algorithms

MDPI

Article

APA3CID: An Intrusion Detection Algorithm Based on Feature Optimization and Asynchronous Actor-Critic Learning

Jiantao Cui [1], Huicong Yu [1,*], Jiahe Liu [1], Ruipeng Li [1], Wanwei Huang [1], Haiyan Sun [1] and Sunan Wang [2]

[1] Department of Software Engineering, Zhengzhou University of Light Industry, Zhengzhou 450007, China; cjeant@zzuli.edu.cn (J.C.); 332516060904@zzuli.edu.cn (J.L.); 542313330221@zzuli.edu.cn (R.L.); 2016044@zzuli.edu.cn (W.H.); sunhaiyan@zzuli.edu.cn (H.S.)

[2] School of Electronics and Communication Engineering, Shenzhen Polytechnic University, Shenzhen 518000, China; wangsunansps@163.com

* Correspondence: yuhc812@163.com

Abstract

As the Industrial Internet of Things becomes increasingly interconnected with critical infrastructure, intrusion traffic exhibits characteristics such as high-dimensional redundancy, class imbalance, and temporal correlation, posing challenges for detection systems in terms of feature representation, model complexity control, and real-time performance. To address the aforementioned issues, this paper proposes an intrusion detection algorithm based on feature optimization and asynchronous advantage actor-critic learning (APA3CID). First, the raw dataset was preprocessed using methods such as label encoding and normalization. Feature selection was performed using the improved Whale Optimization Algorithm (WOA) to reduce data redundancy and eliminate irrelevant features. The samples were then serialized based on the order in which they were collected. Second, we model the detection process as a Markov decision process, use a sliding window to construct states that capture recent temporal features, and, building upon the Asynchronous Advantage Actor-Critic (A3C) framework, we incorporate an adaptive exploration mechanism to address the issues of insufficient exploration in the early training phase and unstable convergence in the later phase. Additionally, we introduce an asynchronous lag correction strategy that utilizes truncated importance weights to mitigate the bias caused by policy lag in asynchronous parallel training, thereby enhancing the stability and robustness of policy updates. Finally, experimental results show that on the X-IIoTID dataset, APA3CID achieves a 3.51% increase in detection rate and a 4.26% increase in F1-score compared to the traditional A3C algorithm. On the WUSTL-IIoT-2021 dataset, single-sample prediction takes as little as 11.56 microseconds, with Acc, DR, and F1-score all exceeding 90%. This outperforms comparison models such as LR, XGBoost, CNN, and the baseline A3C, meeting the requirements of industrial IoT scenarios for low false-negative rates and high real-time performance.

Keywords: industrial internet of things; intrusion detection; feature selection; asynchronous actor-critic; adaptive exploration; asynchronous lag correction

Academic Editor: Carolina Del-Valle-Sot

Received: 11 April 2026
Revised: 15 May 2026
Accepted: 21 May 2026
Published: 23 May 2026

1. Introduction

With the rapid development of the Industrial Internet and smart manufacturing, the Industrial Internet of Things (IIoT) has been widely adopted in critical infrastructure sectors, including energy, power, transportation, and chemicals. The IIoT connects a vast array of devices—including sensors, controllers, and smart terminals—to the internet, profoundly

transforming traditional industrial production models and management practices. The IIoT enables real-time monitoring, precise control, and intelligent decision-making throughout the entire production process, significantly enhancing the production efficiency, operational flexibility, and resource utilization of industrial systems [1]. However, as a large number of industrial devices, control systems, and sensors become interconnected via the internet, the attack surface of IIoT systems has expanded significantly, making them more vulnerable to cyberattacks such as asset discovery, unauthorized access, and the injection of control commands. Malicious attacks on critical infrastructure in sectors such as electricity, energy, and manufacturing can lead to production disruptions and equipment damage, which in turn can trigger major safety incidents. Compared to security methods that rely primarily on static protection—such as firewalls, network segmentation, and encryption—the deployment of intrusion detection systems (IDS) [2] enables continuous monitoring of network traffic and system behavior, supporting the identification of anomalous behavior and attack incidents [3], and has become a critical technical component of the active defense framework for the IIoT.

