependencies, reaching 92.9% accuracy. Hybrid architectures have also emerged; for instance, Chouhan et al. [9] proposed a CNN-LSTM (HCL) framework to extract both spatial and temporal patterns. Although these sequential models are effective for attack identification, their substantial computational overhead and memory footprint pose a significant barrier to deployment in resource-scarce agricultural IoT environments.

Furthermore, ensemble and feature fusion techniques have been proposed to enhance robustness. Ravi et al. [10, 11] integrated RNNs and GRUs with kernel PCA and ensemble learning, achieving up to 99% detection rates. Similarly, Kumar et al. [12] combined DL with threat-hunting strategies to reach 98.85% accuracy. Despite their high precision, these studies frequently overlook critical operational metrics such as inference latency and energy dissipation, which are vital for real-time edge processing.

2.2 Energy-Efficient Neuromorphic Approaches

To mitigate the power consumption of dense neural networks, Spiking Neural Networks (SNNs) have been investigated as a neuromorphic alternative. Kucik et al. [13] and Kundu et al. [14] demonstrated that SNNs could achieve over 85% energy reduction compared to CNNs on standard datasets like CIFAR-10, with energy costs as low as 3.5×10^{-7} J per inference. While these works underscore the potential of event-driven computation, their application has been largely confined to computer vision, leaving a gap in their evaluation for SDN-IoT traffic analysis.

2.3 Privacy-Preserving Federated Learning

The need for decentralized security has led to the adoption of Federated Learning (FL). Albanbay et al. [6] evaluated FL using DNNs and CNNs, achieving 95.27% accuracy, while Ali et al. [15] integrated blockchain with FL for Industrial IoT security. However, these FL frameworks typically rely on standard DL models as local learners, failing to address the energy consumption at the edge. Additionally, most of these studies do not validate their models on specialized datasets like *Farm-flow* [16], which reflects the unique threat landscape of smart agriculture.

2.4 Research Gap and Contributions

Analysis of the reviewed literature reveals three complementary limitations. First, conventional deep-learning IDS approaches frequently achieve high detection performance but are rarely evaluated in terms of local inference latency and operation-level computational cost. Second, existing SNN studies demonstrate the potential of sparse and event-driven processing, but their evaluation is largely restricted to computer-vision applications rather than SDN-IoT network-flow analysis. Third, current FL-based IDS frameworks primarily rely on dense local learners and seldom investigate whether spike-based models can provide a more favourable trade-off between detection performance and local computational efficiency.

To the best of our knowledge, limited attention has been given to the deployment of SNN-based local intrusion detection models within federated SDN-IoT architectures tailored to smart agriculture. Existing studies also rarely assess agriculture-specific traffic, local inference efficiency, and cross-domain adaptation within a unified experimental setting.

To address these limitations, FL-SNNIDS is evaluated on the SDN-IoT and Farm-Flow datasets using classification performance, inference latency, federated convergence, and operation-level analytical energy estimates. Its transferability is further examined through few-shot adaptation on the ASEADOS-SDN-IoT dataset under domain-shift and limited-data conditions.

Table 1 provides a comparative overview of related work and highlights the significance of our study.

3 Network Architecture

Our proposed FL-SNNIDS architecture is a hierarchical framework designed to bridge the gap between local real-time responsiveness and global collaborative security. As illustrated in Fig. 1, the network is organized into three distinct operational tiers distributed across multiple farm sites:

- Perception and Data Forwarding Tier: At the lowest level, a variety of IoT devices generate heterogeneous telemetry data. These devices are connected to an SDN Switch (OpenFlow), which serves as the data plane. The switch forwards network traffic to the control plane for inspection while maintaining the flexibility to apply security rules dynamically.
- Local Control and Intelligence Tier (The FL Client): Each individual farm operates a Local SDN Controller, which acts as the FL client in the proposed framework. The controller is responsible for:

Table 1: Analysis of Related Works.

Reference	Year	Model	Dataset	Privacy (FL)	Energy Analysis	SDN	Application
[13]	2021	SNN	CIFAR-10	✗	✓	✗	Energy-efficient AI applications
[14]	2021	Optimized SNN (Spike-Thrift)	CIFAR-10, ImageNet	✗	✓	✗	Energy-efficient intrusion detection
[11]	2022	GRU + Ensemble + Feature Fusion	SDN-IoT	✗	✗	✓	SDN-IoT Networks
[10]	2022	RNN + Ensemble (Hybrid)	SDN-IoT, KDD, UNSW-NB15, CICIDS-2017	✗	✗	✓	SDN-IoT Networks
[7]	2023	LSTM	SDN-IoT	✗	✗	✓	SDN-IoT Networks
[8]	2024	BiLSTM	SDN-IoT	✗	✗	✓	SDN-IoT Networks
[9]	2025	CNN + LSTM (HCL)	SDN-IoT	✗	✗	✓	SDN-IoT Networks
[12]	2025	Deep Learning (Threat-Hunting Framework)	SDN-IoT	✗	✗	✓	SDN-IoT Networks
[16]	2025	ML(RF)	Farm-flow	✗	✗	✗	Smart Agriculture
[6]	2025	CNN (FL)	CICIoT2023	✓	✗	✗	IoT Networks
[15]	2025	FL + Blockchain	Industrial IoT	✓	✗	✗	Smart Industry
Our Work	2025	SNN	SDN-IoT Farm-flow	✓	✓	✓	Smart Agriculture

  - Traffic Collection and Monitoring: The SDN controller receives traffic statistics and flow information from the OpenFlow switch for network monitoring and security analysis.
  - IDS Operation and Local Training: The SDN controller performs real-time intrusion detection using the deployed SNN-based IDS. The SNN is trained using farm-specific traffic data.
  - Privacy-Preserving Model Updates: Instead of transmitting sensitive raw traffic data, only locally updated SNN parameters are shared with the global FL aggregation server.
  - Dynamic Rule Enforcement: Based on IDS decisions, the SDN controller can dynamically update forwarding or mitigation rules in the OpenFlow switch to respond to detected threats.

- Global Orchestration Tier: The Global SDN layer hosts the FL Aggregation Server. During the IDS training and validation, the central cloud-based orchestrator manages the global security policy. It iteratively collects model updates from distributed farm sites, and aggregates them to improve the global IDS model, and redistributes the updated model to local controllers. The FL mechanism is disabled during the operational phasis.

This hierarchical structure effectively decentralizes the computational burden both during the IDS training and operational inference tasks. During the operational significantly reducing the overhead on the SDN controller and ensuring a low-power, highly scalable network management solution for smart agriculture.

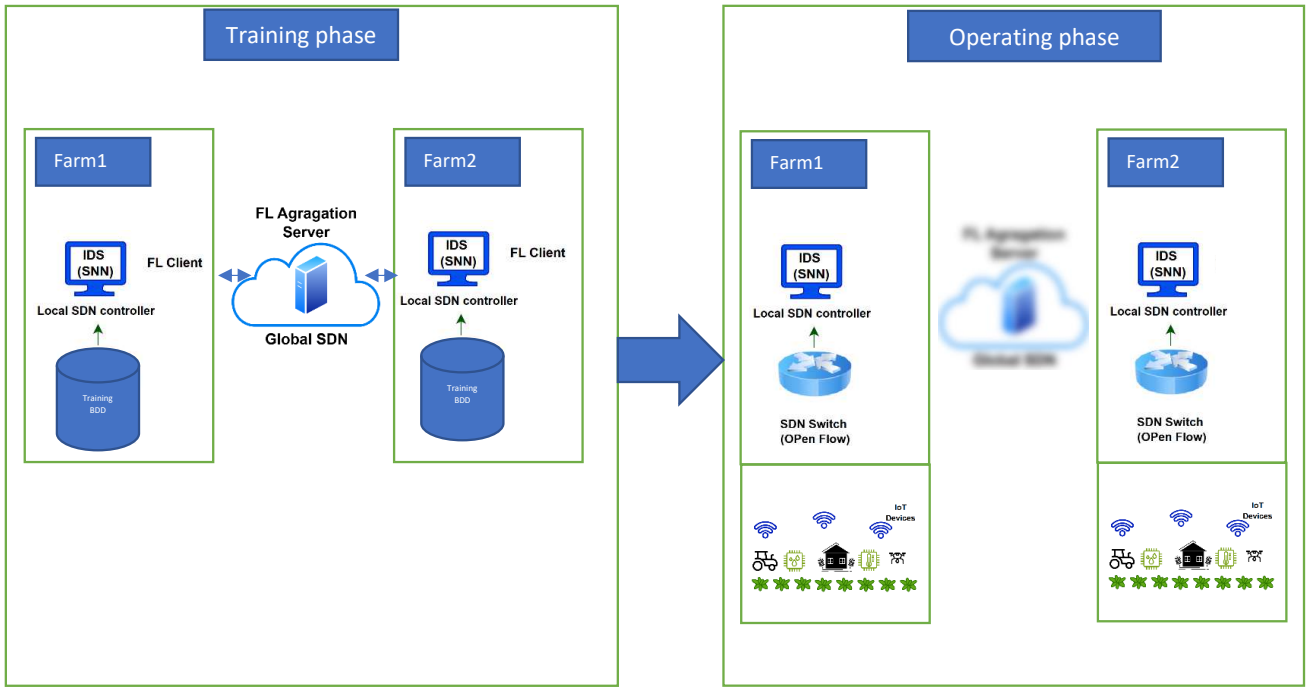


Figure 1: Proposed intrusion detection in decentralized farm system. On the left, the model training architecture and on the right, the distributed IDS operating architecture.

4 Methodology

4.1 Datasets description

"SDN-IoT" dataset meets the growing demand for authentic data to enhance intrusion detection in SDN-IoT environments. This dataset bridges a vital gap by offering a representative samples of the traffic patterns in SDN-IoT networks. "SDN-IoT" dataset consists of two subsets: a static IoT network containing 27.9 million flow records and a dynamic IoT network with 30.2 million records, created using a testbed featuring Mininet, the ONOS SDN controller, and simulated IoT devices. The dataset encompasses benign traffic alongside five unique attack types (Table 2), with 33 SDN-specific features captured in real-time every second (Table 3), making it highly suitable for machine learning-based IDS [17].

Table 2: Type of attacks in SDN-IoT dataset

Class Type	Number of Instances
Normal	35000
DoS	35000
DDoS	35000
Port Scanning	35000
OS and Service Detection	35000
Fuzzing	35000

The "Farm-flow" dataset was developed to provide a realistic benchmark for intrusion detection in Agricultural IoT (AG-IoT) networks based on network flow data. It was collected from a smart greenhouse environment equipped with IoT devices such as sensors (DHT11, DHT22, BME280, soil moisture, LDR) and microcontrollers (ESP32, Arduino Yun, ESP8266) communicating through standard protocols (HTTP, MQTT, TCP, UDP). The dataset includes both benign and malicious traffic. The final dataset contains approximately 1.28 million records (532 MB) distributed across nine traffic classes, with a significant imbalance between normal and attack samples, as shown in Table 4. This dataset is particularly suited for both binary (normal vs. attack) and multiclass intrusion detection tasks in smart agriculture environments [16].

