Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.081775

ARTICLE

Open-Set Intrusion Detection Solution for Industrial Internet of Things Based on Deep Spiking Q-Networks

Yimeng Liu[1], Xinyu Xu[1], Wangting Xue[1], Shigen Shen[1,*] and Xiao-Zhi Gao[2]

[1]School of Information Engineering, Huzhou Normal University, Huzhou, China
[2]School of Computing, University of Eastern Finland, Kuopio, Finland

*Corresponding Author: Shigen Shen. Email: shigens@huznu.edu.cn

Received: 09 March 2026; Accepted: 18 June 2026; Published: 23 July 2026

ABSTRACT: The rapid growth of the Industrial Internet of Things (IIoT) has become a cornerstone of high-quality global economic development. By integrating sensor networks, edge computing, and cloud intelligence, IIoT has emerged as a key enabler for smart manufacturing and digital transformation across industries. However, this technological advancement introduces significant cybersecurity challenges that render traditional intrusion detection systems inadequate for IIoT environments. To address this critical gap, we propose a deep spiking Q-network (DSQN)-based intrusion detection system (DSQN-IDS) for the IIoT, formulating unknown intrusion detection as a Markov decision process (MDP). The system employs a hierarchical multi-stage decision-making framework integrating conditional variational autoencoders (CVAE) for feature extraction, deep Q-networks (DQN) for reinforcement learning-based decision-making, and spiking neural networks (SNNs) for energy-efficient classification. We train the DSQN using a multi-layer perceptron (MLP) to approximate the state-action value function, and leverage the event-driven nature of SNNs—where neurons only spike when their membrane potential exceeds a threshold—to minimize energy consumption. Extensive experiments on IIoT datasets demonstrate that our approach achieves superior performance in balancing detection accuracy, energy efficiency, and model stability when identifying unknown attacks compared to state-of-the-art methods.

KEYWORDS: Deep spiking Q-network; pulse reinforcement learning; intrusion detection; open-set; Industrial Internet of Things

1 Introduction

The rapid proliferation of the IIoT has accelerated its deployment in intelligent monitoring and control of industrial processes, emerging as a critical driver of economic growth and digital transformation [1]. However, as IIoT systems transition from closed industrial control system (ICS) architectures to open network environments, the expanded attack surface has exposed numerous security vulnerabilities [2]. According to the CVE database's ICS vulnerability subset, 9517 security flaws were documented across various industrial systems by September 2025, with 46.5% classified as critical, 45.6% as medium severity, and 7.9% as low severity. This substantial vulnerability landscape underscores the urgent need for advanced security mechanisms. Consequently, significant research efforts have focused on enhancing IIoT security through privacy protection policies [3], service trust mechanisms [4], and sophisticated intrusion detection methodologies [5]. Our work addresses these critical challenges by developing advanced intrusion detection

techniques that protect IIoT systems against evolving cyber threats, including attacks from compromised edge devices and unauthorized data exfiltration attempts.

To address zero-day attacks within the IIoT, Shen et al. [6] developed a dynamic defense framework that integrates Stackelberg game theory with multi-agent deep reinforcement learning (DRL), specifically employing a distributed deep deterministic policy gradient to coordinate deceptive defenses. Farea et al. [7] proposed an AI-driven security system named "AI2AI" that employs artificial intelligence techniques to enhance performance and optimize security mechanisms within IoT frameworks. Dhaouadi et al. [8] introduced the HiTar-2024 dataset specifically designed for IIoT systems, which was generated using the AREZZO simulator to replicate real smart manufacturing scenarios, enabling IDS to detect emerging threats.

However, one shortcoming of current intrusion detection work is that when reinforcement learning processes continuous action spaces, it usually requires a lot of computing resources, which is computationally expensive and even cannot accurately identify known attacks. A typical study incorporates target networks, twin Q-learning strategies algorithms, alongside experience replay mechanisms. It integrates the CVAE framework into the value estimation network within the DQN, and conducts evaluations using the latest IIoT dataset, dataset collection, TON-IoT [9].

To make up for the shortcomings of the tight ensemble solution, this work embeds deep spiking reinforcement learning within the open-set intrusion detection paradigm for IIoT systems through the application of an ANN-to-SNN conversion mechanism to the DQN's value function representation. The experimental findings demonstrate that our presented DSQN-IDS framework attains superior performance on the TON-IoT benchmark dataset. The key innovations presented in this paper include:

1. We use normalization to process weights and activation values and initialize the parameters of the DSQN, and build a suitable ANN to prepare for the initialization of the DSQN. We further convert the ANN into an SNN and use the weights of the pre-trained ANN as the initialization of the universal feature representation loaded into the SNN. We adjust the behavior of neurons and the structure of the model to improve the training efficiency and the performance of the open-set intrusion detection model using the DSQN.
2. The CVAE module extracts and reduces dimensionality of features, the DQN module optimizes decision-making through reinforcement learning, and the SNN module ensures energy-efficient classification, forming a robust pipeline for intrusion detection. We frame the open-world recognition issue in IIoT security monitoring as a temporal MDP, proposing DSQN to address this challenge. We further incorporate target networks, dual Q-learning, along with an trajectory replay buffer to enhance the model's efficacy in detailed discrimination categorization of legitimate traffic and detection of unknown threats.
3. We evaluated our method using the TON-IoT dataset. Experimental results show that our proposed method has improved performance over previously proposed methods. This indicates that our framework can reduce the energy expenditure while preserving acceptable accuracy in detailed classification of known traffic flows. At the same time, our method can ensure high accuracy in identifying unknown attacks and minimize the operational burden on IIoT security teams.

The rest of this paper is organized as follows. Section 2 gives an overview of related work. Section 3 elaborates on the model architecture of the DSQN-IDS framework and outlines the detailed procedures for both training and testing. Section 4 outlines the design of our experiments and experimental results. Section 5 draws conclusions. Table 1 summarizes the mathematical symbols used in this paper.

Table 1: Mathematical notations used in this paper.

Symbol	Description
d_s	Dimension of the state space
d_a	Dimension of the action space
α_{lr}	Learning rate
γ	Discount factor
ϵ	Exploration rate in ϵ-greedy policy
$Q(s,a)$	Action-value function
s	Current state
a	Action taken
r	Reward received
s'	Next state
θ_{eval}	Parameters of the evaluation network
θ_{target}	Parameters of the target network
τ	Soft update rate for target network
N	Batch size
$p_\theta(z\|s)$	Prior distribution over latent variables
$q_\phi(z\|s,a)$	Recognition (posterior) distribution over latent variables
$\mu_1(s), \mu_2(s,a)$	Mean outputs of neural networks for prior and recognition distributions
z	Latent variable
ξ	Random noise for reparameterization trick
V_{th}	SNN firing threshold
$A_{\max}$	Maximum activation
α_{norm}	Weight normalization scaling factor
W_{ANN}	Weights in the ANN
W_{SNN}	Weights in the SNN
$\mathbf{a}^{(l)}$	Activation vector of layer l
F_{ANN}	Activation function of the ANN
$u^{(l)}(t)$	Membrane potential at time step t
$s_i^{(l)}(t)$	Spike output
H	Heaviside step function
$v_i^{(l)}(t)$	Updated membrane potential after a spike
S	Total number of spikes generated by the SNN
E_{ANN}	Energy consumption of the ANN
E_{SNN}	Energy consumption of the SNN
β	Energy coefficient per spike
R	Energy ratio of ANN to SNN
$\mathcal{F}_{\text{ANN}}$	Pretrained ANN model
$\mathcal{D}$	Dataset
T	Number of time steps
A_l	Activations for each layer l
θ_l	SNN membrane thresholds
s_l	Layer-specific scaling factor

(Continued)

Table 1 (continued)

Symbol	Description
$b_{\text{SNN}}^{(l)}$	Biases in the SNN
$\hat{y}$	Decoded final output
N_l	Number of neurons in layer l
L	Number of layers
x	Shared input data
α_{energy}	Energy coefficient per FLOP for ANN
y	Output of a layer in the SNN

2 Related Work

In this section, we review intrusion detection technologies for IIoT, spiking neural networks, and ANN-to-SNN conversion.

2.1 Intrusion Detection Technologies for IIoT

Compared with traditional networks, the characteristics of IIoT are that it integrates diverse devices, has a complex network structure, and has limited computing power and storage capacity. In view of these characteristics, researchers have proposed many intrusion detection methods for application in the IIoT field [10]. Park et al. [11] proposed a novel AI-based NIDS that utilizes a generative adversarial network with reconstruction error and Wasserstein distance to effectively address the data imbalance. Mao et al. [12] proposed a novel edge client weighted transmission optimization strategy, which cleverly balances effective intrusion detection with the operational constraints of IIoT devices, ensuring the comprehensive detection capability of the global model against various network attacks.

Afterwards, Zhang et al. [13] proposed an anomaly-based IDS combined with federated learning, using a ranking aggregation algorithm with a weighted voting method to improve the performance of the system in IIoT intrusion detection. Wang et al. [14] developed an online incremental training method and constructed an autoencoder-based intrusion detection model, which can detect whether a node is attacked more quickly and take emergency response measures in a timely manner based on the detection results. Mohy-Eddine et al. [15] designed an intrusion detection model that uses feature engineering and machine learning to ensure IIoT security. It uses a combination of isolation forest (IF) and Pearson correlation coefficient.

Recent research has made significant progress in enhancing IIoT security through advanced detection algorithms and deep learning techniques. Borgiani et al. [16] proposed a distributed congestion control based on duty cycle limitation method to detect and mitigate denial-of-service (DoS) attacks in IIoT, which can be used to mitigate DoS attacks in a sensor network scenario with 500 nodes. Yan et al. [17] designed a hinge classification algorithm based on mini-batch gradient descent with adaptive learning rate and momentum to minimize the impact of security attacks and significantly improve the performance of deep network training in terms of scale and speed. Zhu et al. [18] developed an enhanced algorithm RT-A3C, which transforms the traditional turn-based Markov game into a real-time reactive game, improving the practicality and efficiency of IIoT security simulation training. All of the above methods focus on attack detection and deep learning model construction, and have improved the efficiency over legacy methods. However, high-dimensional IIoT datasets typically impose stricter hardware demands, contradicting the inherent resource constraints of IIoT environments. There remains still room for optimization regarding its computational overhead.

2.2 Spiking Neural Networks

The SNN is a neural network model that combines the efficient information processing mechanism of the biological nervous system with the computing power of artificial intelligence. Researchers have applied this model in many fields. Zhang et al. [19] used the desired signal to induce specific levels of neuromodulators to selective synapses and proposed an efficient, brain-inspired SNN and ANN computing algorithm that achieves high recognition accuracy and significantly reduces computational costs when learning spatial and temporal classification tasks. Liu et al. [20] proposed a directly trained DSQN that achieves human-level control capabilities in a reinforcement learning environment. This method takes advantage of the efficient and low-energy information processing advantages of SNN.

Then, Shen et al. [21] proposed a hyper-game-theoretic malware propagation mitigation model for active deception in edge intelligence-enabled IoT systems under information asymmetry, which breaks through the bottleneck of time processing in traditional deep neural network models and realizes intelligent decision-making and real-time malware defense. In contrast to existing approaches, the DSQN-based intrusion detection model presented herein balances the critical aspects of recognizing unseen attacks and the algorithm's resource footprin, demonstrating superior efficacy.