Intrusion detection systems rely on sophisticated detection algorithms to identify and classify intrusion attacks. In traditional network environments, where attack patterns are relatively fixed and traffic behavior boundaries are well-defined, detection methods based on rule matching or static classification models can achieve relatively stable results [4]. However, as industrial environments become increasingly complex in IIoT scenarios, the types of intrusion attacks have also become more diverse. These include a heightened emphasis on the temporal sequence of attacks, significantly improved adversarial capabilities such as evading detection and obfuscating legitimate control commands, as well as dynamic changes in device status, operational load, and communication patterns. Consequently, traditional intrusion detection algorithms face challenges such as high detection latency, insufficient generalization capabilities, and high false negative rates [5].

As a key technical approach to industrial cybersecurity, intrusion detection has shifted in recent years from traditional methods based on rules and feature engineering to intelligent detection methods such as machine learning and deep learning. However, given the high-dimensional, heterogeneous data and stringent real-time requirements of the IIoT, existing intelligent intrusion detection systems face challenges such as high-dimensional feature processing, high computational complexity, and a lack of adaptive learning capabilities, making them unable to meet the real-time and high-accuracy requirements of industrial environments [6]. To address the aforementioned issues, this paper proposes an intrusion detection algorithm based on feature optimization and asynchronous advantage actor-critic with adaptive exploration and policy-lag correction (APA3CID), which aims to improve accuracy and real-time performance in industrial IoT intrusion detection while reducing false negative rates. The APA3CID algorithm first preprocesses the dataset through label encoding and normalization, then feeds the processed data into the feature selection module; next, it applies the GSLDWOA algorithm to perform feature selection on the dataset, eliminating redundant and irrelevant features to reduce the dimensionality of the data; Finally, an agent is constructed using A3C, modeling the dynamic industrial IoT environment as a Markov decision process. The policy network and value network are optimized through multi-step cumulative advantage and multi-asynchronous parallel training. An adaptive exploration mechanism is incorporated to enhance early-stage exploration and ensure stable convergence in the later stages of training. Additionally, an asynchronous lag correction strategy is introduced to mitigate policy lag biases caused by asynchronous updates, reduce gradient noise and update oscillations, enabling the agent to rapidly identify known attacks in the dynamic IIoT environment. This approach effectively

reduces both training and testing times while improving the model's detection rate and accuracy. The main contributions of this paper are as follows:

(1) To address the issues of data heterogeneity and high-dimensional redundancy in the IIoT, we have established a data processing workflow that combines preprocessing with GSLDWOA feature selection. By leveraging Gaussian variation and variable-step Lévy flights, we enhance the accuracy of feature screening, effectively filter out noisy features, and provide high-quality, low-dimensional data for subsequent modeling.

(2) To address the demands of complex industrial environments for model robustness and real-time performance, we have developed a modeling framework based on Markov processes. States are represented using sliding-window encoding to capture recent attack behaviors and traffic characteristics, while actions are mapped to specific attack subcategories to align with practical requirements. The reward function incorporates differentiated weights, and a discount factor is introduced to amplify the impact of long-term risks, thereby resolving the challenge that traditional models face in adapting to the dynamic nature of industrial scenarios.

(3) To improve training efficiency and policy stability, we designed an asynchronous parallel training mechanism for multiple subnetworks based on the A3C framework. By incorporating entropy-based annealing and uncertainty-driven exploration, we achieved thorough exploration in the early stages and stable convergence in the later stages. Additionally, we introduced an asynchronous lag correction strategy that utilizes truncated importance weights to mitigate policy lag bias caused by asynchronous updates, thereby reducing gradient noise and update oscillations and further enhancing convergence stability and detection performance.