Table 3: Features of SDN datasets [17]

Feature	Description
srcMac	Source MAC address
dstMac	Destination MAC address
srcIP	Source IP address
dstIP	Destination IP address
srcPort	Source port number
dstPort	Destination port number
last-seen	Record last time
Protocol	Textual representation of network protocol
proto-number	Numerical representation of network protocol
Dur	Average duration of aggregated records
Stddev	Standard deviation of aggregated records
Min	Minimum duration of aggregated records
Max	Maximum duration of aggregated records
Pkts	Total count of packets in transaction
Bytes	Total number of bytes in transaction
Spkts	Source-to-destination packet count
Dpkts	Destination-to-source packet count
Sbytes	Source-to-destination byte count
Dbytes	Destination-to-source byte count
srate	Source-to-destination packets per second
drate	Destination-to-source packets per second
Sum	Total duration of aggregated records
TnBPSrcIP	Total number of bytes per source IP
TnBPDstIP	Total number of bytes per destination IP
TnP-SrcIP	Total number of packets per source IP
TnP-DstIP	Total number of packets per destination IP
TnP-PerProto	Total number of packets per protocol
TnP-PerDport	Total number of packets per destination port
N-IN-Conn-P-SrcIP	Number of inbound connections per source IP
N-IN-Conn-P-DstIP	Number of inbound connections per destination IP
Attack	Attack or not
Category	Traffic category

Table 4: Type of Attacks in Farm-flow dataset [16]

Class Type	**Number of Instances**
Arp Spoofing (Arp_Spoofing)	2967
BotNet (BotNet_DDOS)	64
HTTP Flood (HTTP_Flood)	265025
ICMP Flood (ICMP_Flood)	76849
MQTT Flood (MQTT_Flood)	132506
Normal (Normal)	27,458
Port Scanning (Port_Scanning)	126700
TCP Flood (TCP_Flood)	381765
UDP Flood (UDP_Flood)	296553
Total	**1282429**

The "ASEADOS-SDN-IoT dataset" [18] was additionally considered for external cross-dataset validation. ASEADOS-SDN-IoT is a recent SDN-IoT intrusion detection dataset containing multiple network attack categories, including DoS, DDoS, Probe, and Bot traffic, collected from a distinct SDN-enabled IoT environment. The dataset contains 83 statistical traffic features extracted from SDN-IoT network flows. In this work, the dataset was used exclusively to evaluate the generalization capability and robustness of the proposed framework under domain-shift conditions.

4.2 Dataset Preprocessing

The SDN-IoT and Farm-flow datasets were carefully preprocessed prior to model training. First, the datasets were cleaned by removing missing values and duplicate entries before performing the train/validation/test split in order to avoid potential data leakage between subsets.

Then, categorical features were numerically encoded using the LabelEncoder. The numerical features were normalized using the MinMaxScaler for the SDN-IoT dataset, since its attributes exhibit homogeneous and bounded value ranges, which facilitates the convergence of neural network models. In contrast, for the Farm-flow dataset, the numerical variables show high variability and different scales; therefore, they were standardized using the StandardScaler, which is more suitable for network flow data characterized by dispersed and unbounded distributions.

Finally, the data were split into training (80%), validation (10%), and test (10%) sets using a stratified split approach to preserve the original class distribution.

For the Farm-flow dataset, a random oversampling with replacement (resampling) technique was applied only to the training set after the data splitting process in order to obtain the same number of samples across classes while avoiding data leakage and ensuring a fair model evaluation. In contrast, the SDN-IoT dataset was already balanced during its preparation and therefore did not require any additional balancing step. Figure 2 presents an overview of the preprocessing workflow.

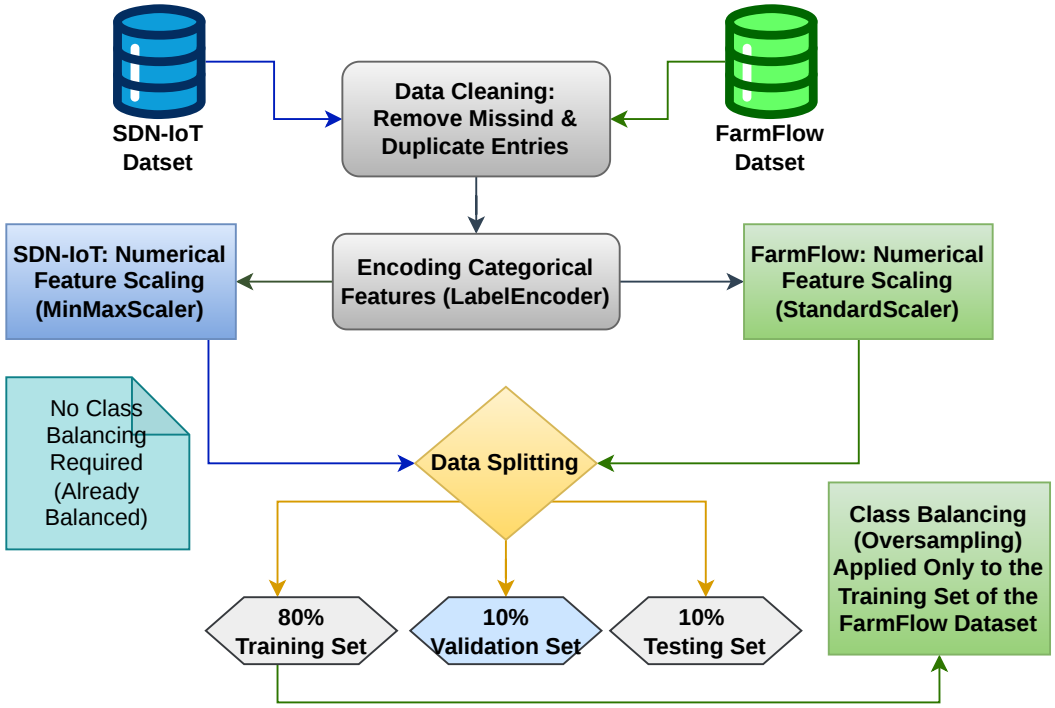


Figure 2: Preprocessing Workflow

4.3 Selection of the Classifier Model

SNNs are the third generation of neural networks, designed to emulate the brain's event-driven processing. Unlike conventional Artificial Neural Networks (ANNs) that rely on continuous activations, SNNs communicate through discrete electrical spikes, enabling efficient spatio-temporal data processing [19, 20]. Their sparse, event-driven computation significantly reduces power consumption, especially on neuromorphic hardware, making them suitable for energy constrained environments such as agricultural IoT systems, where sensor nodes often rely on limited battery power or renewable energy sources [21, 22]. Moreover, SNNs can naturally capture temporal dynamics in sensor data, such as soil moisture or temperature variations supporting real-time, adaptive decision-making in precision agriculture. Thanks to their robustness and ability to learn complex patterns without explicit mathematical models, SNNs provide an efficient and biologically inspired solution for intelligent, low-power agricultural monitoring [21, 22].

Random Forest (RF) and Convolutional Neural Networks (CNN) were selected as baseline models to ensure a comprehensive and fair comparison with the proposed SNN. Each represents a distinct paradigm in machine learning: RF constructs multiple decision trees from random subsets of data and features, aggregating their predictions to enhance robustness and generalization. It demonstrates strong resilience to overfitting, can effectively handle high-dimensional and heterogeneous data, and requires minimal parameter tuning. Moreover, RF performs consistently well on structured and tabular datasets, where its interpretability and low computational cost make it a standard benchmark for evaluating novel learning architectures [23, 24]. Its inclusion in this study provides a solid foundation for quantifying the performance improvements achieved by neural models in intrusion detection tasks.

CNNs, on the other hand, represent the cornerstone of modern DL, originally developed for image recognition but increasingly adapted to non-visual domains. CNNs employ convolutional and pooling layers to automatically extract local and global spatial correlations, followed by dense layers for classification. This hierarchical representation learning enables CNNs to capture complex dependencies between input features, an ability that extends to structured and temporal data such as network traffic flows [25]. Recent studies have successfully demonstrated CNNs' capacity to operate on tabular data, revealing their effectiveness in uncovering latent feature interactions and nonlinear dependencies that traditional models may overlook [26]. Therefore, comparing SNNs against CNNs and RF establishes a rigorous experimental baseline, allowing the evaluation of how event-driven spiking computation can simultaneously maintain high accuracy while drastically reducing inference time and energy

consumption in constrained and latency-sensitive environments such as agricultural IoT systems.

4.4 Local SNN based-IDS

To efficiently classify network traffic while maintaining low power consumption, an SNN model was implemented using the `snnTorch` framework [27]. The model exploits temporal spiking dynamics to encode information across discrete time steps, enabling event-driven computation and improved energy efficiency. Each local SNN module is trained on the associated farm private data. To simulate the privacy of those data, the training dataset obtained by the preprocessing step (Figure 2) is divided over the different farms.

4.4.1 SNN Model Architecture

The proposed local SNN based-IDS model consists of three fully connected (FC) layers interleaved with Leaky Integrate-and-Fire (LIF) neurons. Each layer includes normalization and regularization mechanisms to ensure stable convergence and prevent overfitting. The detailed configuration is summarized in Table 5.

Table 5: Layer configuration of the `SNN Model`.

Layer	**Type**	**Input / Output Size**	**Activation / Notes**
Input	Input Layer	Input size = number of features	–
Layer 1	Fully Connected (FC1)	192 neurons	LIF ($\beta = 0.95$)
	Batch Normalization	Normalization across features	–
Layer 2	Fully Connected (FC2)	96 neurons	LIF ($\beta = 0.95$)
	Batch Normalization	–	–
	Dropout	$p = 0.2$	Regularization
Output	Fully Connected (FC3)	Number of classes	LIF ($\beta = 0.95$)
Time Dynamics	Simulation	$n_steps = 20$	Spike accumulation over time

During each forward pass, the input data are encoded into continuous vectors and propagated across the network for 20 discrete time steps ($n_steps = 20$). At each step, LIF neurons integrate incoming currents, leak a portion of their membrane potential, and fire a spike when the potential exceeds a predefined threshold. The accumulated spike count over all steps determines the final class prediction.

Mathematically, the LIF neuron dynamics are expressed as:

$$U_t = \beta U_{t-1} + W X_t - S_{t-1} V_{\text{reset}}, \tag{1}$$

where U_t denotes the membrane potential at time t, β is the leak constant (set to 0.95), W represents the synaptic weights, X_t is the input current, and S_{t-1} is the spike output from the previous step. The term V_{reset} represents the reset potential applied after a spike. To enable gradient-based learning despite non-differentiable spike events, a surrogate gradient function is applied during backpropagation.

4.4.2 Training Strategy

The model was trained using the Adam optimizer with a learning rate of 5×10^{-4}, coupled with a Cosine Annealing Learning Rate Scheduler to promote smooth convergence. The Label Smoothing Cross-Entropy loss was adopted to enhance generalization and mitigate overconfidence in class predictions. Class weights were also incorporated to address data imbalance.

To prevent overfitting, early stopping was applied when validation loss stagnated for seven consecutive epochs. Training was conducted for a maximum of 50 epochs, with a batch size of 128. The training procedure is summarized in Algorithm 1.

4.5 Integration with Federated Learning

FL is a distributed machine learning paradigm that enables multiple nodes to collaboratively train a shared global model without exchanging raw data. Each participant performs local training and sends only model updates to a central aggregator, thereby preserving privacy and reducing communication overhead [28]. Within SDN for IoT, FL distributes intelligence across local controllers or edge devices, alleviating the computational and communication load of a centralized controller while improving scalability, resilience, and fault tolerance in large and heterogeneous deployments [29].