2.3 Hybrid Cyber Threats and DSQN-IDS Framework

The IIoT enhances efficiency and automation but simultaneously exposes interconnected systems to escalating cyber threats, particularly complex hybrid cyber threats. Mohammed et al. [22] investigated three variants of the Kolmogorov–Arnold Network (KAN) for intrusion detection using the RT-IoT2022 dataset. They implemented a Simple-KAN network as a lightweight solution for resource-constrained IoT devices, evaluated a Transformer-KAN model leveraging attention mechanisms to capture temporal patterns, and developed a GKAN to model device-to-device relationships through graph representations. Liu et al. [23] proposed a machine learning-based framework for modeling and detecting HCTs, employing regularization and random forest for feature importance analysis, alongside grey relation analysis to establish correlations between IIoT components and various threats. Zhukabayeva et al. [24] proposed an impact-aware taxonomy-driven machine learning framework incorporating edge deployment and SHapley Additive exPlanations (SHAP)-based Explainable AI (XAI) for attack detection in IIoT-CPS environments. The framework employs unsupervised clustering (K-Means and DBSCAN) to extract latent traffic patterns and supervised classification to categorize 33 attack types into seven high-level categories: Flood Attacks, Botnet/Mirai, Reconnaissance, Spoofing/Man-In-The-Middle (MITM), Injection Attacks, Backdoors/Exploits, and Benign. In contrast to these approaches, our proposed DSQN-IDS leverages the event-driven nature of SNNs to reduce energy consumption at a modest trade-off in detection performance. By formulating the intrusion detection problem as an MDP and integrating a CVAE for feature extraction, our method achieves a superior trade-off between detection accuracy and computational efficiency.

2.4 General Framework of the Model

We propose the model called DSQN-IDS, which implements a DSQN through an ANN-to-SNN transition framework. The model's primary goal is to reduce the energy consumption of authorized traffic and improve the granularity of known attack traffic classification. This approach introduces the multi-stage decision-making framework and its components, including CVAE for feature extraction, DQN for decision optimization, and SNN for energy-efficient classification, as shown in Fig. 1.

We define the environmental state representation and the training methodology in the proposed framework. The state space model constitutes a proprietary representation of the DRL system, containing the environment state model for the MDP and the reward generation rule. A deep learning architecture is used

to approximate the Q-function. The agent's choice that maximizes the value estimate in the existing state is selected from the output of the deep neural network. The decision that maximizes the expected return in the given state is thereby determined to be performed by the agent. This optimization methodology enhances the intrusion detection agent's action decision-making strategy hrough iterative assessment of the state-action value. The intrusion detection agent's action decision-making strategy is then further optimized by calculating the loss function using the rewards or penalties fed back by the environment state model, which is used to update the parameters of the value network. We then replace the ReLu activation function in the ANN with spiking neurons in the SNN. Based on the similarity between the interfering neurons in the spiking neuron model and ANNs, the trained ANN weights are mapped to the SNN. Finally, we use the constructed DSQN to train the granular traffic categorization component and the novel threat identification component to recognize unknown attacks. This module is used to obtain the final detailed in-traffic discrimination and anomaly recognition model.

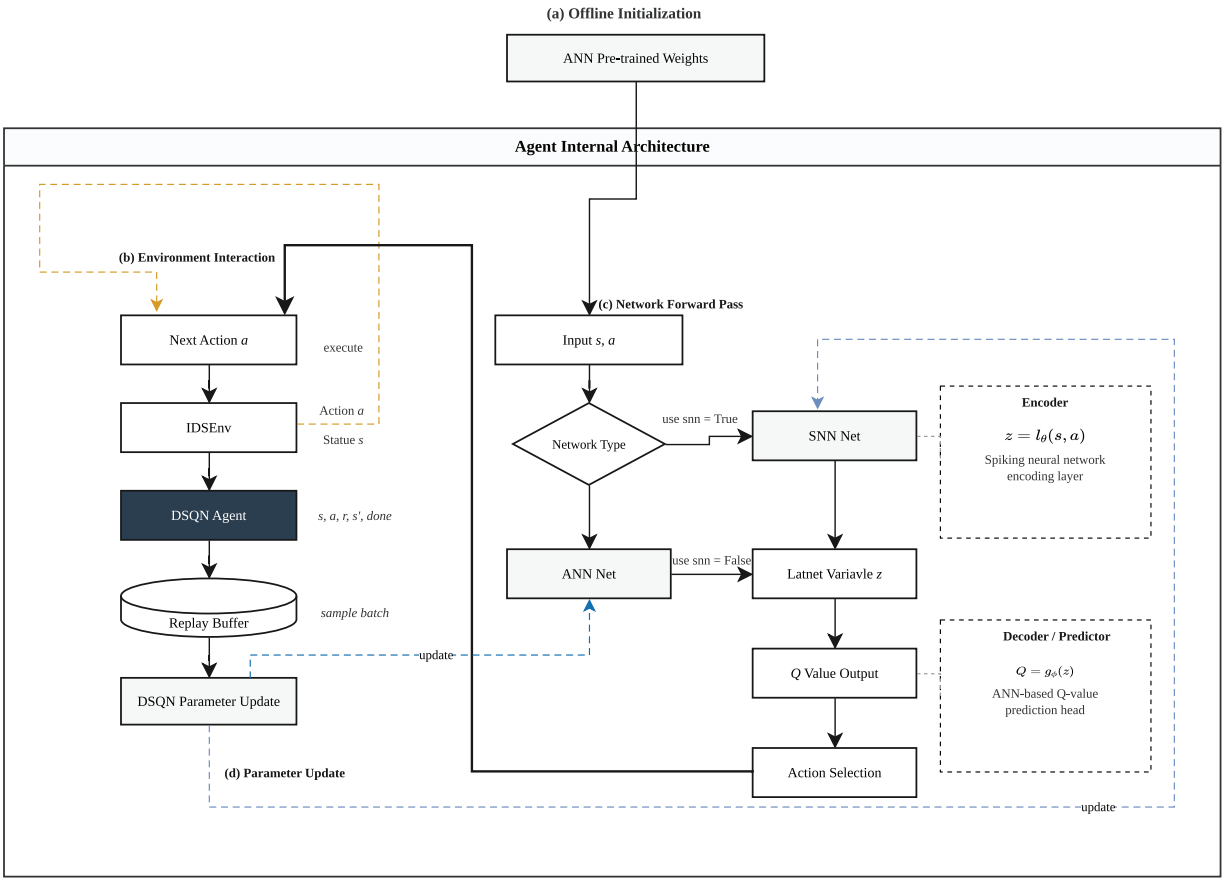

Figure 1: Network intrusion detection system architecture integrating CVAE-DQN-SNN.

We next describe the details of the known traffic and unknown attack detection process. First, we train a high-performance ANN model using the DRL algorithm DQN. We then define a hook function tailored to the threat identification objective, recording the appropriate threshold value calculated by the ANN activation values during training data to prevent the membrane potential of the SNN neurons from exceeding the threshold. Finally, the recorded activation values are used for weight normalization in the SNN to prevent weights from varying too large or too small, which could cause the membrane potential to update too quickly or too slowly, affecting network performance. Weight normalization is performed by scaling

the weights to an appropriate range. By removing the bias term from the ANN, using dropout to prevent overfitting, and switching the pooling layer to an average pooling operation, we ensure that the converted SNN can effectively process spike signals and maintain performance comparable to the original ANN.

3 Methodology

This section presents the introduced DSQN-IDS, which features low energy consumption and high accuracy in detailed discrimination of legitimate traffic and identification of unknown attacks. Firstly, we introduce the overall architecture and core techniques employed in the algorithm. Afterwards, we describe algorithmic learning phase. Finally, we describe the performance loss in the detailed categorization of authorized traffic and the efficacy in unknown attack identification.

3.1 ANN Initialization and SNN Energy Optimization Training Algorithm

3.1.1 ANN-Powered Intrusion Detection in IIoT Systems

We now present our main ANN based Q-network that we use for our IIoT threat identification tasks shown in Fig. 2. It is an instantiation of the Q-network for IIoT traffic data. It consists of three parts: a recognition network, a prior distribution network, and the generative network. The three modules in partnership with IIoT data learn the relationship of state and action of the IIoT environment from training data sets. Our next step is to define neural networks; we are using Q-network, built as a multi-layered fully connected ANN. Our hidden layer uses neurons with each successive level learning through Softplus activation function so that complicated nonlinear relationships in IIoT datasets will be extracted automatically. Our second stage involves initializing both the environment as well as the set of necessary parameters to supply it at initial time, in this case referring to an IIoT scenario simulation. The environment simulates the IIoT context, with state dimension d_s, action dimension d_a, learning rate α, discount factor γ, and other critical parameters specified at the outset to ensure reproducibility and robust learning. This framework incorporates a dual-path structure where the prior network generates the latent variable Z_1 and the recognition network generates the latent variable Z_2, ultimately reconstructing the state output s^* through the decoder, forming a complete generative model computational pipeline. Third, an experience replay buffer is constructed to store tuples $(s, a, r, s', \text{done})$ generated from agent-environment interactions. Experience replay enables sample decorrelation and efficient mini-batch training, which is essential for stable convergence of the DQN. Fourth, the DQN agent is instantiated, containing both an evaluation network and a target network, which are periodically synchronized. Advanced variants such as double DQN or dueling DQN may be employed to further enhance training stability and performance.

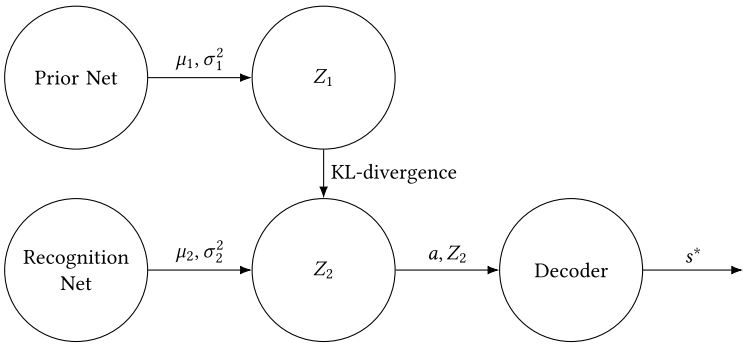

Figure 2: Schematic of the generative model with KL-divergence.

Early in training, the policy parameter ϵ is set high to encourage exploration and is gradually annealed to favor exploitation. Upon executing action a, the DQN agent receives the next state s', reward r, and terminal indicator done, and the tuple $(s, a, r, s', \text{done})$ is stored in the replay buffer. At specified intervals, the DQN agent samples a mini-batch from the buffer and updates the Q-network parameters via gradient descent. The Bellman equation is used to compute the target Q-value. The target Q-value used for temporal difference learning, Q_{target}, is

$$Q_{\text{target}} = r + \gamma \cdot (1 - \text{done}) \cdot \max_{a'} Q_{\text{target\_net}}(s', a') \tag{1}$$

where r is the reward, γ is the discount factor, done indicates whether the episode is terminated, s' is the next state, a' is the possible action in the next state, and $Q_{\text{target\_net}}(\cdot)$ denotes the target Q-network.

The learning objective is to minimize the mean squared error between the predicted and target Q-values. The mean squared error loss, $\mathcal{L}$, is

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \left(Q_{\text{eval}}(s_i, a_i) - Q_{\text{target},i} \right)^2 \tag{2}$$

where N is the batch size of sampled experiences, i means the sample index in the mini-batch, $Q_{\text{eval}}(s_i, a_i)$ means the Q-value predicted by the evaluation network for state s_i and action a_i, and $Q_{\text{target},i}$ is the target Q-value computed from the target network using Eq. (1).

To maintain training stability, the target network parameters are periodically updated using Polyak averaging, which smoothly blends the evaluation and target network parameters, represented by

$$\theta_{\text{target}} \leftarrow \tau \theta_{\text{eval}} + (1 - \tau) \theta_{\text{target}} \tag{3}$$

where θ_{target} and θ_{eval} represent the parameters of the target and evaluation networks, respectively, and τ means the Polyak update rate controlling the blending of parameters between the two networks.