(4) On the X-IIoTID and WUSTL-IIoT-2021 datasets, we conducted ablation experiments to demonstrate the synergistic advantages of each innovative module. We compared APA3CID with mainstream models such as LR, XGBoost, and DDPG to validate the model's significant advantages in key metrics such as accuracy and F1-score.

2. Related Work

In research on intrusion detection in the IIoT, network traffic features are typically characterized by high dimensionality, high redundancy, and complex correlations, which directly impact the training efficiency and detection performance of detection models [7]. Consequently, identifying effective feature selection methods to identify features closely associated with attack behavior, reducing the feature dimension, and eliminating redundancy—thereby providing high-quality, low-dimensional data for subsequent detection models—has become one of the key research areas in the field of intrusion detection. At the same time, as detection scenarios become increasingly dynamic and attack methods continue to grow more sophisticated, it is difficult to balance requirements such as high detection rates and low false-negative rates. Deep reinforcement learning-based detection methods, with their strong feature extraction capabilities and advantages in dynamic decision-making, have gradually become a hot topic in research.

2.1. Feature Selection in Intrusion Detection

Feature selection is a key prerequisite for the effectiveness of intrusion detection. Its core objective is to identify a subset of features highly correlated with attack behavior from high-dimensional network traffic data, thereby reducing redundant information and noise interference, which in turn lowers computational complexity and improves detection performance. Existing research has explored methods such as wrapper-based, filter-based, embedded, and intelligent feature selection optimization from various perspectives.

Early research focused primarily on traditional methods such as wrappers, filters, and embedding. Talpur et al. proposed a novel wrapper-based Sand Cat Swarm Optimization (SCSO) technique that combines swarm intelligence search mechanisms with wrapper-based feature selection to improve the global coverage of feature subset search [8]. Liu et al. proposed an embedded feature selection method using the weighted Gini index (WGI) to analyze the applicability of different statistical criteria in low-dimensional feature selection scenarios [9]. Emmanuel et al. proposed a method that combines feature selection with classification model design, using a stacked classification framework to enhance the overall discriminative power of the model following feature screening [10].

As behavioral data from industrial IoT devices has become increasingly high-dimensional, heterogeneous, redundant, and subject to growing noise interference, intelligent feature selection optimization algorithms have emerged as a research hotspot in recent years. Gong et al. proposed a novel feature selection evaluation algorithm, CONMI_FS, which performs joint modeling based on two aspects—the correlation between features and classes and the redundancy among features—to obtain a more discriminative subset of features [11]. Liu et al. proposed a hybrid lightweight IDS architecture that combines an embedded model (EM) for feature selection and a convolutional neural network (CNN) for attack detection and classification, leveraging deep learning models to achieve end-to-end attack detection [12]. Huang et al. proposed a network intrusion detection feature selection algorithm based on an improved binary swarm-based optimization (SABPIO) algorithm to enhance the stability and convergence efficiency of the search process [13].

As mentioned above, Talpur et al. [8] proposed the SCSO algorithm, which improves search comprehensiveness; however, it tends to converge slowly when applied to high-dimensional industrial traffic data, making it difficult to meet the requirements for efficient training of detection models. Gong et al. [11] proposed the CONMI_FS algorithm, which balances feature correlation and redundancy; however, during the screening of complex attack features, it is prone to overlooking important features strongly associated with attack chains, thereby affecting subsequent detection accuracy. To address these shortcomings, APA3CID introduces the GSLDWOA feature selection method. By enhancing global search capabilities, it efficiently explores potential high-quality feature combinations within a large solution space, thereby mitigating issues such as local optima and slow convergence. It maintains good stability in feature selection under imbalanced data conditions while minimizing the loss of critical attack features, effectively addressing the shortcomings of existing algorithms.