Algorithm 1: Training Procedure of the Spiking Neural Network (SNN)

Input: Training data $\mathcal{D} = \{(X_i, y_i)\}$, epochs $E = 50$, batch size $B = 128$, leak constant $\beta = 0.95$, time steps $n_steps = 20$
Output: Trained model parameters θ^*
Initialize network parameters θ, optimizer (Adam), scheduler (Cosine Annealing), and early stopping;
for $e = 1$ **to** E **do**
  foreach *batch* (X, y) *in* $\mathcal{D}$ **do**
    Reset membrane potentials $U_t = 0$;
    for $t = 1$ **to** n_steps **do**
      Compute input current $I_t = WX_t$;
      Update membrane potential $U_t = \beta U_{t-1} + I_t$;
      Generate spikes $S_t = \Theta(U_t - V_{th})$;
    Accumulate spike counts over time and compute predicted class $\hat{y}$;
    Compute loss $\mathcal{L} = \text{LabelSmoothingCrossEntropy}(y, \hat{y})$;
    Backpropagate gradients using surrogate gradient function;
    Update parameters $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}$;
  Adjust learning rate using Cosine Annealing scheduler;
  if *validation loss does not improve for 7 epochs* **then**
    Trigger early stopping;
    break;
return θ^*

Beyond privacy preservation, FL enhances SDN-IoT network management by supporting adaptive traffic monitoring, anomaly detection, and load balancing. By exchanging only model parameters instead of raw data, FL minimizes bandwidth consumption—an essential advantage in resource-constrained IoT environments [30]. In agricultural IoT systems, FL allows farms and stakeholders to collaboratively train intrusion detection models without sharing sensitive data, addressing issues of data ownership and trust. Its decentralized design is particularly effective in rural areas with intermittent connectivity, ensuring robust model updates and energy-efficient operation under constrained network conditions [31].

To enable decentralized and privacy-preserving training of the proposed SNN model, the Flower (FLWR) framework was employed to simulate an FL environment. The objective was to distribute the learning process across multiple local clients, representing agricultural SDN controllers, while maintaining model accuracy and reducing communication overhead. The implementation was developed using PyTorch and snnTorch for model definition, and Flower for orchestration of the federated training and aggregation process. The workflow can be summarized as follows:

- **Environment Configuration and Setup:** All necessary dependencies were installed, including flwr, torch, snntorch, and scikit-learn. Random seeds were fixed across all modules to ensure full reproducibility, and all hyperparameters were carefully tuned to ensure balanced convergence across clients.
- **Federated System Design:** The FL simulation was organized into a client–server architecture, consisting of:
  - Clients: Each client represented a local SDN controller. Clients trained local copies of the SNN model on their respective datasets for a fixed number of epochs and transmitted only the model parameters to the central server.
  - Server: The central FL server acted as the global SDN orchestrator, aggregating local updates received from all clients to build an improved global model using the Federated Averaging (FedAvg) strategy.

  This decentralized structure ensured privacy preservation, as raw data never left the local nodes, and enhanced scalability by distributing computational workloads across multiple participants.
- **Client Implementation:** Each federated client was implemented as a customized NumPyClient class, which handled the local training and evaluation processes. Locally, each client:
  - Initialized its SNN model.
  - Trained the model using the Adam optimizer with a cosine annealing learning rate schedule for stable convergence.
  - Applied early stopping when the validation loss plateaued to reduce unnecessary computation.
  - Returned updated model parameters and local metrics to the server at the end of each round.

- **Server-Side Aggregation and Evaluation:** The server employed the FedAvg aggregation method to combine the parameters received from all clients proportionally to their local dataset sizes. After each global communication round, the aggregated model was evaluated on a global validation set to monitor its convergence and stability. The evaluation function measured the global accuracy and loss at each round, updating the best-performing global model parameters. An early stopping mechanism was integrated at the server level to terminate training automatically if no performance improvement was observed over multiple rounds, thereby minimizing communication costs and overfitting.

Figure 3 illustrates the workflow of the proposed FL-SNN system.

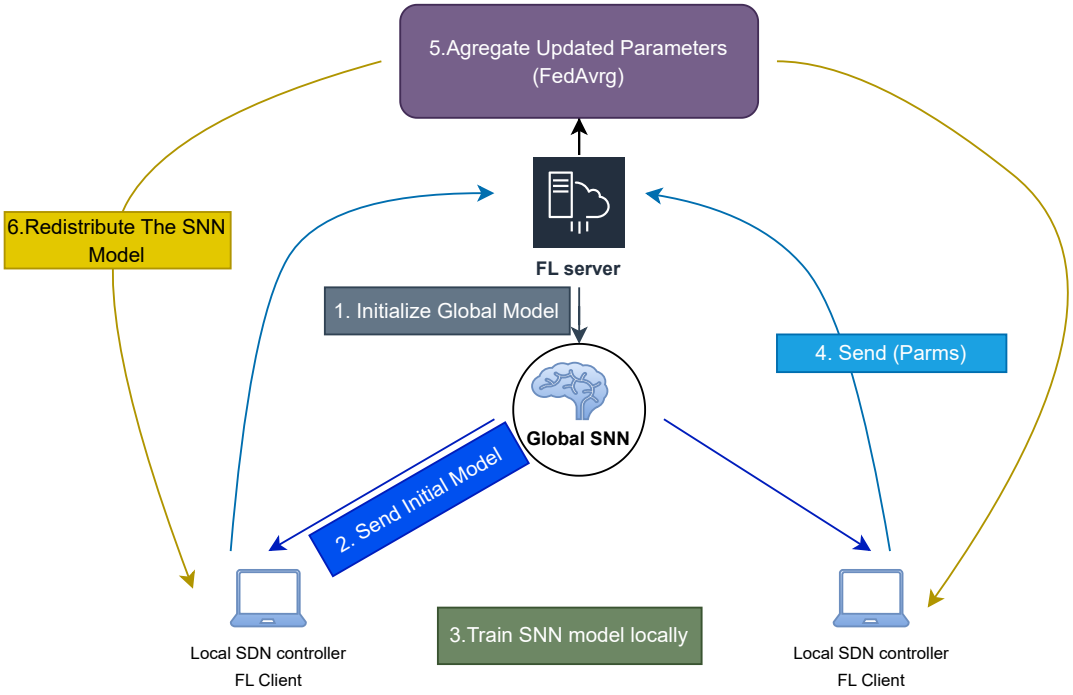


Figure 3: Proposed FL-SNN training Workflow

5 Experimental Setup

This section presents the evaluation metrics employed to measure model performance, along with the hyperparameter settings used during training. All experiments were conducted in Python using the Kaggle platform[1].

5.1 Evaluation Metrics

The performance of our system and its benchmarking against other approaches are evaluated using a set of metrics derived from the confusion matrix framework, as detailed in [32]. These metrics are essential for assessing our system's ability to accurately detect network anomalies. The chosen metrics include Accuracy, Precision, Recall, F1-score, False Positive Rate (FPR), and Testing Time, each offering distinct insights into the system's effectiveness.

The confusion matrix forms the basis for these metrics, defined by four core components:

- **True Positives (TP)**: Network threats correctly pinpointed by the system.
- **True Negatives (TN)**: Benign instances accurately classified by the system.
- **False Positives (FP)**: Benign instances incorrectly identified as threats.
- **False Negatives (FN)**: Threats undetected by the system.

[1] https://www.kaggle.com

The metrics are defined as follows:

- **Accuracy**: Measures the proportion of all instances correctly classified, including threats and benign cases, relative to the total dataset, providing a comprehensive evaluation of our system's reliability.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{2}$$

- **Precision**: Assesses the fraction of correctly flagged threats among all instances classified as threats, indicating the system's precision in threat detection.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3}$$

- **Recall**: Quantifies the share of actual threats accurately detected, reflecting the system's ability to identify true positives.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4}$$

- **F1-Score**: Combines Precision and Recall through their harmonic mean, offering a balanced measure of our system's overall performance.

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5}$$

- **Testing Time**: Denotes the time, in seconds, required by our system to process and classify the test dataset, assessing its computational efficiency.

5.2 Energy Consumption Estimation

The estimation of the total energy consumption of an SNN relies on two main components: the energy associated with spike events and the energy consumed by Multiply Accumulate (MAC) operations. The reported energy values correspond to hardware-independent algorithmic complexity estimations based on synaptic operations and spike activity, following recent neuromorphic benchmarking principles [33].This analytical estimation provides a hardware-independent approximation of the computational energy required during inference.

- **The total energy:** $E_{\text{total,SNN}}$ (in joules) consumed by the SNN can be expressed as the sum of spike-based and MAC-based energy components:

$$E_{\text{total,SNN}} = E_{\text{spike}} + E_{\text{MAC}} \tag{6}$$

- **The spike energy:** represents the cost of generating and transmitting neuronal spikes, while the MAC energy corresponds to the synaptic computations across network layers. The total spike energy is calculated as:

$$E_{\text{spike}} = N_{\text{spikes}} \times E_{\text{spike_per_event}} \tag{7}$$

  where:

  - N_{spikes} is the total number of spikes generated across all neurons and time steps.
  - $E_{\text{spike_per_event}}$ is the average energy consumed per spike event.

  Following prior neuromorphic studies [34–38], the energy per spike is typically approximated to 1 pJ (1×10^{-12} J), which represents a realistic benchmark for low-power, event-driven computation.

- **MAC Operation Energy:** For SNNs, the number of MAC operations depends on both the network connectivity and the spiking sparsity. In the implemented model, dense MAC computations are estimated for the first layer as:

$$N_{\text{MAC,dense}} = \text{batch_size} \times \text{in_features} \times \text{out_features} \tag{8}$$

  while subsequent layers use sparse MAC estimates based on active spikes:

$$N_{\text{MAC,sparse}} = \sum_t (\text{spikes}_t \times \text{out_features}) \tag{9}$$

This distinction accurately reflects the event-driven computation of SNNs, where operations occur only upon spike arrivals, unlike dense computations CNN and RF. The total MAC-based energy in SNNs is thus obtained as:

$$E_{\text{MAC,SNN}} = N_{\text{MAC,total}} \times E_{\text{MAC_per_operation}} \tag{10}$$

where:

- $N_{\text{MAC, SNN}}$ denotes the total number of MAC operations performed by the network.
- $E_{\text{MAC_per_operation}}$ is the energy per MAC.

According to [39], the energy required for a single MAC operation in 45 nm CMOS technology is approximately 4.6 pJ (4.6×10^{-12} J). This benchmark value is widely adopted in the energy modeling of neural networks and serves as a standard reference for digital computations.

For CNNs, all layers perform dense computations, leading to a higher number of MAC operations. In the implemented model, the total MACs are estimated as the sum of all convolutional and fully connected layers [40, 41]:

$$N_{\text{MAC,CNN}} = N_{\text{Conv1}} + N_{\text{Conv2}} + N_{\text{FC3}} \tag{11}$$

where:

$$N_{\text{Conv1}} = \text{batch_size} \times \text{input_size} \times \text{out_channels}_1 \times k_1, \tag{12}$$

$$N_{\text{Conv2}} = \text{batch_size} \times \text{out_channels}_1 \times \text{out_channels}_2 \times k_2, \tag{13}$$

$$N_{\text{FC3}} = \text{batch_size} \times (\text{out_channels}_2 \times \text{input_size}) \times \text{out_features}_3, \tag{14}$$

with k_1 and k_2 denoting the kernel sizes of the first and second convolutional layers, respectively. The total energy is therefore computed as:

$$E_{\text{total,CNN}} = N_{\text{MAC,CNN}} \times E_{\text{MAC_per_operation}} \tag{15}$$

For the RF model, which is CPU-bound and non-neuromorphic, the energy consumption is estimated from the inference time and average CPU power during execution [42]:

$$E_{\text{total,RF}} = P_{\text{CPU}} \times T_{\text{inference}} \tag{16}$$

where:

- P_{CPU} is the average CPU power consumption during inference. In this study, we assume $P_{\text{CPU}} = 80$ W, which corresponds to a typical desktop CPU under computational load [42].
- $T_{\text{inference}}$ is the total inference time (in seconds).