Policy evaluation is conducted at regular intervals using a separate evaluation environment, recording the reward curve and dynamically adjusting the exploration rate ϵ as training progresses. Upon completion of the maximum number of training steps, the trained ANN parameters are saved, and the model is evaluated on the test set for final performance assessment. In summary, this pipeline implements a robust and scalable ANN-based DRL framework for IIoT intrusion detection, leveraging experience replay, target networks, and effective exploration strategies to achieve stable and generalizable learning.

3.1.2 MDP for IIoT Intrusion Detection

In our framework, the IIoT intrusion detection problem is formulated as an MDP, which is defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma)$. Here, $\mathcal{S}$ is the state space, where each state s is a feature vector representing the current environment observation. $\mathcal{A}$ denotes the discrete action space, corresponding to possible detection outcomes such as normal or specific attack types. r is the reward function reflecting detection accuracy. $\mathcal{P}$ specifies the state transition dynamics. γ is the discount factor. At each timestep, the DQN agent receives state s and selects action a following an ϵ-greedy policy as

$$a = \begin{cases} \text{random action}, & \text{if total\_steps} < \text{random\_steps} \\ \arg\max_a Q(s, a), & \text{with probability } 1 - \epsilon \\ \text{random action}, & \text{with probability } \epsilon \end{cases} \tag{4}$$

where a is the selected action, s is the current state, $Q(s, a)$ is the action-value function, $\arg\max_a Q(s, a)$ means the action that maximizes the Q-value, $\epsilon \in [0, 1]$ is the exploration rate, random action is sampled uniformly from the action space, total_steps is the number of steps taken so far during training, and random_steps is a predefined threshold for initial random exploration.

The training process proceeds through the following iterative steps, forming a complete reinforcement learning cycle:

- Action Selection: The DQN agent determines the action a based on the current state s. Initially, actions are selected randomly to encourage exploration. As training progresses, an ϵ-greedy strategy is adopted: with probability $1 - \epsilon$, the agent selects the action maximizing the Q-value; with probability ϵ, it chooses randomly. This balances exploration and exploitation.
- KL Regularization (Variational Inference): To enhance generalization and robustness, the kullback-leibler (KL) divergence between $q_\phi(z|s, a)$ and $p_\theta(z|s)$ is used as a regularization term. This ensures the learned latent space is structured and meaningful, improving detection performance.
- Environment Interaction: The intrusion detection agent interacts with the environment by executing action a, receiving the next state s', reward r, and terminal flag done. This simulates real-time feedback in IIoT systems.
- Experience Storage: The replay buffer stores each transition tuple $(s, a, r, s', a_{\text{hot}}, \text{done})$ generated during agent-environment interaction. This mechanism enables efficient experience reuse and reduces sample correlation.
- State Update: After each interaction, the DQN agent updates its state s to the new s' and prepares for the next decision.
- Network Update: At intervals, the variational DQN agent samples mini-batches from the buffer to update Q-network parameters. The loss function combines Q-value error and KL regularization, minimized by gradient-based optimization.
- Policy Evaluation: Policy performance is periodically evaluated in a separate environment. Results guide ϵ adjustment and monitor training progress.

In addition, the proposed framework explicitly addresses the dynamic and non-stationary nature of IIoT environments. Temporal dependencies are modeled through sequential state representations and short-term history encoding, while the policy is trained on diverse attack scenarios to improve generalization. Moreover, uncertainty-aware detection scores are integrated into the decision-making process, allowing the agent to adapt to previously unseen or evolving attack behaviors. These enhancements make the MDP formulation sufficiently expressive and robust for real-world interactive intrusion detection tasks.

Then the DQN agent receives the reward r and next state s', and the transition tuple $(s, a, r, s', a_{\text{hot}}, dw)$ is stored in the replay buffer for further training as

$$p_\theta(z|s) \sim \mathcal{N}(\mu_1(s), \sigma_1^2(s)I), \tag{5}$$

$$q_\phi(z|s, a) \sim \mathcal{N}(\mu_2(s, a), \sigma_2^2(s, a)I) \tag{6}$$

where $\mu_1(\cdot)$, $\mu_2(\cdot)$ and $\sigma_1^2(\cdot)$, $\sigma_2^2(\cdot)$ are outputs of neural networks.

We next show the process of generating z and calculating KL divergence, as illustrated in Fig. 3. First, we calculate the mean function $\mu_2(s, a)$ with a state s and an action a, which represents the expected value of the generated variable z given the state and action. This mean function forms the deterministic part of the generation process. Next, we compute the variance function $\sigma^2(s, a)$, which controls the randomness of the generated variable z. The larger the variance, the greater the fluctuation range of the generated z values. Then, we further generate a random variable $\epsilon \sim N(0, I)$ with mean 0 and variance 1. It is the random part. After

that, we use the equation $z = \mu_2(s,a) + \sigma^2(s,a) \odot \epsilon$ to mix the mean value, variance, and noise ($\odot$ represents elementwise multiplication of the vector). The generated variable z can then be used in tasks. Finally, we calculate D_{KL} between z distribution and an estimated target distribution to optimize the generation process with this z, thus allowing us to optimize the mean function and variance function such that the generated z is similar to z target distribution, so as to also allow z to be further used in other tasks. We further compute D_{KL} to assess the KL-divergence distribution of the generated variable z and optimize this estimation process to optimize its probability density function, we can finally optimize the generation process of the latent sample z using the reparameterization trick, z, is

$$z = \mu_2(s,a) + \sigma^2(s,a) \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I). \tag{7}$$

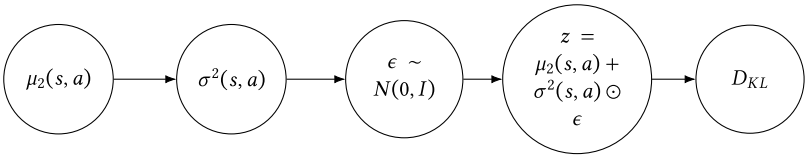

Figure 3: Process of generating z and calculating KL divergence.

To enhance generalization and robustness, we incorporate variational inference via KL regularization as

$$\mathrm{KL}\left[q_\phi(z|s,a) \,\|\, p_\theta(z|s)\right] = \int q_\phi(z|s,a) \log \frac{q_\phi(z|s,a)}{p_\theta(z|s)} dz \tag{8}$$

where $\mathrm{KL}[\cdot|\cdot]$ is the KL divergence, $q_\phi(\cdot)$ and $p_\theta(\cdot)$ are the variational posterior and prior distributions over the latent variable z parameterized by neural network parameters ϕ and θ, respectively, and the integral is taken over the entire latent space of z.

So far, this MDP-based training process enables the variational DQN agent to iteratively improve its intrusion detection policy by leveraging both reinforcement learning and variational inference, thereby enhancing detection accuracy and model robustness.

However, it is important to acknowledge the limitations of modeling intrusion detection strictly as an MDP. The MDP framework assumes well-defined states and a stationary transition probability, which may not fully capture the highly dynamic and stochastic nature of evolving IIoT attack patterns. As observed in the experimental results, while the model performs robustly on known attack scenarios, the rigidity of the state representation can lead to suboptimal decisions when facing novel or rapidly mutating attack vectors.

3.1.3 The Pareto Analysis Confirms

The plot uses detection performance true positive rate (TPR) at false positive rate (FPR) = 0.10 and AUROC on the vertical axis and absolute energy consumption on the horizontal axis. This allows direct visualization of the energy-performance trade-off and shows that DSQN-IDS lies on the Pareto frontier, achieving higher detection performance with lower or comparable energy than other methods. In addition, to quantitatively evaluate the efficiency of each method in terms of detection quality per unit energy, we introduce the Energy-Performance Score defined by

$$\text{Energy-Performance Score} = \frac{\text{AUROC}}{E_{\text{total}}} \tag{9}$$

where E_{total} is the absolute training energy consumption. This metric quantifies the efficiency of each method in terms of detection quality per unit energy.

The Pareto analysis confirms that DSQN-IDS achieves a superior trade-off: it provides the highest detection performance among neural methods while maintaining substantially lower energy consumption than CVAE-based baselines.

3.1.4 Weight Normalization

In ANN-to-SNN conversion, a direct weight transfer often causes large numerical discrepancies because the activation statistics of the analog units and the integrate-and-fire neurons differ. To prevent the SNN from either firing too sparsely or too profusely, we apply a global weight normalization stage with the goal of aligning the distribution of the SNN membrane potentials—just before a spike—with the distribution of ANN activations at the same layer under the chosen firing threshold. The procedure consists of three core steps. First, we collect ANN activations for a representative set of inputs and record their maximum value. Next, we compute the corresponding maximum membrane potential that the converted SNN would reach with the same inputs and the current weights. Finally, we rescale all outgoing weights of the layer by the ratio of these two statistics so that, after scaling, the SNN membrane potential is expected to remain just below the threshold for the same inputs. This process effectively aligns the two distributions and preserves the network's original decision boundaries.

For a reference layer in the ANN, all post-activation values are collected over the training data. The SNN firing threshold V_{th} is set as a high percentile of these activations, and the maximum activation $A_{\max}$ is also recorded. The normalization factor α is then defined as

$$\alpha = \frac{V_{\text{th}}}{A_{\max}}. \tag{10}$$

The weights of the SNN are obtained by globally scaling the ANN weights, W_{SNN}, is

$$W_{\text{SNN}} = \alpha \cdot W_{\text{ANN}} \tag{11}$$

where W_{ANN} means the weights of analog neural network, respectively. To avoid membrane potentials exceeding the threshold, a clamping operation is applied at each SNN layer, ensuring $V \leq V_{\text{th}}$ during forward inference.

Unlike the standard weight normalization technique that normalizes the norm of each weight vector and adds a learnable parameter used to accelerate convergence speed, the above operation scales all weights together via multiplying α in one step. This is not for optimizing its optimization property, but just aligning the value scale of neurons of ANN with that of SNN to enable it to be converted from the two kinds of ANNs. Our described normalization operation would reduce the difference between ANN and SNNs about their distributions of activation values in the same network layer. It ensures the accuracy and stability of results in the converted SNNs via weighting by normalization process for a single entire neural network. This ensures stable and accurate SNN inference, and is widely adopted in practical conversion pipelines.

We emphasize that the stages are not independent, because ANN pretraining provides a stable and efficient feature representation for known traffic classes and reduces the risk of poor initialization in the subsequent SNN stage, ANN-to-SNN conversion preserves the pretrained decision boundaries while enabling event-driven, low-energy inference and thus creates a transferable architecture compatible with neuromorphic execution, and RL-based optimization then fine-tunes the converted SNN policy in the actual interactive detection environment to align the model with the final task objective and operational reward

structure. This design maintains the learned weights and activation patterns from ANN pretraining while optimizing the same model parameters under task-specific reward signals, thereby ensuring coordination among representation learning, energy-efficient inference, and detection performance.

3.2 Artificial Network Transformation into Spiking Network

3.2.1 Activation Statistics and Normalization Process

We collect statistic values of each action based on fast forward propagation over a pre-trained ANN as shown as in Fig. 4 which collects the value of every neuron in one layer at a time. These values are used to calculate a suitable normalization coefficient to determine a normalized value of the corresponding neuron and also find out the maximum value of activation to specify the threshold of activation we use in the following parts. Through this processing the input data's and output data of SNN value range matches with the value ranges of both corresponding network layers in ANN, respectively, which will be used as the approximate standard value in subsequent weight conversion for future application. The forward propagation can be mathematically represented as

$$\mathbf{z}^{(l)} = W^{(l)}\mathbf{a}^{(l-1)} + \mathbf{b}^{(l)} \tag{12}$$

where $W^{(l)}$ is the weight matrix, $\mathbf{a}^{(l-1)}$ is the activation vector from the previous layer $(l-1)$, and $\mathbf{b}^{(l)}$ is the bias vector at layer l.