2.2. Applications of Deep Reinforcement Learning in Intrusion Detection

To address the challenges posed by high-dimensional features, complex attack patterns, and dynamic network environments in industrial settings, researchers have gradually incorporated deep learning and reinforcement learning methods, driving the evolution of intrusion detection technology toward greater intelligence. Existing research in this area can be broadly categorized into two main types: intrusion detection methods based on deep learning and those based on reinforcement learning.

(1) Deep learning-based intrusion detection algorithms

Mehedi et al. introduced the concept of deep transfer learning and enhanced the adaptability of IDS across different data distributions by constructing a ResNet-based transfer model [14]. Wazid et al. proposed a deep learning-based malware attack detection system (SDLMA-IITS), which integrates deep learning into IoT-enabled intelligent transportation systems, with a focus on detection performance in malware attack scenarios [15]. Attique et al. proposed a bidirectional long short-term memory (BiLSTM) model based on an adaptive attention mechanism to enhance the model's ability to focus on key temporal

features under limited data conditions [16]. Bella et al. proposed a Deep Neural Decision Forest-based Intrusion Detection System (DNDF-IDS), which combines deep neural networks with decision forests to simplify the model structure and improve detection performance through various feature selection strategies [17]. To address the issue of attacks on roadside units in IoV scenarios, Nie et al. proposed a deep learning architecture based on convolutional neural networks and designed a CNN-based deep architecture to characterize link traffic behavior [18]. Although the aforementioned deep learning methods perform well on static datasets, they rely on a large number of labeled samples, are difficult to train incrementally online, and have limited generalization capabilities against unknown attacks, making it difficult to meet the demands of rapidly evolving attack patterns in IIoT environments.

(2) Reinforcement learning-based intrusion detection algorithms

Liang et al. proposed an intrusion detection system for VANETs (GaDQN-IDS) based on Bayesian game theory and deep Q-learning networks, modeling the intrusion detection problem in VANETs as a dynamic game between the IDS and the attacker, and incorporating deep Q-learning to achieve adaptive optimization of detection strategies [19]. Duan et al. proposed a novel and robust multi-layer defense system (DosSink) that combines variational autoencoders with actor-critic reinforcement learning to achieve joint optimization of detection and mitigation strategies [20]. Yu et al. proposed an intrusion detection strategy based on stochastic games and deep reinforcement learning, characterizing the adversarial relationship between attackers and detectors in IIoT from the perspective of stochastic games, and introducing deep reinforcement learning to reduce system overhead [21]. Ren et al. proposed a Multi-Agent Feature Selection Intrusion Detection System (MAFSIDS), which integrates feature selection with the intrusion detection process and enhances detection performance through multi-agent collaborative learning [22]. Existing reinforcement learning-based IDSs primarily employ single-agent DQN or simple Actor-Critic frameworks, which are prone to policy oscillations in high-dimensional state spaces and lack the ability to model long-term returns for multi-step attack chains.

In summary, Ren et al. [22] proposed a multi-agent feature selection framework; however, its use of centralized training makes it difficult to scale to large-scale IIoT edge nodes. Meanwhile, Liang et al.'s [19] GaDQN-IDS relies on experience replay, which incurs high memory overhead on resource-constrained devices. In contrast, A3C employs multi-worker asynchronous updates and does not require experience replay, making it better suited for lightweight, low-latency deployments at the edge. By combining APA3CID with an improved A3C model, we perform sequential decision modeling for the attack detection process. This approach enhances training efficiency and real-time response capabilities while maintaining detection accuracy, thereby improving the model's usability and practicality in complex industrial IoT environments.

To further clarify the differences between this study and existing work, Table 1 presents a systematic comparative analysis of existing IIoT intrusion detection methods, examining key dimensions such as technical approach, core strengths, and major shortcomings.

Table 1. Comparative Analysis of Existing IIoT Intrusion Detection Methods.