- **Energy per Inference:** To estimate the average energy consumed per sample during inference, the total energy is divided by the number of test samples:

$$E_{\text{inference}} = \frac{E_{\text{total}}}{N_{\text{samples}}} \tag{17}$$

where E_{total} is the total energy consumed by the model (SNN, CNN, or RF), and N_{samples} is the total number of test samples.

Although this estimation omits memory access and communication costs, it captures the dominant computational components of SNN inference. By leveraging the sparse spiking activity inherent to SNNs, this method demonstrates how energy efficiency arises naturally from event-driven computation. It also provides a quantitative basis for comparing energy-aware models under FL or IoT deployment constraints, where minimizing inference energy is a key requirement.

5.3 Parameters Setting

Table 6 presents the parameter settings used for the SNN and the FL framework.

Table 6: Summary of SNN Model and Federated Learning Parameters.

Category	Parameter	Value / Description
Architecture	Hidden Layer 1	192 neurons
	Hidden Layer 2	96 neurons
	Dropout rate	0.2
	Leak constant (β)	0.95
	Time steps (n_{steps})	20
Training	Optimizer	Adam
	Learning rate	5×10^{-4}
	Weight decay	1×10^{-5}
	Scheduler	Cosine Annealing LR
	Batch size	128
	Epochs	50
	Early stopping patience	7
	Loss function	Label Smoothing Cross-Entropy (smoothing = 0.1)
Federated Learning Configuration	Number of clients ($N_{clients}$)	10 simulated clients
	Epochs per client (E_{local})	5 local epochs per round
	Global rounds (R_{global})	10 aggregation rounds
	Aggregation strategy	Federated Averaging (FedAvg)
	Early stopping patience (server)	5
	Client–server communication mode	Synchronous updates

6 Results and Discussion

This section reports the experimental results obtained from the proposed SNN integrated with FL. To demonstrate the robustness and effectiveness of the SNN, its performance is compared against two benchmark models, CNN and RF. The evaluation is conducted using two real-world datasets, SDN-IoT and Farm-flow. Furthermore, the results of the FL framework are analyzed, and the outcomes of our proposed system are discussed in relation to existing research works.

6.1 Results of the SNN Model

SNN Model Performance: Figure 4 presents the confusion matrices of the proposed SNN evaluated on both the SDN-IoT and Farm-Flow datasets. These matrices provide a detailed representation of the classification performance across benign traffic and multiple attack categories. The strong dominance of the diagonal elements indicates the model's high capability to accurately discriminate between different traffic patterns, while the limited number of off-diagonal misclassifications demonstrates the robustness and reliability of the proposed approach across diverse attack scenarios and traffic distributions. Furthermore, the confusion matrices enable a fine-grained analysis of class-level prediction behaviour and highlight the SNN's effectiveness in handling both balanced and imbalanced traffic distributions.



(a) SDN-IoT Dataset

(b) Farm-flow Dataset

Figure 4: Confusion Matrices of the SNN on the SDN-IoT and Farm-flow Datasets

Figure 5 illustrates the training and validation curves of the SNN on the SDN-IoT dataset, while Figure 6 presents the corresponding curves on the FarmFlow dataset. These learning curves provide a comprehensive visualization of the model's optimization process throughout training, highlighting its convergence stability, learning efficiency, and generalization capability. Furthermore, they enable the assessment of the relationship between training and validation performance, thereby facilitating the identification of potential overfitting or underfitting behaviours during the learning phase.

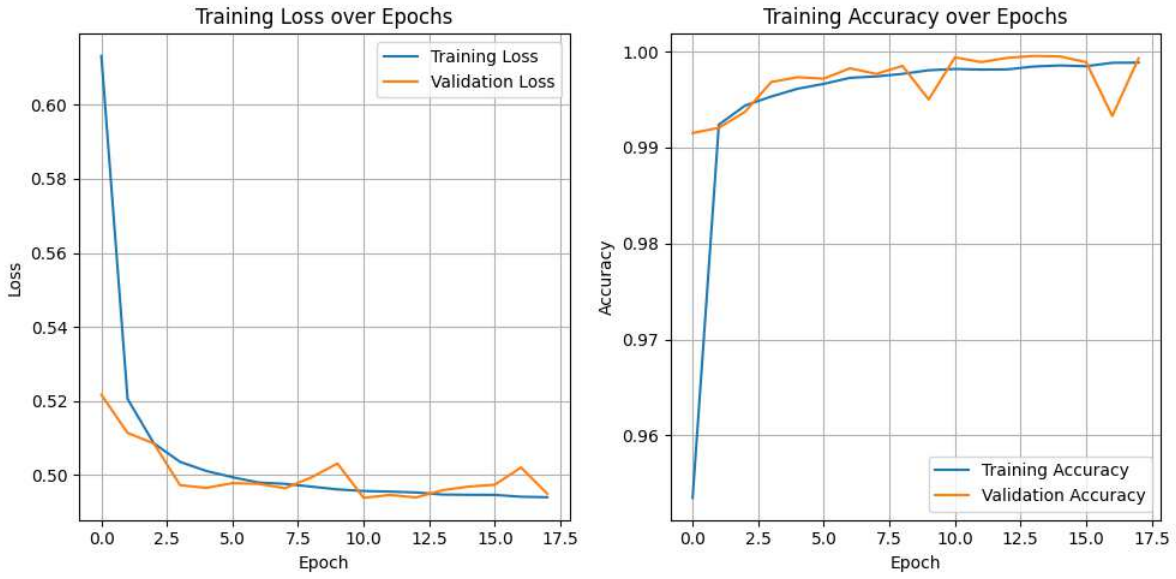


Figure 5: Learning curves of the SNN on the SDN-IoT dataset

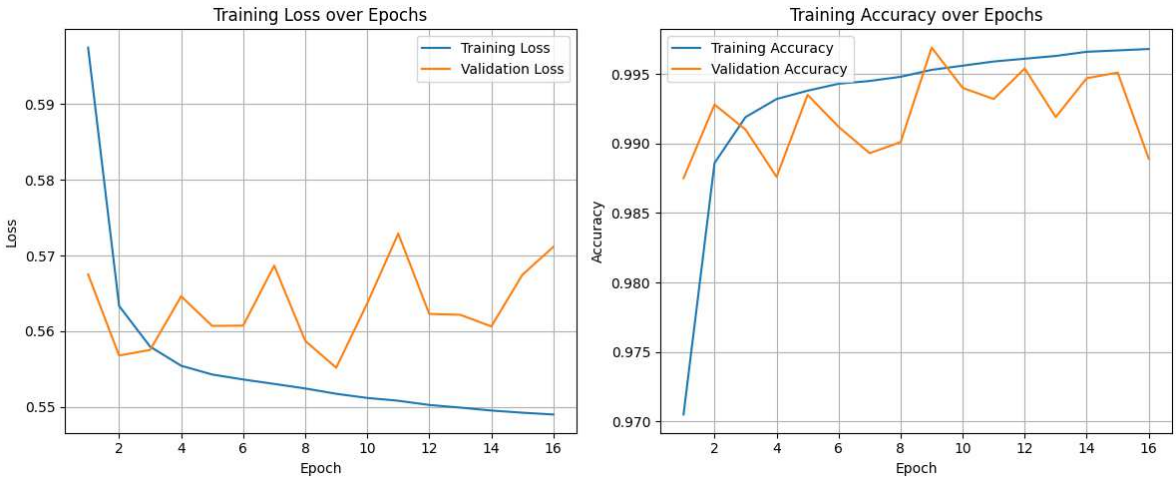


Figure 6: Learning curves of the SNN on the Farm-flow dataset

Performance Comparison of SNN, CNN, and RF Models: Figures 7 and 8 present a comparative evaluation of the proposed SNN against two widely used baseline models, CNN and RF, on the SDN-IoT and Farm-Flow datasets, respectively.

On the SDN-IoT dataset, the proposed SNN achieved an accuracy of 99.93%, closely matching the RF classifier and slightly outperforming the CNN model. Similarly, on the Farm-Flow dataset, the SNN maintained a high classification accuracy of 99.68%, significantly surpassing the CNN model, while remaining highly competitive with RF. The strong performance obtained across both datasets demonstrates the capability of the proposed SNN to effectively capture complex traffic behaviours and discriminate between benign and malicious network activities under different traffic distributions.

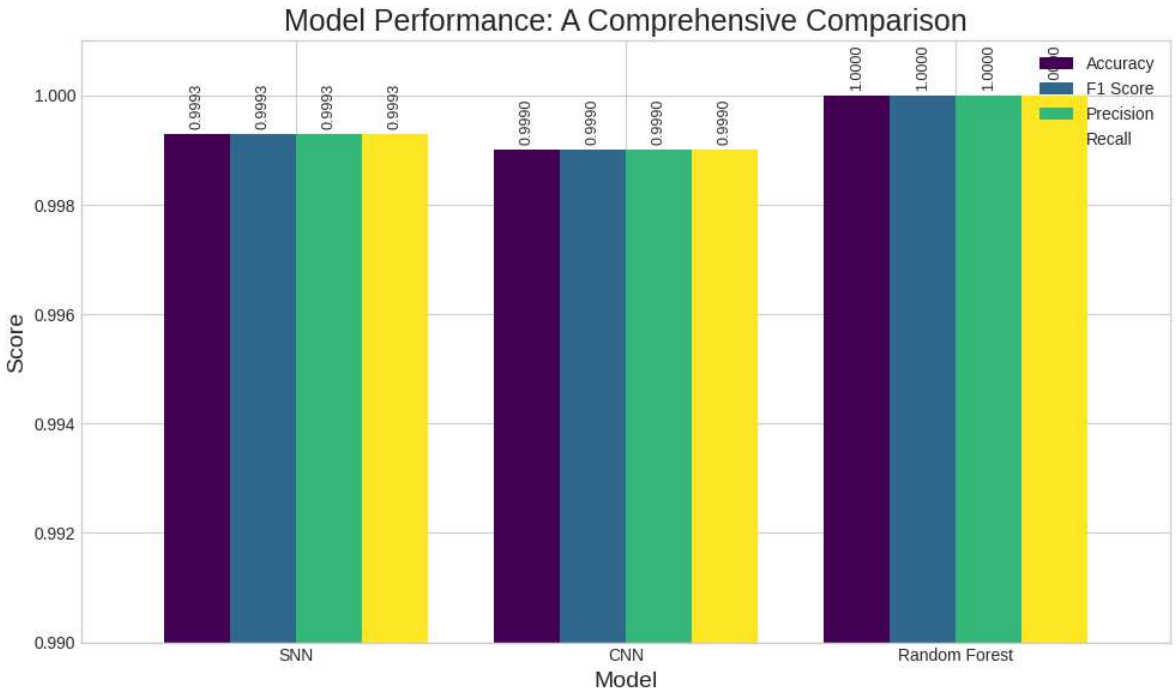


Figure 7: Performance comparison of SNN, CNN, and RF models on the SDN-IoT dataset