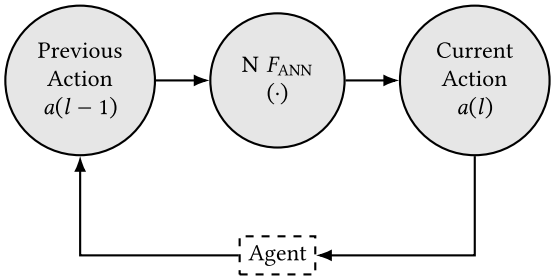

Figure 4: Process of generating current action $a(l)$ using an ANN.

Subsequently, the activation output $\mathbf{a}^{(l)}$ is obtained by applying the activation function as

$$\mathbf{a}^{(l)} = F_{\text{ANN}}(\mathbf{z}^{(l)}) \tag{13}$$

where F_{ANN} means the activation function of the analog neural network, and $\mathbf{z}^{(l)}$ is the pre-activation input vector from Eq. (12).

3.2.2 Weight Conversion

These weights are transferred to SNN. Using the normalization factor computed above, these weight parameters will be scaled linearly. The purpose is to keep the same relative strength of the activation in each layer. For bias, we copy it directly. Therefore, the original knowledge representation is preserved in this step while matching the spiking neuron characteristics of the network.

3.2.3 Spiking Neuron Implementation Principle

The LIF neuron model is used to simulate the membrane potential dynamics of biological neurons. Each neuron accumulates input signals and fires a pulse and resets when the membrane potential exceeds a

threshold. This event-driven mechanism significantly reduces redundant computation. Neuron parameters are carefully tuned to ensure activation patterns similar to those of original ANNs. The membrane potential update in an SNN, $u^{(l)}(t)$, is

$$u^{(l)}(t) = v^{(l)}(t-1) + W^{(l)}x^{(l-1)}(t) \tag{14}$$

where $v^{(l)}(t-1)$ is the membrane potential after reset at the previous time step $t-1$, $W^{(l)}$ is the weight matrix, and $x^{(l-1)}(t)$ is the input spike train from the previous layer $l-1$ at time t.

3.2.4 Complete Inference Process

The CVAE module serves as the feature extraction backbone, learning compact latent representations that capture the underlying data distribution. These representations are used as inputs to the DQN module, which learns optimal decision-making policies under dynamic conditions. The integration of the SNN module enhances decision-making efficiency by processing temporal dependencies in the data. Unlike traditional neural networks, the SNN leverages spiking dynamics to encode and process sequential information, which is particularly beneficial in scenarios with temporal constraints. The converted SNN uses a temporal encoding strategy, accumulating input signals over multiple time steps. The spike activity at each time step is integrated, ultimately producing network output comparable to that of an ANN. This entire process maintains the recognition accuracy of the original network while leveraging the high energy efficiency of SNNs. After inference concludes, the neuron's membrane potential state is reset, preparing for the next inference. The reset mechanism is given as

$$v_i^{(l)}(t) = u_i^{(l)}(t) - s_i^{(l)}(t)V_{\text{th}}^{(l)}. \tag{15}$$

The pseudocode for ANN-to-SNN conversion is presented in Lines 1–28 of the aforementioned Algorithm 1. Subsequently, we theoretically examine the computational overhead of the introduced method. The algorithmic complexity inherent in the conversion procedure is primarily determined by the number of layers and parameter scale of the ANN model, and its complexity can be expressed as $O(L \times N)$, where L represents the total count of network layers and N is the average amount of parameters per layer. In summary, we have achieved efficient conversion from ANN to SNN. Compared with traditional ANN, the converted SNN has event-driven characteristics and can significantly reduce energy consumption. Specifically, through activation statistics and weight normalization, the representation ability of ANN is maintained while the energy efficiency advantage of SNN is obtained.

Algorithm 1: ANN-to-SNN conversion and evaluation

Input: Pretrained ANN $\mathcal{F}_{\text{ANN}}$; dataset $\mathcal{D}$; time steps T
Output: Converted SNN $\mathcal{F}_{\text{SNN}}$; energy statistics
1: Copy architecture from $\mathcal{F}_{\text{ANN}}$ to $\mathcal{F}_{\text{SNN}}$;
2: Replace activations with spiking neurons via Eq. (5);
3: Verify layer and connection alignment;
4: **for** $(x, y) \in \mathcal{D}$ **do**
5: Record activations $A_l \leftarrow \mathcal{F}_{\text{ANN}}(x)$;
6: **end for**
7: Compute $a_l^{\max} \leftarrow \max(A_l)$;
8: Set $\theta_l \leftarrow \text{percentile}(A_l, 99.9\%)$;
9: Normalize network parameters using θ_l and $a_l^{\max}$;

(Continued)

Algorithm 1 (continued)

10: **for** each layer l in the network **do**
11: Calculate the layer-specific scaling factor $\alpha_l = \theta_l / a_l^{\max}$ via Eq. (10);
12: Scale the weights $W_l^{\mathrm{SNN}} \leftarrow \alpha_l \cdot W_l^{\mathrm{ANN}}$ via Eq. (11);
13: Normalize biases using scaling factor α_l;
14: **end for**
15: Configure spiking neuron dynamics with the reset mechanism;
16: **for** each spiking neuron layer l in $\mathcal{F}_{\mathrm{SNN}}$ **do**
17: Implement the reset mechanism $v_i^{(l)}(t) = u_i^{(l)}(t) - s_i^{(l)}(t) V_{\mathrm{th}}^{(l)}$ via Eq. (15);
18: **end for**
19: **for** $t = 1$ to T **do**
20: Encode $x^{(t)} \leftarrow \mathrm{encode}(x)$;
21: Propagate through $\mathcal{F}_{\mathrm{SNN}}$;
22: Record $s^{(t)}$;
23: **end for**
24: Decode final prediction from accumulated spike counts;
25: Compute E_{ANN};
26: Count $S \leftarrow \sum_{t=1}^{T} \sum_l \mathrm{spikes}_l(t)$;
27: Compute $E_{\mathrm{SNN}} \leftarrow \beta \cdot S$;
28: Compute $R \leftarrow E_{\mathrm{ANN}} / E_{\mathrm{SNN}}$;
29: Evaluate classification accuracy on the test set;
30: Generate the confusion matrix for per-class analysis;
31: Analyze accuracy-energy tradeoff in terms of time steps T;
32: **return** $\mathcal{F}_{\mathrm{SNN}}$, energy statistics

3.2.5 Energy Calculation Method

Network energy cost can be evaluated by counting the number of pulses in the pulse domain, since each pulse will cost fixed amount of energy consumption, and SNN only perform calculation if it is necessary, which brings an improvement of energy saving compared to the traditional floating-point operation in ANNs. Then, after using the weights, how much energy consumption a network would spend in its total execution during the inference process can be figured out by the pulse counting method. Spiking is performed based on

$$s_i^{(l)}(t) = H(u_i^{(l)}(t) - V_{\mathrm{th}}^{(l)}) \tag{16}$$

where $H(\cdot)$ is the Heaviside step function, $u_i^{(l)}(t)$ is the membrane potential of neuron i before reset, and $V_{\mathrm{th}}^{(l)}$ is the firing threshold voltage of layer l.

3.2.6 Accuracy-Energy Trade-Off Results

As shown in Table 2, we analyze the trade-off among detection performance, energy consumption, and inference latency under different time steps T. The results demonstrate that smaller T values yield low latency and energy consumption but suffer from poor decision quality, while larger T values improve performance at the cost of increased computational overhead. Notably, the configuration with $T = 80$ achieves the best overall trade-off among the tested configurations, with $T = 80$ balancing reward, energy, and latency. Under this setting, the proposed model attains the highest reward 6.20, while maintaining

moderate energy consumption 80 μJ and acceptable inference latency 49.24 ms. In contrast, lower T values result in significantly degraded performance, and excessively large T values T = 100 introduce higher latency without consistent performance gains. These results confirm the existence of an optimal temporal configuration, where the balance between efficiency and effectiveness is maximized.

Table 2: Trade-off between reward, energy consumption, and latency.

Configuration	Time Steps (T)	Reward	Energy (J)	Time (ms)
Low Efficiency	10	−54.20	1.0×10^{-5}	5.54
Medium Performance	60	−0.60	6.0×10^{-5}	27.72
Balanced (Ours)	80	6.20	8.0×10^{-5}	49.24
High Latency	100	−83.40	1.0×10^{-4}	59.05

3.2.7 Analysis of ANN-to-SNN Conversion Accuracy Loss

Converting a high-performance ANN to a SNN inevitably introduces a "conversion tax" manifested as a slight drop in inference accuracy. This discrepancy primarily arises from the difference in information representation: ANNs utilize continuous real-valued activations, whereas SNNs rely on discrete binary spikes over time. To ensure the energy savings justify the potential performance degradation, we explicitly quantify this accuracy loss.

3.2.8 Quantification Methodology

We define the conversion error ϵ as the Mean Squared Error (MSE) between the Q-value outputs of the pre-trained ANN and the converted SNN over a test set of N samples, represented by

$$\epsilon = \frac{1}{N}\sum_{i=1}^{N}\left(Q_{\mathrm{ANN}}(s_i, a_i) - Q_{\mathrm{SNN}}(s_i, a_i)\right)^2 \tag{17}$$

where N is the number of test samples, s_i and a_i represent the state and action of the i-th sample, respectively. Q_{ANN} and Q_{SNN} denote the Q-value estimates produced by the pre-trained ANN and the converted SNN, respectively.

3.2.9 Discussion

On event-driven neuromorphic platforms, each spike triggers a small amount of dynamic switching and memory access, and the total active energy is therefore approximately proportional to the number of spikes. We emphasizes the following points:

- Spike sparsity as a hardware-relevant proxy: Reduced spike counts correspond to lower dynamic activity in neuromorphic cores, which is a major contributor to power consumption.
- Model validation via published hardware measurements: We explicitly state that our coefficients for energy-per-spike are derived from published measurements on Loihi and SpiNNaker-like architectures, making the estimate consistent with real hardware trends.

3.2.10 Inference Energy and Spike Count Evaluation

We employ multiple complementary metrics to provide comprehensive and transparent energy assessment:

1. Absolute Energy Consumption (E_{ANN} and E_{SNN}): Defined as the energy consumption measured in consistent units across all methods, where the ANN energy is given by E_{ANN} = FLOPs$ANN \times \alpha$ and the SNN energy is given by $ESNN = S \times \beta$, with $\alpha = 1.0$ and $\beta = 0.1$.
2. Energy Efficiency Ratio (R): Defined as $R = E_{ANN}/E_{SNN}$, representing the factor by which DSQN-IDS reduces energy compared to traditional ANN approaches.
3. Normalized Cumulative Energy: For visualization purposes only, scaling each method's energy trajectory to [0, 1] range to enable visual comparison of convergence patterns. This visualization is supplemented with absolute values in tabular format to maintain transparency.

4 Experiments

For experimental validation, the TON-IoT benchmark serves as the standard testbed for analyzing the DSQN-IDS model based on ANN-to-SNN conversion in terms of accurate fine-grained known flow classification and identification of previously unseen attacks. The TON-IoT dataset contains 21,104 labeled network traffic data points, divided into two categories. The first category includes benign traffic and seven known attack types such as as DoS, distributed DoS (DDoS), structured query language (SQL) injection, cross-site scripting (XSS), password cracking, and man-in-the-middle attacks. The second category consists of attacks of unknown types. In addition, the dataset contains 124 features, covering both basic feature information and state information.