Method Categories	Technical Approach	Key Strengths	Main Shortcomings	Research Gaps	Improvements to APA3CID
Traditional machine learning	Feature Engineering + Classifier	Highly interpretable and computationally efficient	Features rely on manual design and have poor generalization capabilities	Adaptive Feature Extraction	Federated Feature Learning + Automatic Feature Selection

Table 1. *Cont.*

Method Categories	Technical Approach	Key Strengths	Main Shortcomings	Research Gaps	Improvements to APA3CID
Deep Learning	End-to-End Feature Learning	Automatic feature extraction with high accuracy	Computationally intensive and lacking in interpretability	Lightweight deployment	Knowledge Distillation + Model Compression
Reinforcement Learning	Environmental Interaction + Strategy Optimization	Adaptive decision-making, handling unknown attacks	Low sample efficiency and training difficulties	Edge Deployment Optimization	Model Pruning + Quantization

3. Overall Algorithm Architecture

This paper proposes an intrusion detection algorithm based on feature optimization and an asynchronous actor-critic approach, aimed at addressing the issues of insufficient accuracy, lack of real-time performance, and high false negative rates in industrial IoT intrusion detection. The architectural structure of the APA3CID algorithm is shown in Figure 1. It consists of a data preprocessing module, a GSLDWOA feature selection module, and an APA3CID intrusion detection module. First, the dataset is preprocessed by converting non-numeric features into numerical data through label encoding and applying normalization to eliminate differences in feature scales; subsequently, the GSLDWOA algorithm is employed for feature selection to remove redundant and irrelevant features and reduce the dimensionality of the data; Finally, an agent is constructed using A3C reinforcement learning, incorporating an adaptive exploration mechanism to enhance exploration in the early training phase and ensure stable convergence in the later stages. An asynchronous lag correction strategy is introduced to mitigate policy lag bias caused by asynchronous updates. This approach simultaneously shortens training and testing times while improving detection rates and accuracy and reducing false negative rates. The specific implementation process is as follows:

(1) Preprocess raw IIoT traffic and device status data, including data integration, cleansing, encoding, and normalization. The process then moves on to the feature selection stage, where the GSLDWOA feature selection algorithm is used to screen features. This algorithm simulates the hunting behavior of whales to search for the optimal solution, thereby identifying the optimal subset of features and eliminating redundant and irrelevant features.

(2) We construct APA3CID based on MDP modeling, utilizing a multi-step cumulative advantage function and an asynchronous parameter synchronization mechanism, combined with an adaptive exploration mechanism to enhance convergence stability. Simultaneously, an asynchronous lag correction strategy is introduced, employing truncated importance weights to correct policy gradients and value updates, thereby reducing the bias and gradient noise caused by asynchronous updates. This enables efficient learning of the temporal evolution patterns of attacks and optimal detection strategies, achieving rapid and high-precision identification of attack behaviors.