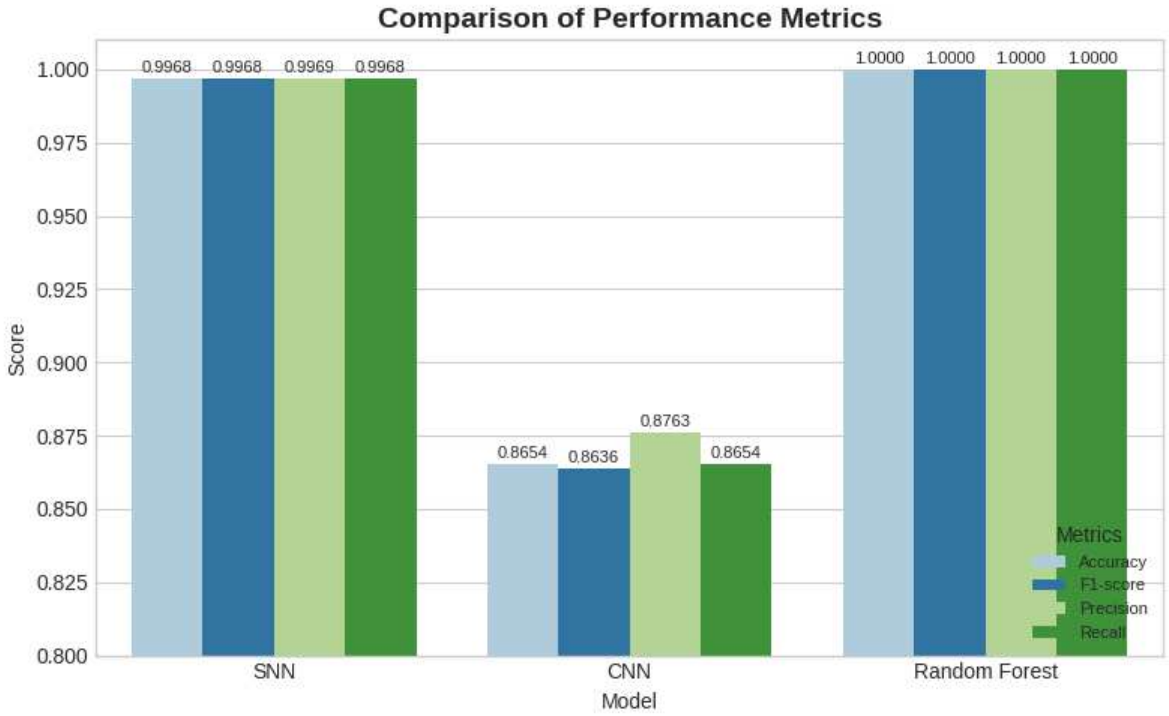
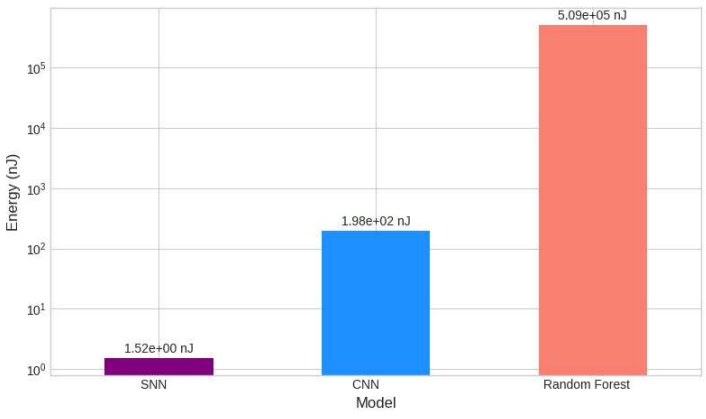
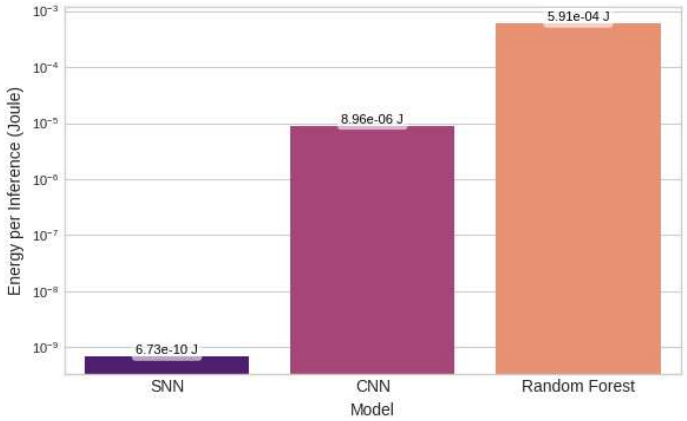


Figure 8: Performance comparison of SNN, CNN, and RF models on the Farm-flow dataset

Energy Consumption Comparison Across SNN, CNN, and RF Models: Figure 9 compares the inference energy consumption of the SNN, CNN, and RF models on both datasets according to the hardware-independent formulation of equations 6, 15, 16 . The proposed SNN consistently exhibited the lowest energy consumption among all evaluated approaches. On the SDN-IoT dataset, the SNN required only 1.5174×10^{-9} J per inference, whereas the CNN and RF models consumed substantially higher energy due to their dense computational operations and continuous processing mechanisms. A similar trend was observed on the Farm-Flow dataset, where the SNN maintained ultra-low energy consumption while preserving high detection performance. These results highlight the suitability of SNNs for energy-constrained SDN-IoT environments, particularly in smart agricultural infrastructures where resource efficiency is a critical requirement.



(a) SDN-IoT Dataset

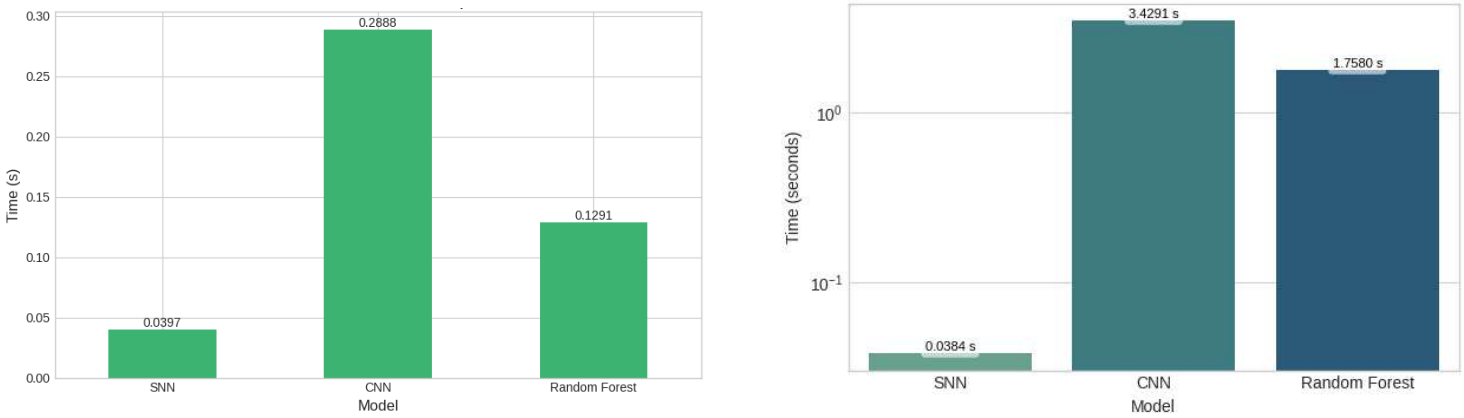


(b) FarmFlow Dataset

Figure 9: Energy Consumption Comparison Across SNN, CNN, and RF Models on Both Datasets

Inference Time Comparison of SNN, CNN, and RF Models: Figure 10 illustrates the inference time comparison between the evaluated models on both datasets. The proposed SNN achieved the lowest inference latency, requiring only 0.0397 s on the SDN-IoT dataset and 0.0384 s on the Farm-Flow dataset. In contrast, the CNN and RF models exhibited noticeably higher inference times, particularly on the Farm-Flow dataset. The reduced latency of the SNN can be attributed to its event-

driven computational paradigm, which activates neurons only when significant spike events occur, thereby reducing unnecessary computations. This fast inference capability makes the proposed framework highly suitable for real-time intrusion detection scenarios requiring rapid response and low-latency decision making.



(a) SDN-IoT Dataset

(b) FarmFlow Dataset

Figure 10: Inference Time Comparison of SNN, CNN, and RF Models on Both Datasets

Table 7 summarizes the overall performance of the evaluated models in terms of classification effectiveness, inference latency, and energy efficiency. The presented results highlight the trade-offs between detection capability and computational cost, demonstrating the ability of the proposed SNN framework to simultaneously achieve high classification performance and lightweight resource utilization.

Table 7: Summary of Model Performance on SDN-IoT and Farm-flow Datasets

Dataset	Model	Accuracy	F1-Score	Precision	Recall	Inference Time (s)	Energy per Inference (J)
SDN-IoT	SNN	0.9993	0.9993	0.9993	0.9993	0.0397	1.5174e-09
	CNN	0.9990	0.9990	0.9990	0.9990	0.2888	1.9784e-07
	RF	1.0000	1.0000	1.0000	1.0000	0.1291	5.0922e-04
FarmFlow	SNN	0.9968	0.9968	0.9969	0.9968	0.0384	6.7280e-10
	CNN	0.8654	0.8636	0.8763	0.8654	3.4291	8.9645e-06
	RF	1.0000	1.0000	1.0000	1.0000	1.7580	5.9093e-04

The experimental evaluation conducted on the SDN-IoT and Farm-Flow datasets demonstrates the effectiveness, robustness, and operational efficiency of the proposed FL-SNNIDS framework compared to CNN and RF baselines.