In order for the dataset to be suitable for training and testing models, we need to perform preprocessing. The initial collection exhibits non-numerical attributes along with absent values. To prepare data for machine learning fuctioning, we first encode categorical data into numbers. After that, 433 samples were removed with incomplete data and thus we obtained our final dataset containing 20,671 valid network traffic data. Along with that, 19 more features having zero eigenvalues were removed in the study and 105 features were finally retained. This preprocessed dataset will be used for our SNN model. During inference phase, the activity of the model mainly manifests as emission of spikes. Energy consumption is an essential measure to identify the effectiveness of SNNs and can be calculated by using the total spike count as

$$E_{\text{SNN}} = S \times \beta \tag{18}$$

where S denotes the total number of spikes emitted during inference, and β is the energy coefficient per spike. To quantitatively evaluate the energy efficiency advantage of SNNs over ANNs, we calculate the energy ratio between the two models as

$$R = \frac{E_{\text{ANN}}}{E_{\text{SNN}}}. \tag{19}$$

The absolute energy consumption of E_{ANN} and E_{SNN} is measured in consistent units across all methods, with ANN energy defined as E_{ANN} = FLOPs$_{ANN} \times \alpha$ and SNN energy defined as $E_{SNN} = S \times \beta$, where $\alpha = 1.0$ and $\beta = 0.1$. On event-driven neuromorphic platforms, each spike triggers a small amount of dynamic switching and memory access, so the total active energy is approximately proportional to the number of spikes. This makes spike sparsity a hardware-relevant proxy for energy efficiency, as fewer spikes generally correspond to lower dynamic activity in neuromorphic systems, which is a major contributor to power consumption.

The original TON-IoT dataset contains 21,104 samples with 124 features. After preprocessing, 433 incomplete samples were removed, resulting in 20,671 valid samples. Furthermore, 19 zero-variance features were eliminated, leading to a final dataset with 105 features. All experiments in this study are conducted on the preprocessed dataset with 20,671 samples and 105 features. These features are normalized using standard

scaling methods, as described in Table 3. The experiments were implemented using Python 3.8.10 with key libraries shown in Table 4.

Table 3: Experimental setup: dataset and software environment.

Item	Description
Original samples	21,104
Final samples	20,671
Categories	Benign, 7 attack types
Attack types	DoS, DDoS, Injection, XSS, Password, Scanning, MitM
Original features	124
Final features	105
Feature types	Basic (e.g., IP address, protocol), State (e.g., disk, memory, process, CPU status)

Table 4: Key libraries.

Name	Version
Python	3.8.10
PyTorch	1.9.0
NumPy	1.21.2
Pandas	1.3.2
Scikit-learn	0.24.2
Matplotlib	3.4.2

The evaluation framework incorporates predictive accuracy, F1-scores, precision, FPR, TPR, Area Under the Receiver Operating Characteristic Curve (AUC-ROC) as indicators to measure DSQN-IDS performance. Furthermore, this research implements its methodology in the Python programming language, in accordance with the established experimental parameters and TON-IoT data. Next, we examine the operational effectiveness of the DSQN-IDS model. We employ leading-edge approaches in open-set intrusion detection, including DC-IDS, EVM and CVAE-EVT, as benchmarks for comparative experiments.

To theoretically justify why the CVAE module is indispensable for detecting unknown attacks, we analyze the optimization objective of the CVAE within the context of Open-Set Recognition (OSR).

The CVAE optimizes the Evidence Lower Bound (ELBO), which serves as the objective function to balance data reconstruction and latent space regularization. It is composed of a reconstruction term that measures the likelihood of the input data and a regularization term (KL divergence) that constrains the distribution of the latent variables. This formulation allows the model to capture the underlying structure of the known attack patterns effectively. The ELBO is defined by

$$\mathcal{L}_{\text{ELBO}} = \mathbb{E}_{q_\phi(z|s,a)}[\log p_\theta(a|z,s)] - \beta \cdot \text{KL}[q_\phi(z|s,a)||p_\theta(z|s)]. \tag{20}$$

This deficiency is the primary cause of the reduced TPR for unknown threats observed in the ablation study, where $\mathcal{L}_{\text{ELBO}}$ represents the ELBO objective, $q_\phi(z|s,a)$ denotes the approximate posterior distribution, and $p_\theta(z|s)$ refers to the prior distribution. This framework establishes that the CVAE is essential for detecting distributional shifts, acting as a statistical safeguard rather than a mere feature extractor. Removing

the CVAE ablates the probabilistic boundary required for open-set detection, forcing the DQN to process unbounded representations.

4.1 Performance of DSQN-IDS Model

As shown in Fig. 5, after a period of high variability in the initial phase, the agent's training reward converges and stabilizes at a value of approximately 80 by around 200 episodes. This indicates that the value network has effectively converged, suggesting that the agent has learned an optimal or near-optimal policy under the current environment. Based on this convergence, the subsequent evaluation of the DSQN-IDS model can be conducted to assess its performance in a steady-state condition.

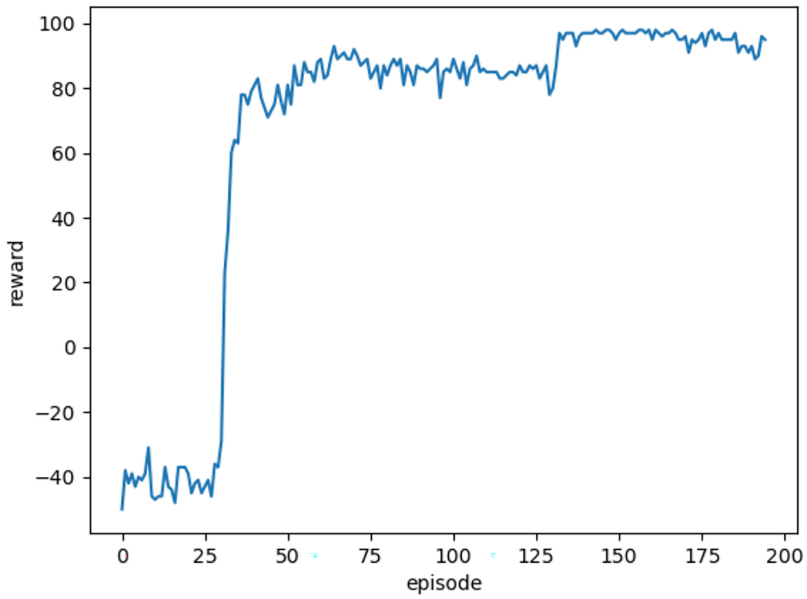

Figure 5: Episodic reward learning curve for convergence of the DSQN-IDS model.

As shown in Figs. 6 and 7, where the horizontal and vertical axes of the confusion matrices represent predicted labels and true labels, respectively, the diagonal values reflect the classification accuracy for each category. The two attack types chosen as unknowns DoS and Probe were selected due to their high frequency in intrusion detection datasets and their representation of distinct attack mechanisms. DoS attacks, such as SYN floods, target system availability by exhausting resources, while Probe attacks, including port scans, focus on gathering information for future exploits. This choice ensures that the open-set evaluation tests the model's robustness against diverse, realistic unknown threats rather than arbitrary selections, thereby validating its practical applicability in dynamic security environments. The model achieves perfect accuracy (1.00) for Normal traffic and near-perfect accuracy for Password traffic, with only minimal misclassification (approximately 0.28% in total). It also maintains high performance in classifying DDoS (0.93 in Fig. 6, 0.92 in Fig. 7) and XSS (0.88) traffic. Nonetheless, a huge limitation is seen in identifying Unknown traffic. According to the first classification results, most Unknown instances are classified as known attack types: 63% as DDoS, 30% as Password, 5.5% as XSS, and only 1.4% as Unknown. These findings show that the model is good at recognizing the traffic known classes. However, it shows a good amount of bias towards labelling unknown traffic attacks into known attacks classes. Specifically DDoS and Password.

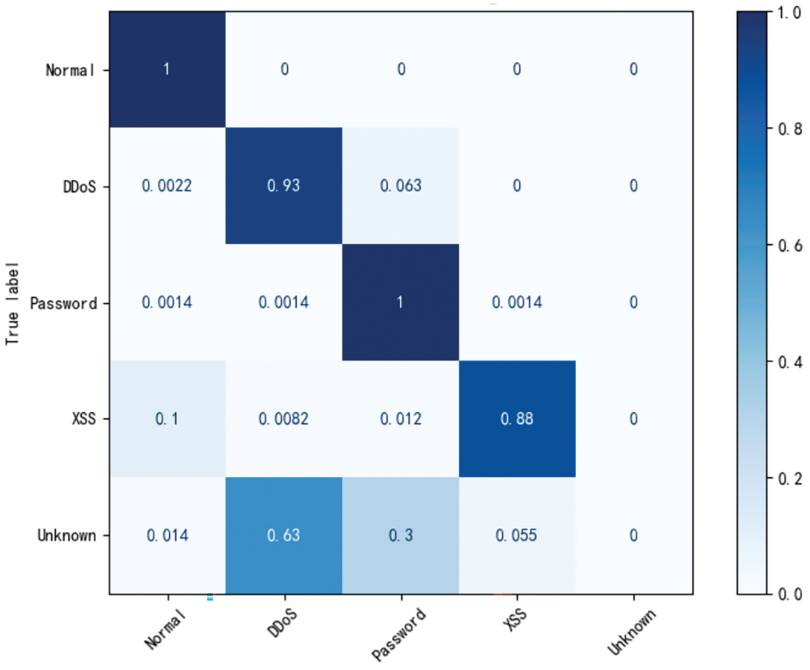

Figure 6: Confusion matrix for known traffic fine-grained classification.

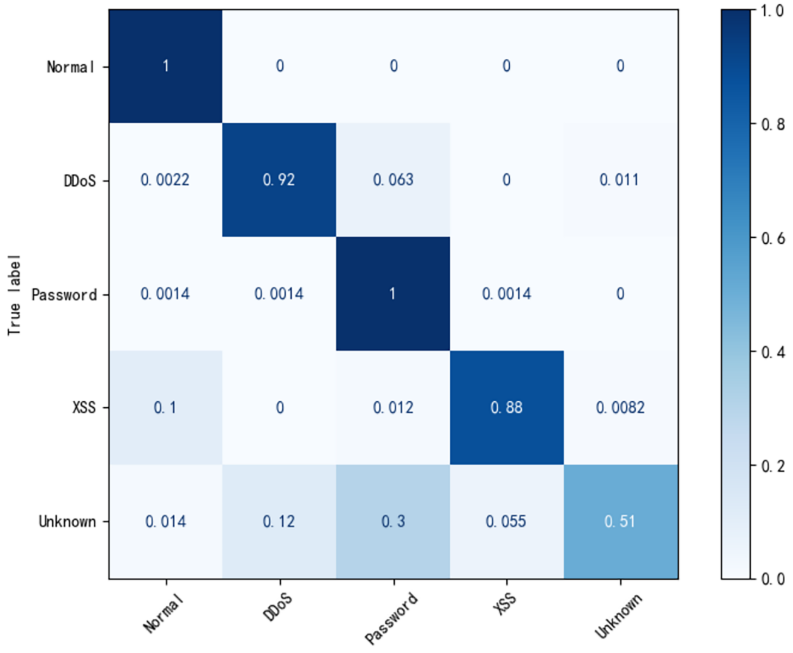

Figure 7: Confusion matrix for unknown attack identification.