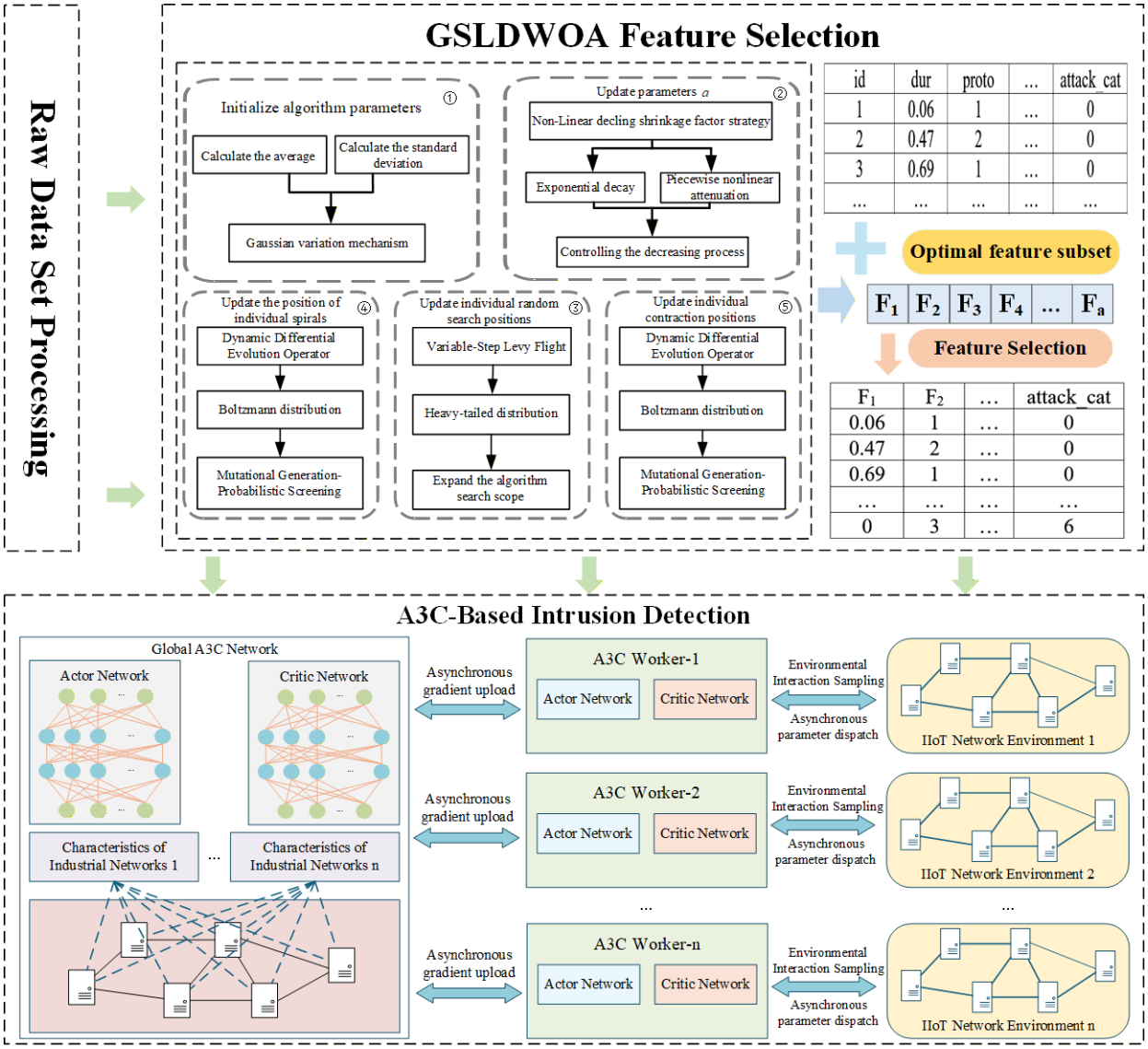

Figure 1. APA3CID Algorithm Framework.

3.1. Data Preprocessing

This paper employs a standard data preprocessing workflow, which primarily includes:

(1) Data cleaning

Remove duplicate records; exclude samples with a missing value rate exceeding 30%; impute missing values in fields with a low missing rate using the median or mode; and detect and trim outliers using the z-score.

(2) Feature encoding

For discrete features with ≤20 categories, one-hot encoding is used; for features with >20 categories or Boolean fields, label encoding is used. The encoding mapping is fitted only on the training set and applied uniformly to the test set.

(3) Standardization

Min–Max normalization is used to map all numerical features to the interval [0, 1], thereby eliminating differences in units. The normalization formula is shown in Equation (1).

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (1)$$

(4) Feature Selection

To address the issue of feature redundancy in high-dimensional industrial IoT data, this paper employs the proposed GSLDWOA feature selection algorithm, whose workflow is shown in Figure 2. This algorithm enhances the diversity of the initial population through a Gaussian mutation mechanism and employs a nonlinear decreasing contraction factor to balance global exploration and local exploitation, thereby preventing premature convergence. Furthermore, by introducing a variable-step-size Lévy flight operator and a dynamic differential evolution strategy, the algorithm improves search efficiency and convergence accuracy in high-dimensional feature spaces, effectively avoiding the trapping of local optima. For a more in-depth understanding of the algorithm's structure, parameter settings, and theoretical foundations, please refer to our previous research [23].