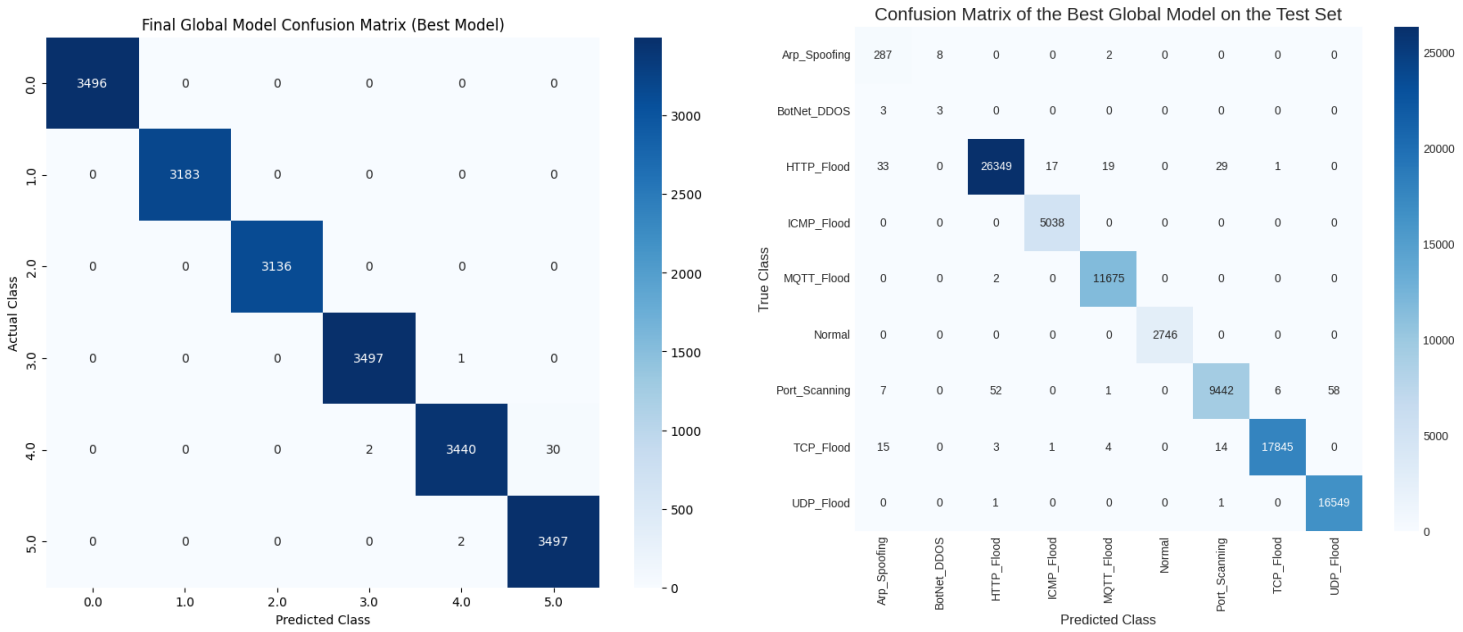
Performance Analysis on the SDN-IoT Dataset: On the SDN-IoT dataset, the proposed SNN achieved an accuracy and F1-score of 99.93%, with the confusion matrix showing only a very limited number of misclassifications. This performance indicates the effectiveness of the SNN in capturing temporal traffic dynamics through its event-driven processing mechanism. In addition, the learning curves demonstrate stable convergence with a small gap between training and validation performance, indicating strong generalization capability and the absence of significant overfitting. Although the RF model achieved slightly higher classification accuracy, it incurred substantially higher computational and energy costs. Similarly, the CNN model required higher inference latency and energy consumption due to its dense floating-point computations. The proposed SNN achieved ultra-low energy consumption of only 1.52×10^{-9} J per inference and a reduced inference time of 0.0397 s, demonstrating its suitability for real-time and resource-constrained SDN-IoT environments.

Robustness on the Farm-Flow Dataset: The Farm-Flow dataset represents a more challenging smart agriculture environment characterized by imbalanced traffic distributions. To ensure realistic evaluation conditions, balancing techniques were applied only to the training set, while the validation and test sets preserved their original distributions. Under these conditions, the proposed SNN maintained strong classification performance with an accuracy of 99.68%, demonstrating high robustness and generalization capability. In contrast, the CNN model experienced a noticeable degradation in performance, suggesting reduced adaptability to imbalanced agricultural IoT traffic. The learning curves further confirm the stability of the proposed framework, with no significant signs of overfitting observed during training.

Energy Efficiency and Practical Implications: A major contribution of this work lies in the achieved balance between detection performance and computational efficiency. Unlike CNN and RF models, the proposed SNN relies on sparse neuronal activations, where neurons fire only when significant spike events occur. This event-driven behaviour substantially reduces unnecessary computations, resulting in lower energy dissipation and faster inference. Overall, the proposed FL-SNNIDS framework provides a scalable, lightweight, and energy-aware intrusion detection solution suitable for distributed and privacy-preserving smart agricultural SDN-IoT infrastructures.

6.2 Results of the federated learning framework

Figure 11 presents the confusion matrices obtained from the proposed FL-SNNIDS framework on the SDN-IoT and Farm-Flow datasets. The results demonstrate that the federated model preserved strong classification capability across distributed clients, with the majority of samples correctly classified and only a limited number of misclassifications observed in minority attack classes.



(a) SDN-IoT Dataset

(b) Farm-flow Dataset

Figure 11: Confusion Matrices of the FL framework on the SDN-IoT and Farm-flow datasets

Figures 12 and 13 illustrate the evolution of the global accuracy and loss during the federated training process. A rapid increase in global accuracy accompanied by a sharp reduction in loss is observed during the initial communication rounds, followed by stable convergence behaviour. This indicates efficient aggregation of local model updates and stable collaborative learning across distributed clients.

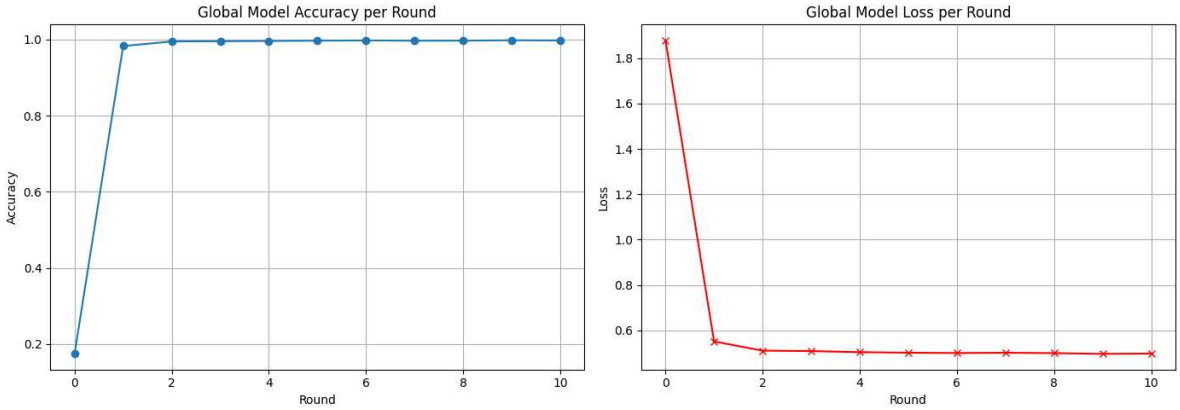


Figure 12: Global accuracy and loss per round of the SNN on the SDN-IoT dataset

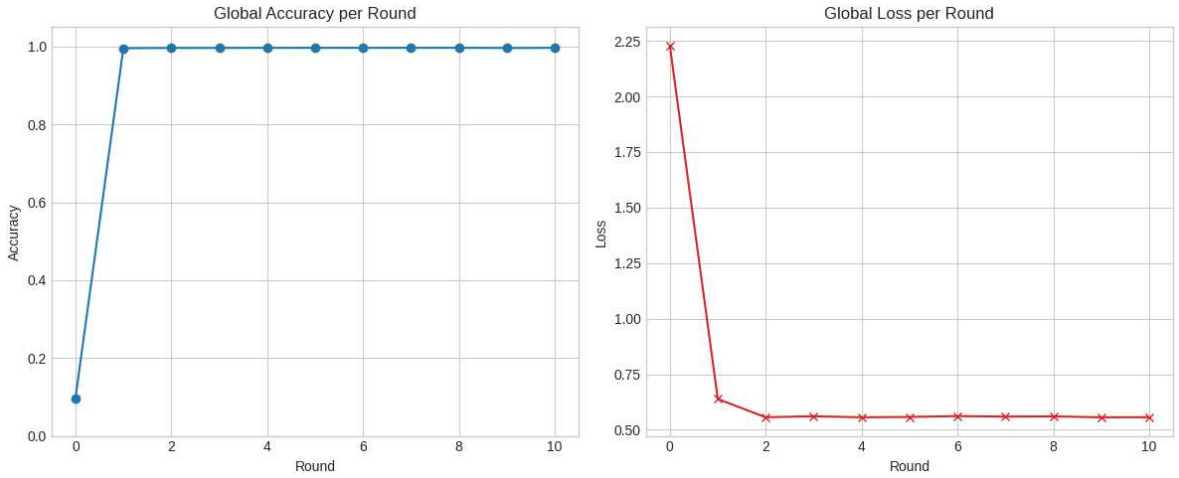


Figure 13: Global accuracy and loss per round of the SNN on the Farm-flow dataset

Table 8 summarizes the quantitative performance of the proposed federated framework. On the SDN-IoT dataset, the FL-SNNIDS achieved a final accuracy of 99.83%, while maintaining stable convergence after a limited number of communication rounds. Similarly, on the Farm-Flow dataset, the framework preserved strong classification capability despite the imbalanced traffic distribution and heterogeneous attack behaviours.

Table 8: Summary of Federated Learning Results Using SNN

Dataset	**Rounds**	**Final Accuracy**	**Final Loss**	**Precision**	**Recall**	**F1-score**
SDN-IoT	10	0.9983	0.4984	0.9983	0.9983	0.9983
FarmFlow	10	0.9968	0.5566	0.8987	0.9386	0.9144

The federated training process exhibited rapid and stable convergence across both evaluated datasets. On the SDN-IoT dataset, the global model converged within only 10 communication rounds, achieving a final accuracy of 99.83% with a loss of 0.4984. This fast convergence can be attributed to the discriminative nature of network traffic patterns, the effectiveness of the adopted preprocessing pipeline, and the ability of SNNs to efficiently capture temporal variations through sparse spike-based representations.

Similarly, the proposed framework maintained strong classification performance on the Farm-Flow dataset, achieving 99.68% accuracy despite the imbalanced traffic distribution. The integration of regularization mechanisms, including dropout, label smoothing, and early stopping, contributed to stabilizing the federated optimization process and improving model generalization across participating clients. Although certain minority classes exhibited slightly lower class-wise F1-scores due to the limited number of training samples and the imbalanced distribution of attack categories in the Farm-Flow dataset, the overall detection

capability remained consistently high. This indicates that the proposed framework preserved strong generalization performance while maintaining stable classification behaviour across both majority and minority traffic classes.

Furthermore, the proposed framework demonstrated high computational efficiency. The sparse and event-driven processing mechanism of SNNs reduced unnecessary neuronal activations, resulting in significantly lower energy consumption and inference latency compared to CNN and RF baselines. These characteristics make the proposed FL-SNNIDS framework particularly suitable for real-time and resource-constrained SDN-IoT smart agriculture environments.

From a system-level perspective, the lightweight computational requirements of the proposed FL-SNNIDS framework suggest its suitability for future deployment on low-power edge devices commonly used in SDN-IoT smart agriculture infrastructures. Platforms such as Raspberry Pi and NVIDIA Jetson devices can support real-time SNN workloads with reduced computational demand [43].