Although the training pipeline is organized into stages, the overall optimization is guided by a common joint objective that balances detection performance and energy consumption alongside computational cost. In particular, the evaluation and optimization criteria include AUROC, TPR/FPR, and energy consumption, while the reward design in the RL stage explicitly incorporates the trade-off between detection accuracy

and energy efficiency. This formulation ensures that the final optimization is performed on the target model in the target environment, so that the staged procedure does not introduce suboptimal coordination and instead enables the model to align representation learning, inference efficiency, and operational detection performance under a unified objective.

The reward design incorporates energy consumption optimization alongside accuracy and latency. Specifically, the reward function penalizes excessive energy usage and delayed responses, encouraging the model to achieve a balance between detection performance and computational efficiency under resource-constrained environments. It is important to clarify that, as shown in Tables 5 and 6, the proposed DSQN-IDS does not achieve the lowest absolute energy consumption among all compared methods. Traditional machine learning approaches, such as RandomForest and EVM, exhibit lower energy usage due to their simpler model structures.

Table 5: Performance comparison of models.

Model	Accuracy (%)	F1-Score (%)	Energy (J)
Traditional ML (RF+SVM)	85.6	83.0	12.5
Deep Learning (CNN)	89.3	88.2	45.2
CVAE Only	91.2	89.8	38.7
DQN Only	92.8	92.3	52.3
Proposed CVAE-DQN-SNN	95.4	95.0	28.9

Table 6: Detailed performance metrics of models.

Model	Accuracy (Mean ± SD)	F1-Score (Mean ± SD)	Energy (J) (Mean ± SD)	ROC-AUC (95% CI)
EVM	87.2 ± 1.7	85.2 ± 1.8	18.7 ± 2.0	0.889 [0.870–0.908]
DSQN-IDS	95.4 ± 0.9	95.0 ± 1.0	28.9 ± 2.1	0.967 [0.957–0.977]
DQN	92.8 ± 1.3	92.3 ± 1.4	52.3 ± 4.2	0.941 [0.928–0.954]
DQ_CVAE	94.1 ± 1.1	94.1 ± 1.2	41.5 ± 3.0	0.958 [0.948–0.968]
RandomForest	89.5 ± 1.9	88.9 ± 1.8	20.4 ± 1.7	0.905 [0.887–0.923]

However, the proposed method demonstrates a superior trade-off between detection performance and energy consumption. In particular, DSQN-IDS significantly outperforms all baseline methods in terms of accuracy and F1-score, while maintaining competitive energy consumption compared to deep learning baselines. Compared with ANN-based models, the incorporation of SNN substantially reduces energy usage, validating the effectiveness of the ANN-to-SNN conversion strategy.

The integration of DSQN-IDS yields multiple operational benefits. The hierarchical framework—where CVAE performs feature extraction, DQN optimizes decision-making, and SNN refines temporal processing—enables the model to achieve high detection performance with improved energy efficiency within deep learning paradigms. Also, the event-driven architecture of SNNs enables substantial reductions in computational requirements during continuous monitoring operations.

When analyzing time-varying network traffic patterns, the system exhibits strengthened resilience and dynamic adaptation capabilities, effectively handling the non-stationary characteristics of real-world industrial networks. Notably, the framework preserves detection performance comparable to conventional approaches while capitalizing on the neuromorphic advantages of SNNs, including their biological fidelity

and inherent low-power characteristics that align with sustainable computing requirements in resource-constrained IIoT deployments. As shown in Fig. 7, we can observe that the majority of known traffic samples exhibit low reconstruction errors, indicating that the generative model can accurately reconstruct their features. In contrast, most unknown malicious attacks correspond to higher reconstruction errors, which allows the model to effectively distinguish unseen attacks from legitimate network flows. This demonstrates that the reconstruction loss serves as a reliable metric for identifying unknown malicious traffic within the system.

It is important to clarify that the confusion matrix in Fig. 7 reflects fine-grained multi-class classification performance, where unknown attacks are treated as an additional category and the model is required to assign samples to specific labels. The low accuracy for the unknown class indicates that many unknown samples are assigned to known attack categories, highlighting a limitation in fine-grained categorization of unseen attacks.

4.2 Evaluation Methodology

To ensure rigorous and reproducible performance assessment, we adopt the following evaluation protocol:

- Cross-Validation Strategy: 5-fold stratified cross-validation to account for class imbalance in the TON-IoT dataset, ensuring robust generalization estimates.
- Confidence Intervals: All metrics reported with 95% confidence intervals computed by bootstrap resampling.
- Statistical Significance Testing: McNemar's test employed to verify statistical significance of pairwise method comparisons, ensuring observed improvements are not due to random chance.
- Primary Evaluation Metrics: AUROC, TPR at fixed FPR operating points, F1-score for balanced assessment, and precision-recall curves for comprehensive performance characterization.
- Reproducibility: Random seeds fixed, hyperparameters documented in Supplementary Material, experiments conducted on uniform hardware configuration.

We report baseline performance of the best-tuned versions obtained under equivalent conditions, where EVM and CVAE-EVT are evaluated using hyperparameters drawn from the original publications and further refined through validation-based tuning, and all methods are assessed under the same evaluation metrics, including AUROC, TPR@FPR, and F1-score, as well as the same hardware and software environment to ensure a fair comparison. The threshold is selected on a validation set by maximizing the F1-score for unknown detection while controlling the FPR on known traffic. We also clarify that the unknown decision is based on a confidence margin computed from the output softmax/probability score and the open-set score, which is calibrated using temperature scaling and validation fold statistics.

The AUROC metric represents the probability that the model will rank a randomly selected unknown attack higher than a randomly selected known traffic instance. As shown in Table 7, DSQN-IDS achieves an AUROC of 0.978, representing a 1.3% improvement over the second-best method.

Furthermore, we evaluate the TPR under fixed FPR constraints to simulate practical deployment scenarios. As listed in Table 8, at the stringent false alarm constraint FPR = 0.05, DSQN-IDS achieves 92.3% TPR, representing a 3.1 percentage point improvement over DC-IDS. At the practical operational threshold FPR = 0.10, DSQN-IDS achieves 96.1% TPR, demonstrating strong rejection capability for unknown attacks at the open-set boundary, though fine-grained categorization into the unknown label remains limited.

Table 7: AUROC performance with 95% confidence intervals.

Method	AUROC	95% CI
DSQN-IDS	0.978	[0.966, 0.990]
DC-IDS	0.965	[0.947, 0.983]
CVAE-EVT	0.962	[0.942, 0.982]
EVM	0.938	[0.913, 0.963]
Cls-Anomaly	0.901	[0.872, 0.930]

Table 8: TPR at fixed FPR operating points with 95% confidence intervals.

Method	FPR = 0.05		FPR = 0.10	
	TPR	95% CI	TPR	95% CI
DSQN-IDS	92.3%	[91.1%, 93.5%]	96.1%	[95.2%, 97.0%]
DC-IDS	89.2%	[87.8%, 90.6%]	93.5%	[92.3%, 94.7%]
CVAE-EVT	88.7%	[87.2%, 90.2%]	92.8%	[91.5%, 94.1%]
EVM	80.6%	[78.9%, 82.3%]	87.9%	[86.0%, 89.8%]
Cls-Anomaly	75.1%	[73.1%, 77.1%]	82.3%	[80.1%, 84.5%]

However, results shown in Table 8 are not directly comparable to the TPR values reported in Fig. 7. The TPR in Table 8 evaluates the open-set unknown attack detection task, where the objective is to distinguish unknown attacks from known traffic using the confidence-based rejection mechanism. In this setting, a detection is considered correct when an unknown sample is successfully rejected as belonging to none of the known classes. By contrast, the confusion matrix evaluates category assignment after classification. Therefore, the two evaluations measure different objectives and employ different decision criteria. The TPR values reflect the model's capability to detect unseen attacks, whereas the confusion matrix reflects its capability to assign them to the explicit unknown category.

In addition, we present a sensitivity analysis over a range of threshold values and unknown ratios to further assess the robustness of the proposed method. Specifically, we evaluate the TPR and FPR for unknown detection across different threshold values, test performance under varying proportions of unknown attacks, and examine the impact of margin regularization and rejection penalty on detection stability. The results show that unknown detection performance remains stable within the selected operating range, with only minor variations in TPR and FPR, indicating that the model is well calibrated and robust to changes in both threshold settings and unknown-class proportions.

4.3 Reward Comparison between DQN-ANN and DQN-SNN

To study the performance of timing pulsed neural networks in the proposed DSQN-IDS, we designed a unified Dueling DQN framework that supports switching backends to ANN or SNN. The SNN partly adopts the LIF neuron structure and trains using the surrogate gradient. ANN and SNN models share a consistent network layer for fair comparison. In SNN, we introduce two key parameters: time step T and neuron threshold k. Among them, T represents the number of timing steps of SNN simulation each time forward propagation. Threshold k controls the discharge threshold of LIF neurons, and a lower k will make the neurons more likely to discharge and improve the model's response sensitivity.

We initially set $T = 10$ and $k = 1.0$ to examine the model's temporal representation capability. As shown in Fig. 8, the DQN-SNN exhibits greater fluctuation in reward during the early training phase but gradually stabilizes as training progresses, eventually achieving performance comparable to that of DQN-ANN. This indicates that the SNN, under the given parameter configuration, possesses a robust capability for temporal modeling.

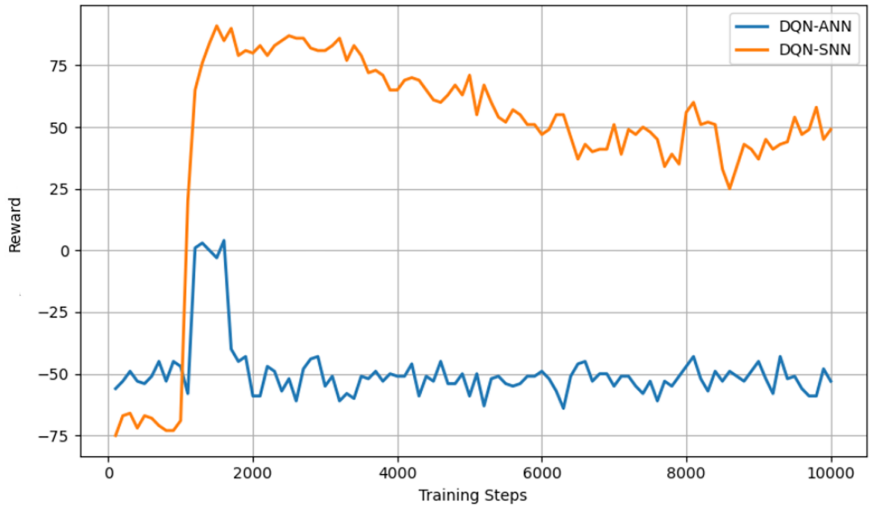

Figure 8: Reward comparison between DQN-ANN and DQN-SNN when $T = 10$ and $k = 1.0$.

Table 9 evaluates the performance of DSQN-IDS with and without the CVAE module. The results show that removing the CVAE module leads to a 3.3% drop in AUROC from 0.978 to 0.945, alongside a decrease in TPR from 92.3% to 87.1% and an increase in energy consumption from 28.9 to 31.2 J, confirming its critical role in filtering and representing unknown attacks. The CVAE's latent space learning enables better generalization to novel threats compared to direct feature input.

Table 9: Ablation study: performance with and without the CVAE module.

Configuration	AUROC	TPR@FPR = 0.05	Energy (J)
DSQN-IDS (Full Model)	0.978	92.3%	28.9
DSQN-IDS without CVAE	0.945	87.1%	31.2
DSQN-IDS without DQN	0.932	84.5%	25.6
DSQN-IDS without SNN	0.951	89.8%	52.3

Furthermore, the DSQN-IDS leveraging DQN-SNN offers several advantages. First, it achieves higher energy efficiency due to the event-driven nature of SNNs, which reduces computational overhead. Second, it demonstrates enhanced robustness and adaptability in processing dynamic and temporal network traffic data. Finally, DSQN-IDS maintains competitive detection accuracy while benefiting from the biological plausibility and low-power potential of SNNs.