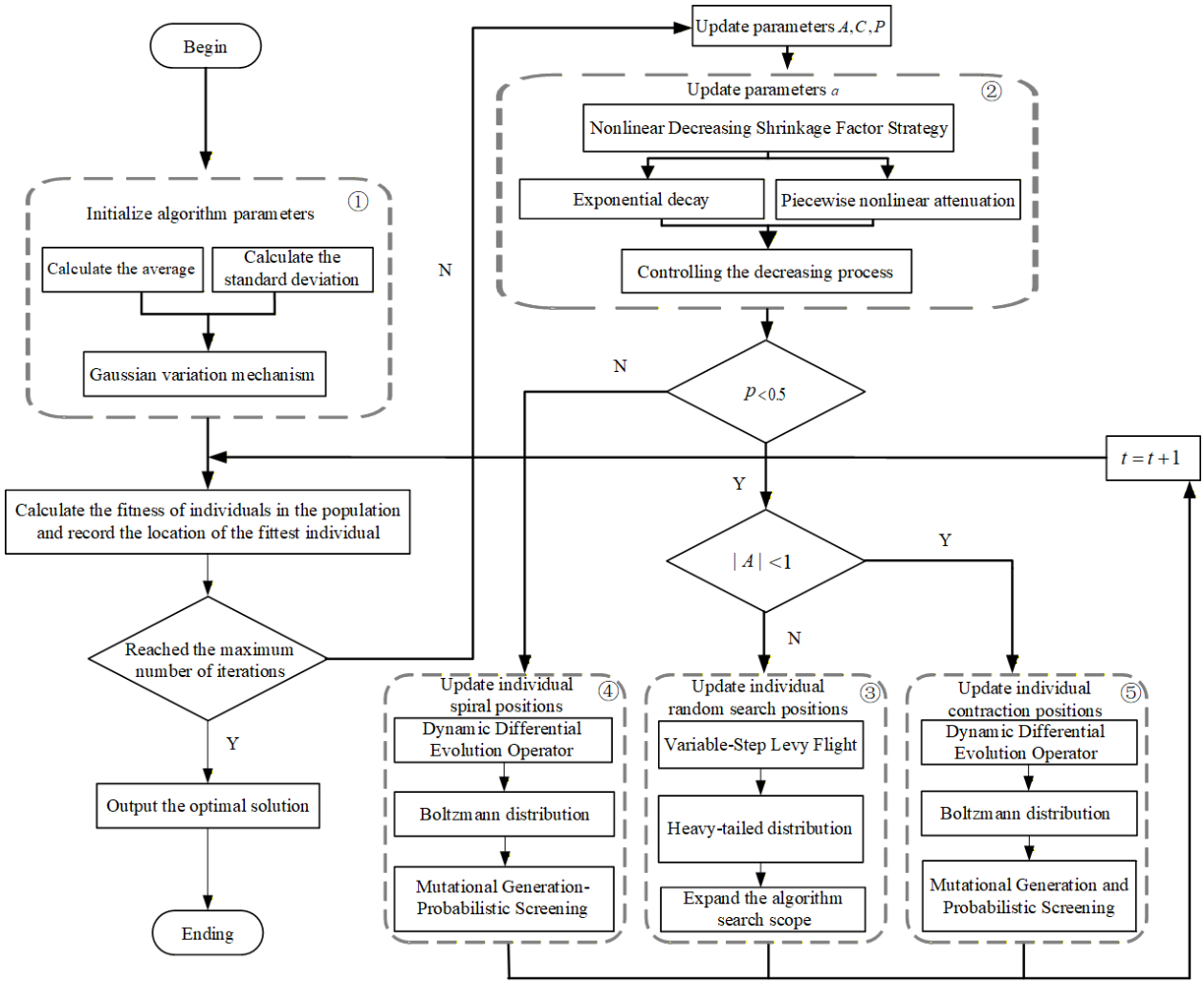

Figure 2. Flowchart of the GSLDWOA Algorithm.