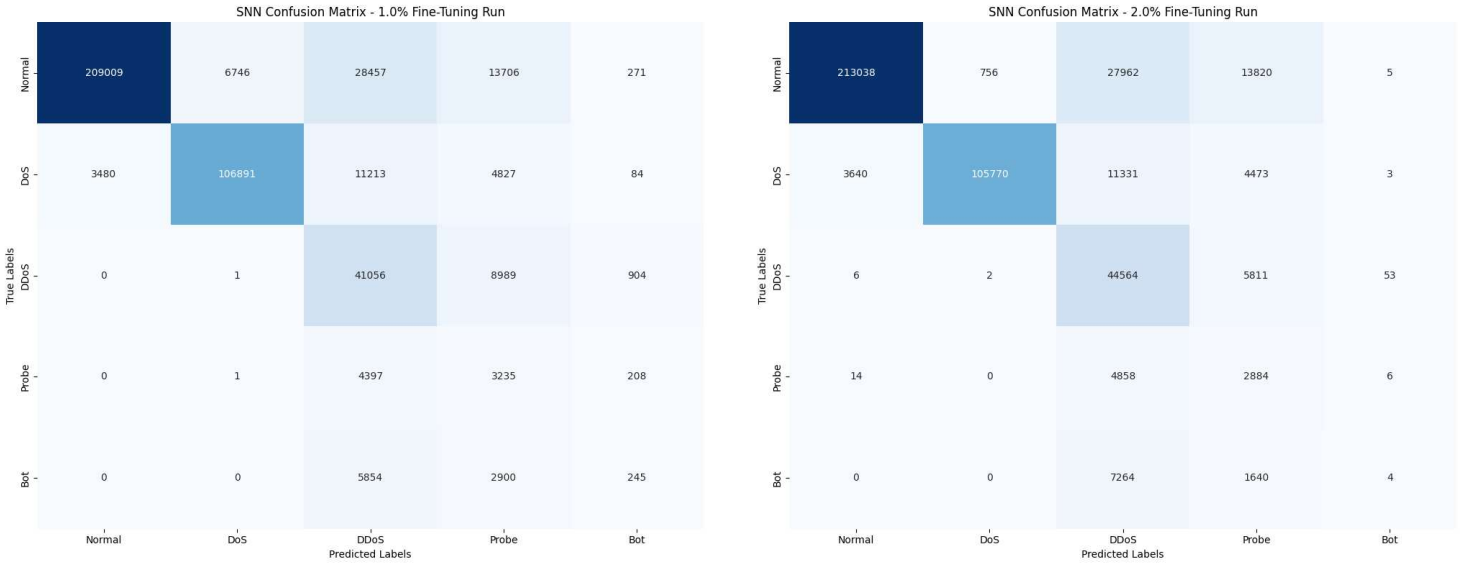
The reported energy values were analytically estimated from synaptic operations and spike activity, and therefore represent hardware-independent algorithmic-level approximations rather than direct hardware measurements [33]. Although these estimations do not account for memory access, communication overhead, or hardware architectural constraints, the observed energy efficiency trend remains consistent with recent neuromorphic computing studies. In particular, real hardware benchmarks on Intel Loihi demonstrated substantially lower energy consumption compared to conventional ANN/CNN-based solutions running on GPU edge platforms such as NVIDIA Jetson [44, 45]. However, achieving such energy gains in practice generally requires deployment on dedicated neuromorphic hardware optimized for sparse spike-based computation.

6.3 External Cross-Dataset Validation

To further evaluate the robustness and generalization capability of the proposed FL-SNNIDS framework, an external cross-dataset validation was conducted using the ASEADOS-SDN-IoT dataset. Unlike conventional evaluations performed on a single dataset, this experimental protocol assesses the ability of the proposed framework to adapt to a completely unseen SDN-IoT environment characterized by different traffic distributions and attack behaviours.

The SNN model was initially pre-trained on the SDN-IoT dataset using its native attack categories to learn discriminative traffic representations and complex decision boundaries. Subsequently, the model was evaluated on the ASEADOS-SDN-IoT dataset under a domain-shift scenario. To ensure structural compatibility between both environments, a strict feature-alignment process was performed by restricting the analysis to 15 common traffic features shared between the two datasets, namely: *srcPort, dstPort, proto_number, Dur, Pkts, Bytes, Spkts, Dpkts, Sbytes, Dbytes, Srate, Drate, Min, Max,* and *Mean*. The final classification layer was then adapted to the five native ASEADOS traffic classes: *Normal, DoS, DDoS, Probe,* and *Bot*.

To evaluate the adaptation capability of the proposed framework under limited-data constraints, a Few-Shot Fine-Tuning (FT) strategy was adopted using only 1.0% and 2.0% of the target-domain training samples. Few-shot fine-tuning refers to adapting a pre-trained model to a new target environment using only a very limited number of labeled samples. This evaluation protocol is particularly important for real-world SDN-IoT environments, where collecting and labeling large-scale attack data from newly deployed agricultural infrastructures is often costly, time-consuming, and impractical. Consequently, this experiment aims to assess whether the proposed FL-SNNIDS framework can rapidly adapt to previously unseen network conditions and attack distributions while maintaining acceptable detection performance under strict data-availability constraints. Figures 14 illustrates the confusion matrices obtained under both adaptation settings, while Table 9 summarizes the detailed classification performance achieved during the external cross-dataset validation process. The reported results demonstrate the ability of the proposed FL-SNNIDS framework to maintain stable detection performance under severe domain-shift and limited-data adaptation conditions.



(a) 1.0% Few-Shot Fine-Tuning on ASEADOS-SDN-IoT

(b) 2.0% Few-Shot Fine-Tuning on ASEADOS-SDN-IoT

Figure 14: Confusion Matrices of the external cross-dataset validation on the ASEADOS-SDN-IoT dataset

Table 9: Performance of the proposed FL-SNNIDS framework during external cross-dataset validation on the ASEADOS-SDN-IoT dataset under few-shot fine-tuning settings.

Class	**1.0% Few-Shot Fine-Tuning**				**2.0% Few-Shot Fine-Tuning**			
	Precision (%)	**Recall (%)**	**F1-Score (%)**	**Support**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**	**Support**
Normal	98.36	80.95	88.81	258189	98.31	83.35	90.22	255581
DoS	94.06	84.50	89.03	126495	99.29	84.47	91.28	125217
DDoS	45.13	80.58	57.86	50950	46.43	88.36	60.87	50436
Probe	9.61	41.26	15.59	7841	10.07	37.16	15.85	7762
Bot	14.31	2.72	4.57	8999	5.63	0.04	0.09	8908
Overall Accuracy (%)	79.66				81.77			
Precision (%)	87.96				89.37			
Recall (%)	79.66				81.77			
F1-Score (%)	82.44				84.13			

The external cross-dataset validation results demonstrate the strong adaptation capability of the proposed FL-SNNIDS framework under severe domain-shift conditions. Unlike the SDN-IoT dataset used during pre-training, the ASEADOS-SDN-IoT dataset introduces previously unseen traffic distributions and attack behaviours, creating a highly challenging evaluation scenario for the proposed framework.

Despite these challenging conditions, the proposed framework achieved stable performance using only a very limited amount of target-domain data. With 1.0% few-shot fine-tuning, the framework reached an overall accuracy of 79.66% and a weighted F1-score of 82.44%. Increasing the adaptation ratio to only 2.0% further improved the accuracy to 81.77% and the weighted F1-score to 84.13%, confirming the rapid adaptation capability of the SNN architecture under constrained learning conditions.

The obtained results also show strong detection consistency for the dominant traffic categories. At 2.0% fine-tuning, the framework achieved a weighted precision of 89.37%, while maintaining high F1-scores for *Normal* and *DoS* traffic, reaching 90.22% and 91.28%, respectively. In addition, the confusion matrix analysis shows that the proposed model effectively minimizes attack-to-normal misclassifications, which represents a critical requirement for practical IDS deployment in real-world SDN-IoT environments, as shown in Table 10.

Although the *Probe* and particularly the *Bot* classes exhibited lower detection performance, this limitation can be attributed to their absence from the source-domain training data, their limited representation in the target dataset, and the small amount of labeled data available during few-shot adaptation. These results indicate that the proposed framework transfers effectively to dominant traffic categories, while remaining less reliable for novel and underrepresented attacks.

Overall, the cross-dataset evaluation demonstrates promising but class-dependent transferability across heterogeneous SDN-IoT environments. Since labeled target-domain samples were used during fine-tuning, these results should not be interpreted as

Table 10: Attack-to-Normal Misclassification Analysis

Attack Type	Predicted as Normal	Total Samples	Percentage (%)
DoS	3640	125217	2.91
DDoS	6	50436	0.01
Probe	14	7762	0.18
Bot	0	8908	0.00

a complete zero-day attack detection evaluation.

6.4 Comparison with Related Work

To position the proposed FL-SNNIDS framework within the existing intrusion detection literature, Table 11 presents a contextual comparison with representative ML, DL, SNN, and FL-based approaches. The comparison considers classification performance, inference latency, and energy-related metrics across SDN-IoT, smart agriculture, and distributed IoT security settings. However, the reported results should be interpreted with caution, as the referenced studies rely on different datasets, preprocessing procedures, data partitions, hardware platforms, and experimental protocols. Therefore, the table is intended to provide contextual positioning rather than a direct and strictly controlled performance comparison.

Table 11: Comparative performance analysis of intrusion detection models in SDN-IoT and Smart Agriculture.

Reference	Model	Dataset	Accuracy	F1-Score	Precision	Recall	Inference Time (s)	Energy (J)
[13]	SNN	CIFAR-10	91.0%	–	–	–	–	3.5×10^{-7}
[14]	Optimized SNN (Spike-Thrift)	CIFAR-10	92.97%	–	–	–	0.0005	4.5×10^{-7}
[7]	LSTM	SDN-IoT	97.1%				–	–
[10]	RNN + Ensemble	SDN-IoT	97.0%	96.0%	97.0%	97.0%	–	–
[9]	CNN + LSTM (HCL)	SDN-IoT	92.0%	92.0%	90.0%	91.0%	1.38	–
[11]	GRU + Feature Fusion + Ensemble	SDN-IoT	98.0%	98.0%	98.0%	98.0%	–	–
[8]	BiLSTM	SDN-IoT	92.9%	92.64%	92.82%	92.86%	–	–
[12]	Deep Learning (Threat-Hunting Framework)	SDN-IoT	98.85%	99.31%	98.93%	99.68%	2.88	–
[16]	ML (RF)	Farm-flow	76.37%	72.30%	81.70%	76.37%	–	–
[6]	CNN (FL)	CICIoT2023	95.27%	94.92%	96.07%	95.27%	0.0014	–
[15]	FL + Blockchain	Industrial IoT	97.3%	–	–	–	–	–
Our Work	SNN	SDN-IoT Farm-flow	99.93% 99.68%	99.93% 99.68%	99.93% 99.69%	99.93% 99.68%	0.0397 0.0384	1.52×10^{-9} 6.73×10^{-10}

Discussion of Results: The table compares the proposed FL-SNNIDS framework with recent intrusion detection approaches reported in the literature across SDN-IoT, smart agriculture, and edge-AI environments. Although direct comparison remains challenging due to differences in datasets, attack distributions, feature engineering strategies, and experimental configurations, the proposed framework demonstrates competitive classification performance while simultaneously achieving low inference latency and high computational efficiency.

Compared to conventional deep learning approaches such as CNN, LSTM, and GRU-based intrusion detection models, the proposed SNN framework achieved higher or comparable detection performance while significantly reducing computational overhead. In particular, the sparse and event-driven processing mechanism of SNNs enabled substantially lower energy consumption and faster inference compared to dense ANN-based architectures.

Furthermore, unlike several existing centralized intrusion detection solutions, the proposed framework integrates federated learning to support distributed and privacy-preserving model training across edge devices. This characteristic is particularly important in smart agricultural SDN-IoT infrastructures, where communication overhead, energy constraints, and data privacy represent major deployment challenges.

The obtained results therefore demonstrate that combining federated learning with neuromorphic computing provides an effective balance between detection capability, computational efficiency, scalability, and deployment feasibility for next-generation distributed intrusion detection systems.

7 Conclusion

This paper addressed the security challenges of SDN-enabled IoT infrastructures in smart agriculture by proposing FL-SNNIDS, a distributed intrusion detection framework integrating Federated Learning with SNNs. The proposed framework was designed to balance detection performance, computational efficiency, scalability, and privacy preservation within resource-constrained SDN-IoT environments.

Experimental evaluations conducted on the SDN-IoT and Farm-Flow datasets demonstrated that the proposed framework achieved high classification performance while maintaining low inference latency and reduced computational overhead. The obtained results further confirmed stable convergence during federated training and strong robustness under highly imbalanced traffic conditions. In addition, the sparse and event-driven processing paradigm of SNNs contributed to reducing computational activity and estimated energy consumption, making the framework suitable for edge-based smart agriculture deployments.