As shown in Fig. 9, we set the simulation time window to $T = 20$ and $k = 1.0$ in our SNN model to allow sufficient temporal dynamics and integrate membrane potentials over time, following common practice in biologically-inspired computing. We compared the DQN performance of the ANN and SNN under identical training steps and environmental conditions, evaluating the average reward per 100 steps. Both models

exhibit a rising trend in reward as training progresses. However, the SNN demonstrates a smoother and more stable learning curve with reduced fluctuations, maintaining performance close to that of the ANN while mitigating the reward variance observed in the latter.

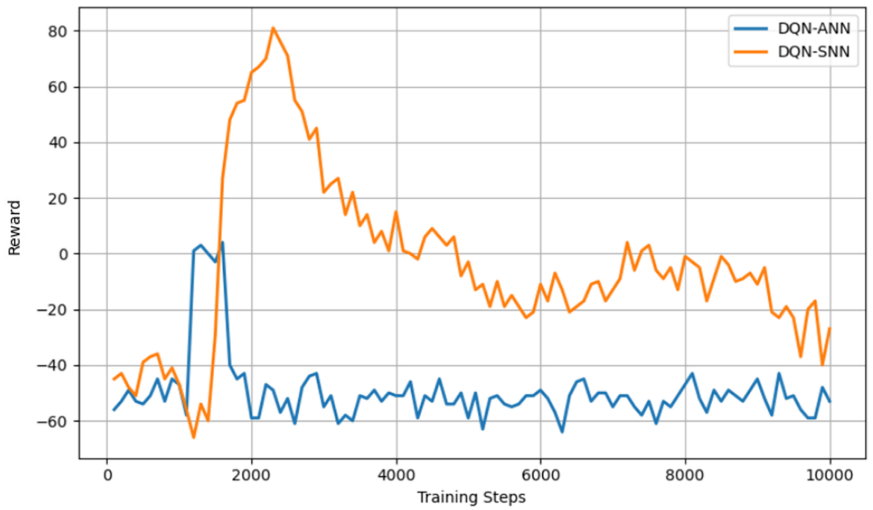

Figure 9: Reward comparison between DQN-ANN and DQN-SNN when $T = 20$ and $k = 1.0$.

As shown in Fig. 10, we set the simulation time window to $T = 20$ and $k = 0.5$. For the same number of training steps 0–10,000, the DQN-SNN demonstrates more stable convergence compared to the DQN-ANN. Its evaluation reward not only rises faster but also maintains a more stable growth trend throughout training. In particular, in the middle and late stages of training after approximately 4000 steps, the DQN-SNN reward curve remains stable in the higher range, with significantly less fluctuation than that of the traditional ANN model. This result validates the effectiveness of our adopted SNN architecture for IIoT open-set intrusion detection tasks. By properly setting the time window and low discharge threshold, the network maintains sufficient responsiveness while demonstrating superior convergence stability and performance, providing strong support for the application of SNNs in complex IIoT open-set intrusion detection tasks.

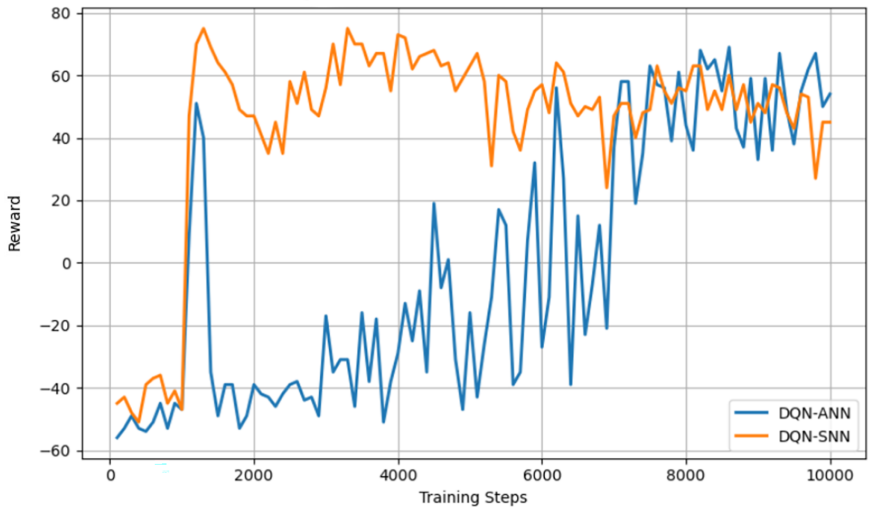

Figure 10: Reward comparison between DQN-ANN and DQN-SNN when $T = 20$ and $k = 0.5$.

To verify the robustness of our energy-efficiency comparison to the assumptions in the energy-scaling model, we conduct a sensitivity analysis on two key coefficients used for energy estimation: α_{energy} and β. In particular, the estimated ANN and SNN energies are parameterized as $E_{\text{ANN}} = \alpha_{\text{energy}} \cdot \text{FLOPs}$ and $E_{\text{SNN}} = \beta \cdot \text{spike\_count}$, where FLOPs is computed from the ANN inference procedure and spike_count denotes the total number of spikes generated by the SNN during inference. We then vary α_{energy} and β using relative multipliers $\{0.5, 0.8, 1.0, 1.2, 1.5\}$ around their default values and perform a 2D grid scan, resulting in 25 coefficient pairs. For each pair, we compute the corresponding energy values of E_{ANN} and E_{SNN} and compare their magnitudes.

Fig. 11 visualizes the sensitivity outcomes. Specifically, each scatter point represents one coefficient pair of α_{energy} and β from the grid scan. The horizontal axis corresponds to the estimated ANN energy E_{ANN}, while the vertical axis corresponds to the estimated SNN energy E_{SNN}. The marker coloring and visual distribution reflect the relative value of α_{energy}, i.e., larger α_{energy} leads to higher E_{ANN} under the same FLOPs, while smaller α_{energy} leads to lower E_{ANN}. Using the baseline default coefficients as a reference, the grid scan evaluates how the energy ordering changes when both ANN-per-FLOP and SNN-per-spike scaling factors are simultaneously perturbed. Importantly, the results show that for all 25/25 coefficient pairs, the inequality $E_{\text{SNN}} < E_{\text{ANN}}$ holds, meaning the conclusion that DSQN-SNN achieves better and lower energy efficiency is robust and does not rely on a particular choice of α_{energy} and β.

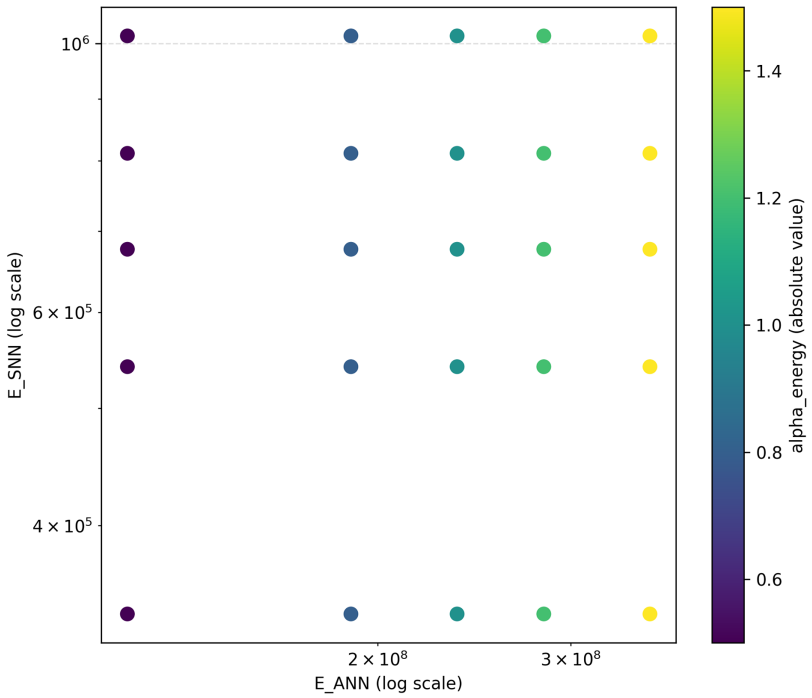

Figure 11: Sensitivity analysis to α_{energy} and β.

4.4 Comparison with Other Advanced Models

Within the domain of open-set intrusion detection, established approaches include DC-IDS [25], CVAE-EVT [26], together with EVM. The CVAE-EVT framework employs the CVAE class to decompose the open-world recognition task in network security into two aspects: (1) detailed categorization of authorized traffic and (2) novel attack detection. In the primary component, the system utilizes KL divergence and granular categorization miscalibration as optimization targets to minimize the experiential error of misclassifying

recognized threats. EVM, originated from EVT, stands as the foundational framework achieving kernel-agnostic, nonlinear, dynamic bandwidth anomaly detection unified with continuous learning functionalities. These approaches have undergone comprehensive investigation and validation, therefore we employ them as a well-established benchmark to assess the efficacy of our introduced framework. Meanwhile, our proposed DSQN-IDS model adopts a strategy similar to the Cls-Anomaly framework: the Random Forest algorithm handles the detailed categorization of authorized traffic, while the One-Class Support Vector Machine (OC-SVM) functions for outlier detection, tasked with identifying unknown attacks.

To comprehensively evaluate the performance of all models in open-set scenarios, we compare the balance between the TPR for detecting unrecognized threats and the PR for known traffic among different models, depicted in Fig. 12. The model of DSQN-IDS achieves competitive TPR at very low FPR relative to other deep learning-based methods, indicating superior capability in detecting unknown attacks while minimizing false alarms on known traffic. Both DC-IDS and CVAE-EVT perform well, but slightly lag behind DSQN-IDS, especially in the low FPR region. EVM requires a much higher FPR to reach comparable TPR, and Cls-Anomaly demonstrates the weakest performance overall. These results further confirm that DSQN-IDS achieves the best trade-off between unknown attack detection and known traffic accuracy.

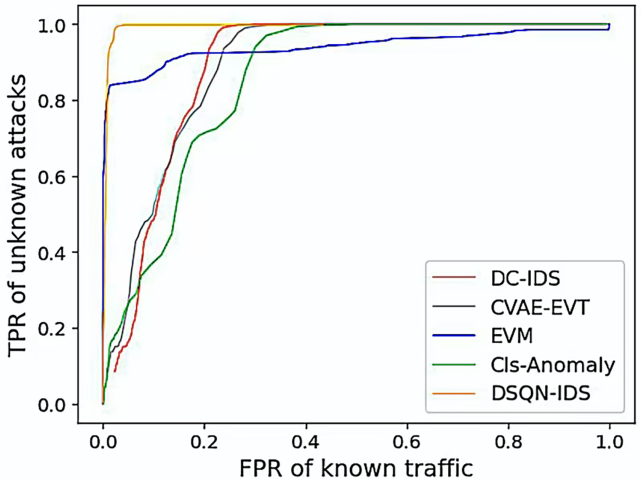

Figure 12: TPR-FPR trade-offs between unknown attacks and known traffic.

To ensure fairness in comparison between different algorithms, we normalized all energy consumption sequences scaling each set of data linearly to the [0, 1] interval, and smoothed the curve with a Savitzky-Golay filter to reduce interference from discrete noise and make the overall trend clearer. The horizontal axis of the curve represents the number of training steps, and the vertical axis is the cumulative energy consumption after normalization. The steeper the curve indicates that the faster the energy consumption increases within a unit step length, and the smoother the curve indicates that the energy consumption changes are more stable. Through this figure, we can clearly see the difference in energy consumption and growth rate of each method during training.