To further evaluate the generalization capability of the proposed framework, an external cross-dataset validation was conducted using the ASEADOS-SDN-IoT dataset under few-shot adaptation settings. Despite severe domain-shift conditions and previously unseen attack behaviours, the proposed framework maintained stable global detection performance using only a minimal amount of target-domain data. These results demonstrate the transferability and robustness of the proposed FL-SNNIDS framework across heterogeneous SDN-IoT environments.

Overall, the combination of federated learning and neuromorphic computing provides a promising direction for developing scalable, privacy-aware, and energy-efficient intrusion detection systems for next-generation smart agriculture infrastructures.

Future Work: Future research will focus on extending the proposed framework through more advanced federated optimization and aggregation strategies to better handle heterogeneous and non-IID data distributions across large-scale agricultural infrastructures. In addition, future work will investigate real-world deployment on physical SDN-IoT testbeds and dedicated neuromorphic hardware platforms to perform direct hardware-level energy measurements and evaluate real-time operational performance.

References

[1] Maanak Gupta, Mahmoud Abdelsalam, Sajad Khorsandroo, and Sudip Mittal. Security and privacy in smart farming: Challenges and opportunities. *IEEE access*, 8:34564–34584, 2020.

[2] Abbas Yazdinejad, Behrouz Zolfaghari, Amin Azmoodeh, Ali Dehghantanha, Hadis Karimipour, Evan Fraser, Arthur G Green, Conor Russell, and Emily Duncan. A review on security of smart farming and precision agriculture: Security aspects, attacks, threats and countermeasures. *Applied Sciences*, 11(16):7518, 2021.

[3] Adebunmi Okechukwu Adewusi, Njideka Rita Chiekezie, and Nsisong Louis Eyo-Udo. Securing smart agriculture: Cybersecurity challenges and solutions in iot-driven farms. *World Journal of Advanced Research and Reviews*, 15(03):480–489, 2022.

[4] Selma Amrani, Khedidja Medani, Chirihane Gherbi, and Hakim Mabed. Machine learning application in software-defined networking based agricultural internet of things: a systematic literature review. *Cluster Computing*, 28(10):647, 2025.

[5] Danish Javeed, Tianhan Gao, Muhammad Shahid Saeed, and Prabhat Kumar. An intrusion detection system for edge-envisioned smart agriculture in extreme environment. *IEEE Internet of Things Journal*, 11(16):26866–26876, 2023.

[6] Nurtay Albanbay, Yerlan Tursynbek, Kalman Graffi, Raissa Uskenbayeva, Zhuldyz Kalpeyeva, Zhastalap Abilkaiyr, and Yerlan Ayapov. Federated learning-based intrusion detection in iot networks: Performance evaluation and data scaling study. *Journal of Sensor and Actuator Networks*, 14(4):78, 2025.

[7] Rajasekhar Chaganti, Wael Suliman, Vinayakumar Ravi, and Amit Dua. Deep learning approach for sdn-enabled intrusion detection system in iot networks. *Information*, 14(1):41, 2023.

[8] G Sri Vidhya and R Nagarajan. A novel bidirectional lstm model for network intrusion detection in sdn-iot network. *Computing*, 106(8):2613–2642, 2024.

[9] Ankit Chouhan, Nashid Shahriar, and JingTao Yao. Hcl: A hybrid cnn-lstm framework for intrusion detection in sdn-iot networks. In *2025 International Conference on Computing, Networking and Communications (ICNC)*, pages 254–258. IEEE, 2025.

[10] Vinayakumar Ravi, Rajasekhar Chaganti, and Mamoun Alazab. Recurrent deep learning-based feature fusion ensemble meta-classifier approach for intelligent network intrusion detection system. *Computers and Electrical Engineering*, 102:108156, 2022.

[11] Vinayakumar Ravi, Rajasekhar Chaganti, and Mamoun Alazab. Deep learning feature fusion approach for an intrusion detection system in sdn-based iot networks. *IEEE Internet of Things Magazine*, 5(2):24–29, 2022.

[12] Prabhat Kumar, Alireza Jolfaei, and AKM Najmul Islam. An enhanced deep-learning empowered threat-hunting framework for software-defined internet of things. *Computers & Security*, 148:104109, 2025.

[13] Andrzej S Kucik and Gabriele Meoni. Investigating spiking neural networks for energy-efficient on-board ai applications. a case study in land cover and land use classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2020–2030, 2021.

[14] Souvik Kundu, Gourav Datta, Massoud Pedram, and Peter A Beerel. Spike-thrift: Towards energy-efficient deep spiking neural networks by limiting spiking activity via attention-guided compression. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 3953–3962, 2021.

[15] Anas Ali, Mubashar Husain, and Peter Hans. Federated learning-enhanced blockchain framework for privacy-preserving intrusion detection in industrial iot. *arXiv preprint arXiv:2505.15376*, 2025.

[16] Rafael Ferreira, Ivo Bispo, Carlos Rabadao, Leonel Santos, and Rogerio Luis de C Costa. Farm-flow dataset: Intrusion detection in smart agriculture based on network flows. *Computers and Electrical Engineering*, 121:109892, 2025.

[17] Alper Kaan Sarica and Pelin Angin. A novel sdn dataset for intrusion detection in iot networks. In *2020 16th International conference on network and service management (CNSM)*, pages 1–5. IEEE, 2020.

[18] Tharindu Lakshan Yasarathna and Nhien-An Le-Khac. Aseados-sdn-iot: A novel sdn-iot network intrusion detection dataset and framework. *Internet of Things*, page 101891, 2026.

[19] Geunbo Yang, Wongyu Lee, Youjung Seo, Choongseop Lee, Woojoon Seok, Jongkil Park, Donggyu Sim, and Cheolsoo Park. Unsupervised spiking neural network with dynamic learning of inhibitory neurons. *Sensors*, 23(16):7232, 2023.

[20] Kashu Yamazaki, Viet-Khoa Vo-Ho, Darshan Bulsara, and Ngan Le. Spiking neural networks and their applications: A review. *Brain sciences*, 12(7):863, 2022.

[21] João D Nunes, Marcelo Carvalho, Diogo Carneiro, and Jaime S Cardoso. Spiking neural networks: A survey. *IEEE access*, 10:60738–60764, 2022.

[22] Nadia Adnan Shiltagh and Hasnaa Ahmed Abas. Spiking neural network in precision agriculture. *Journal of Engineering*, 21(07):17–34, 2015.

[23] Matthias Schonlau and Rosie Yuyan Zou. The random forest algorithm for statistical learning. *The Stata Journal*, 20(1):3–29, 2020.

[24] Hasan Ahmed Salman, Ali Kalakech, and Amani Steiti. Random forest algorithm overview. *Babylonian Journal of Machine Learning*, 2024:69–79, 2024.

[25] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, et al. Recent advances in convolutional neural networks. *Pattern recognition*, 77:354–377, 2018.

[26] Arun D Kulkarni. Fuzzy convolution neural networks for tabular data classification. *IEEE Access*, 2024.

[27] Jason K Eshraghian, Max Ward, Emre Neftci, Xinxin Wang, Gregor Lenz, Girish Dwivedi, Mohammed Bennamoun, Doo Seok Jeong, and Wei D Lu. Training spiking neural networks using lessons from deep learning. *Proceedings of the IEEE*, 111(9):1016–1054, 2023.

[28] Chen Zhang, Yu Xie, Hang Bai, Bin Yu, Weihong Li, and Yuan Gao. A survey on federated learning. *Knowledge-Based Systems*, 216:106775, 2021.

[29] A Sandanasamy and P Joseph Charles. Distributed sdn control for iot networks: A federated meta reinforcement learning solution for load balancing. 2025.

[30] Jed Mills, Jia Hu, and Geyong Min. Communication-efficient federated learning for wireless edge intelligence in iot. *IEEE Internet of Things Journal*, 7(7):5986–5994, 2019.

[31] Sai Puppala and Koushik Sinha. Towards secure and efficient farming using self-regulating heterogeneous federated learning in dynamic network conditions. *Agriculture*, 15(9):934, 2025.

[32] Gulshan Kumar. Evaluation metrics for intrusion detection systems-a study. *Evaluation*, 2(11):11–7, 2014.

[33] Jason Yik, Korneel Van den Berghe, Douwe Den Blanken, Younes Bouhadjar, Maxime Fabre, Paul Hueber, Weijie Ke, Mina A Khoei, Denis Kleyko, Noah Pacik-Nelson, et al. The neurobench framework for benchmarking neuromorphic computing algorithms and systems. *Nature communications*, 16(1):1545, 2025.

[34] Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, et al. Loihi: A neuromorphic manycore processor with on-chip learning. *Ieee Micro*, 38(1):82–99, 2018.

[35] Abhronil Sengupta, Yuting Ye, Robert Wang, Chiao Liu, and Kaushik Roy. Going deeper in spiking neural networks: Vgg and residual architectures. *Frontiers in neuroscience*, 13:95, 2019.

[36] Kaushik Roy, Akhilesh Jaiswal, and Priyadarshini Panda. Towards spike-based machine intelligence with neuromorphic computing. *Nature*, 575(7784):607–617, 2019.

[37] Francesco Barchi, Emanuele Parisi, Luca Zanatta, Andrea Bartolini, and Andrea Acquaviva. Energy efficient and low-latency spiking neural networks on embedded microcontrollers through spiking activity tuning. *Neural Computing and Applications*, 36(30):18897–18917, 2024.

[38] Zhanglu Yan, Zhenyu Bai, and Weng-Fai Wong. Reconsidering the energy efficiency of spiking neural networks. *arXiv preprint arXiv:2409.08290*, 2024.

[39] Mark Horowitz. 1.1 computing's energy problem (and what we can do about it). In *2014 IEEE international solid-state circuits conference digest of technical papers (ISSCC)*, pages 10–14. IEEE, 2014.

[40] Min Soo Kim, Alberto A Del Barrio, Hyunjin Kim, and Nader Bagherzadeh. The effects of approximate multiplication on convolutional neural networks. *IEEE Transactions on Emerging Topics in Computing*, 10(2):904–916, 2021.

[41] Susmita Dey Manasi, Farhana Sharmin Snigdha, and Sachin S Sapatnekar. Neupart: Using analytical models to drive energy-efficient partitioning of cnn computations on cloud-connected mobile clients. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 28(8):1844–1857, 2020.

[42] Ioannis Mavromatis, Kostas Katsaros, and Aftab Khan. Computing within limits: An empirical study of energy consumption in ml training and inference. *arXiv preprint arXiv:2406.14328*, 2024.

[43] Jianwei Xue, Lisheng Xie, Faquan Chen, Liangshun Wu, Qingyang Tian, Yifan Zhou, Rendong Ying, and Peilin Liu. Edgemap: an optimized mapping toolchain for spiking neural network in edge computing. *Sensors*, 23(14):6548, 2023.

[44] Peter Blouw, Xuan Choo, Eric Hunsberger, and Chris Eliasmith. Benchmarking keyword spotting efficiency on neuromorphic hardware. In *Proceedings of the 7th annual neuro-inspired computational elements workshop*, pages 1–8, 2019.

[45] Olaf Yunus Laitinen Imanov, Derya Umut Kulali, Taner Yilmaz, Duygu Erisken, and Rana Irem Turhan. Energy-efficient neuromorphic computing for edge ai: A framework with adaptive spiking neural networks and hardware-aware optimization. *arXiv preprint arXiv:2602.02439*, 2026.