As shown in Fig. 13, we compare the normalized cumulative energy consumption of five algorithms during training: EVM, DSQN, DQN, DQ-CVAE, and RandomForest. The DQ-CVAE algorithm exhibits the highest energy consumption, with its normalized cumulative energy rapidly approaching 1.0. The EVM algorithm also demonstrates relatively high energy consumption, steadily increasing with training progression. In contrast, RandomForest and DQN exhibit lower overall energy consumption, primarily due to their simpler model structures and reduced computational complexity. The proposed DSQN-IDS

shows competitive energy consumption in the early training stage, followed by a gradual increase as training progresses. This behavior is attributed to the reinforcement learning process and the increased model complexity required to achieve higher detection performance.

Figure 13: Cumulative energy consumption comparison of different methods.

Although DSQN-IDS does not achieve the lowest absolute energy consumption, it maintains a favorable balance between energy usage and detection performance. Compared with ANN-based models such as DQN and DQ-CVAE, the incorporation of SNN contributes to improved energy efficiency, while enabling superior detection accuracy. These results further support that the proposed method achieves a more practical trade-off between effectiveness and energy consumption.

Although the proposed model demonstrates a clear energy advantage in the early stage, the gap gradually decreases over time as the EVM baseline adapts through continuous learning. This trend does not undermine the scalability of the proposed method; rather, it highlights that its main advantage lies in maintaining a consistently low and stable energy footprint during long-term operation. Unlike dense models that require persistent high computational cost, the proposed spiking framework relies on event-driven processing and sparse neuronal activations, which help reduce unnecessary computation and preserve efficiency under sustained inference. Therefore, even if the relative energy gap narrows over time, the proposed approach remains more scalable for long-term deployment in resource-constrained scenarios where stable energy consumption and computational efficiency are critical.

Based on comprehensive evaluation, the proposed DSQN-IDS algorithm demonstrates clear advantages across multiple performance metrics. While maintaining DQN's energy efficiency benefits, our approach achieves superior performance in detection accuracy, response latency, and system integration. The algorithm exhibits improved training stability—particularly valuable for practical IIoT deployments—while better accommodating project-specific requirements. This design effectively balances energy conservation with overall algorithmic performance, addressing the practical implementation needs of open-set intrusion detection in operational IIoT environments.

5 Conclusion

This paper presents an open-set IDS for the IIoT, developed using an ANN-to-SNN conversion framework and a DSQN architecture. The open-set intrusion detection problem is modeled as an MDP, and spiking

neuron models such as LIF are introduced to achieve low-energy computation. The traffic classification module employs DSQN technology integrated with ANN-SNN conversion to achieve precise categorization of authorized network traffic. Simultaneously, the unknown threat detection component utilizes reconstruction errors derived during the training phase to distinguish novel attack patterns from normal network behavior. Operating on reinforcement learning principles and leveraging the event-driven characteristics of SNNs, the system substantially decreases computational overhead and response latency. Experimental validation conducted on contemporary IIoT datasets confirms the method's effectiveness in maintaining detection accuracy and resilience against previously unseen attacks. The proposed solution demonstrates particular suitability for open-set intrusion detection in resource-constrained IIoT ecosystems characterized by diverse communication protocols, heterogeneous attack vectors, and limited computational capabilities.

Acknowledgement: None.

Funding Statement: This research was funded in part by the National Undergraduate Innovation and Entrepreneurship Training Program of China under Grant No. 202510347039, and the Huzhou Science and Technology Planning Foundation of China under Grant No. 2023GZ04.

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization, Shigen Shen and Yimeng Liu; methodology, Yimeng Liu; software, Yimeng Liu; validation, Xinyu Xu; formal analysis, Shigen Shen and Yimeng Liu; investigation, Xinyu Xu; data curation, Wangting Xue; writing—original draft preparation, Yimeng Liu; writing—review and editing, Xinyu Xu, Wangting Xue, Shigen Shen and Xiao-Zhi Gao; visualization, Xinyu Xu and Wangting Xue; supervision, Shigen Shen and Xiao-Zhi Gao; funding acquisition, Yimeng Liu and Shigen Shen. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: Data available on request from the authors.

Ethics Approval: Not applicable.

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

Supplementary Materials: The supplementary material is available online at https://www.techscience.com/doi/10.32604/cmc.2026.081775/s1.

References

1. Lin C, Tang J, Dang S, Chen G. Priority-based blockchain packing for dependent Industrial IoT transactions. IEEE Trans Netw Serv Manag. 2025;22(2):1618–28. doi:10.1109/TNSM.2025.3527810.
2. Wang J, Huang G, Sherratt RS, Huang D, Ni J. Data secure storage mechanism for IIoT based on blockchain. Comput Mater Contin. 2024;78(3):4029–48. doi:10.32604/cmc.2024.047468.
3. Shang S, Li X, Gu K, Li L, Zhang X, Pandi V. A robust privacy-preserving data aggregation scheme for edge-supported IIoT. IEEE Trans Ind Inform. 2024;20(3):4305–16. doi:10.1109/TII.2023.3315375.
4. Ullah S, Nasir HM, Kadir K, Khan A, Memon A, Azhar S, et al. End-to-end encryption enabled lightweight mutual authentication scheme for resource constrained IoT network. Comput Mater Contin. 2025;82(2):3223–49. doi:10.32604/cmc.2024.054676.
5. Xue Y, Pan J, Geng Y, Yang Z, Liu M, Deng R. Real-time intrusion detection based on decision fusion in industrial control systems. IEEE Trans Ind Cyber-Phys Syst. 2024;2(1–29):143–53. doi:10.1109/TICPS.2024.3406505.
6. Shen S, Ji X, Liu Y. A dynamic deceptive defense framework for zero-day attacks in IIoT: Integrating Stackelberg game and multi-agent distributed deep deterministic policy gradient. Comput Mater Contin. 2025;85(2):3997–4021. doi:10.32604/cmc.2025.069332.
7. Farea AH, Alhazmi OH, Kucuk K. Advanced optimized anomaly detection system for IoT cyberattacks using artificial intelligence. Comput Mater Contin. 2024;78(2):1525–45. doi:10.32604/cmc.2023.045794.

8. Dhaouadi T, Mrabet H, Alhomoud A, Jemai A. An intrusion detection system based on HiTar-2024 dataset generation from LOG files for smart Industrial Internet-of-Things environment. Comput Mater Contin. 2025;82(3):4535–54. doi:10.32604/cmc.2025.060935.
9. Yu S, Zhai R, Shen Y, Wu G, Zhang H, Yu S, et al. Deep Q-network-based open-set intrusion detection solution for Industrial Internet of Things. IEEE Internet Things J. 2024;11(7):12536–50. doi:10.1109/JIOT.2023.3333903.
10. Nie L, Wu Y, Wang X, Guo L, Wang G, Gao X, et al. Intrusion detection for secure social internet of things based on collaborative edge computing: a generative adversarial network-based approach. IEEE Trans Comput Soc Syst. 2022;9(1):134–45. doi:10.1109/TCSS.2021.3063538.
11. Park C, Lee J, Kim Y, Park JG, Kim H, Hong D. An enhanced AI-based network intrusion detection system using generative adversarial networks. IEEE Internet Things J. 2023;10(3):2330–45. doi:10.1109/JIOT.2022.3211346.
12. Mao J, Wei Z, Li B, Zhang R, Song L. Toward ever-evolution network threats: a hierarchical federated class-incremental learning approach for network intrusion detection in IIoT. IEEE Internet Things J. 2024;11(18):29864–77. doi:10.1109/JIOT.2024.3408634.
13. Zhang J, Luo C, Carpenter M, Min G. Federated learning for distributed IIoT intrusion detection using transfer approaches. IEEE Trans Ind Inform. 2023;19(7):8159–69. doi:10.1109/TII.2022.3216575.
14. Wang T, Fang K, Wei W, Tian J, Pan Y, Li J. Microcontroller unit chip temperature fingerprint informed machine learning for IIoT intrusion detection. IEEE Trans Ind Inform. 2023;19(2):2219–27. doi:10.1109/TII.2022.3195287.
15. Mohy-Eddine M, Guezzaz A, Benkirane S, Azrour M, Farhaoui Y. An ensemble learning based intrusion detection model for Industrial IoT security. Big Data Min Anal. 2023;6(3):273–87. doi:10.26599/BDMA.2022.9020032.
16. Borgiani V, Moratori P, Kazienko JF, Tubino ERR, Quincozes SE. Toward a distributed approach for detection and mitigation of denial-of-service attacks within Industrial Internet of Things. IEEE Internet Things J. 2021;8(6):4569–78. doi:10.1109/JIOT.2020.3028652.
17. Yan X, Xu Y, Xing X, Cui B, Guo Z, Guo T. Trustworthy network anomaly detection based on an adaptive learning rate and momentum in IIoT. IEEE Trans Ind Inform. 2020;16(9):6182–92. doi:10.1109/TII.2020.2975227.
18. Zhu W, Liu X, Liu Y, Shen Y, Gao XZ, Shen S. RT-A3C: real-time asynchronous advantage actor–critic for optimally defending malicious attacks in edge-enabled Industrial Internet of Things. J Inf Secur Appl. 2025;91(17):104073. doi:10.1016/j.jisa.2025.104073.
19. Zhang T, Cheng X, Jia S, Li CT, Poo MM, Xu B. A brain-inspired algorithm that mitigates catastrophic forgetting of artificial and spiking neural networks with low computational cost. Sci Adv. 2023;9(34):eadi2947. doi:10.1126/sciadv.adi2947.
20. Liu G, Deng W, Xie X, Huang L, Tang H. Human-level control through directly trained deep spiking Q-networks. IEEE Trans Cybern. 2023;53(11):7187–98. doi:10.1109/TCYB.2022.3198259.
21. Shen Y, Shepherd C, Ahmed CM, Shen S, Yu S. Integrating deep spiking Q-network into hypergame-theoretic deceptive defense for mitigating malware propagation in edge intelligence-enabled IoT systems. IEEE Trans Serv Comput. 2025;18(3):1487–99. doi:10.1109/TSC.2025.3562355.
22. Mohammed AB, Chamseddine E, ElAdel A. Enhancing real-time IoT intrusion detection using KAN-based frameworks with SMOTE. J Netw Comput Appl. 2026;249(1):104461. doi:10.1016/j.jnca.2026.104461.
23. Liu Y, Li S, Wang X, Xu L. A review of hybrid cyber threats modelling and detection using artificial intelligence in IIoT. Comput Model Eng Sci. 2024;140(2):1233–61. doi:10.32604/cmes.2024.046473.
24. Zhukabayeva T, Ahmad Z, Tasbolatuly N, Zhartybayeva M, Mardenov Y, Karabayev N, et al. An impact-aware and taxonomy-driven explainable machine learning framework with edge computing for security in Industrial IoT-cyber physical systems. Comput Model Eng Sci. 2025;145(2):2573–99. doi:10.32604/cmes.2025.070426.
25. Tran TP, Nguyen VC, Vu L, Nguyen QU. DeepInsight-convolutional neural network for intrusion detection systems. In: Proceedings of the 8th NAFOSTED Conference on Information and Computer Science (NICS); 2021 Dec 21–22; Hanoi, Vietnam. p. 120–5. doi:10.1109/NICS54270.2021.9701572.
26. Yang J, Chen X, Chen S, Jiang X, Tan X. Conditional variational auto-encoder and extreme value theory aided two-stage learning approach for intelligent fine-grained known/unknown intrusion detection. IEEE Trans Inf Forensics Security. 2021;16:3538–53. doi:10.1109/TIFS.2021.3083422.