In addition, to support subsequent time-series modeling and reinforcement learning training, this paper sorts and organizes the industrial IoT traffic data into sequences based on sample timestamps after completing data preprocessing. The samples are reconstructed into time-continuous sequences in chronological order, thereby providing the input foundation for subsequent sliding-window state construction and trajectory sampling.

3.2. Modeling Markov Decision Processes

To address challenges in IIoT intrusion detection—such as high-dimensional and heterogeneous data, dynamically evolving attacks, and the need for real-time response—

the Markov Decision Process (MDP) serves as the foundation for deep reinforcement learning models. In this section, the detection process for IIoT attack behaviors is modeled as a MDP. The resulting MDP is defined as a quadruple $M = < S, A, R, \gamma >$, where S represents the state space, A represents the action space, R represents the reward space, and γ represents the reward discount factor.

(1) State space

For the state space $S(t) = \{s_1, s_2 \ldots, s_t\}$, this paper defines the observation vector x_t at each time step as comprising all features in the dataset except for the label. These features include packet size, protocol type, and traffic rate, among others, and are used to characterize the traffic behavior at that time step. Given that IIoT attacks exhibit significant temporal correlations, this paper represents the state s_t using a sliding window composed of the feature vectors from the most recent L time steps, ensuring that the state incorporates both current and recent traffic characteristics. The sliding window is constructed independently on the time series of the training set and the test set, respectively; information from the test set is not used to construct the training set window, thereby preventing temporal leakage. The short-term trends in attack evolution and dynamic changes in traffic are reflected as shown in Equation (2).

$$s_t = [x_{t-L+1}, x_{t-L+2}, \ldots, x_t] \tag{2}$$

(2) Operational space

Each action a_t corresponds to the classification label of the final sample a_t in the window. Suppose the dataset contains C types of samples, including 1 type of normal sample and $K-1$ types of attack samples. In this case, the action space can be represented as $A = \{0, 1, \ldots, C-1\}$, where action $a_t = 0$ marks the current sample as normal traffic; actions $a_t = 1, 2, \ldots, C-1$ correspond to different types of intrusion attacks, such as DoS, Probe, U2R, R2L, and so on. In this paper, the action space is defined as the attack phase or category classification of the current state, and its specific instantiation depends on the set of attack labels provided in the dataset. Taking the X-IIoTID dataset as an example, the attack types are divided into nine categories, and the specific definitions of the action space are shown in Table 2.

Table 2. Definition of the Action Space for the X-IIoTID Dataset.

Action Number	Sample Category	Key Attack Characteristics
0	Normal	Complies with standard industrial network communication protocols and traffic patterns
1	Reconnaissance	Resource discovery activities such as port scanning, vulnerability scanning, and fuzz testing
2	Weaponization	Privilege escalation techniques such as brute-force attacks, dictionary attacks, and malicious internal operations
3	Exploitation	Exploits such as reverse shells and man-in-the-middle attacks
4	Lateral Movement	Internal network penetration activities, such as MQTT subscription hijacking and Modbus register reading
5	Command & Control	Transmission of remote control commands and maintenance of malicious communication
6	Exfiltration	Unauthorized export of critical data and theft of sensitive information
7	Tampering	Data tampering practices such as cloud data poisoning and the injection of false notifications
8	Crypto-Ransomware	The act of encrypting device data and demanding a ransom
9	Ransom DoS	Ransomware attacks combined with DoS attacks

(3) Reward function

The reward function is central to guiding the agent in learning the optimal detection strategy; it requires the design of differentiated reward rules that prioritize scene safety while also taking into account production requirements. If the agent is in state s_t, the true class is c_t, and the agent performs action c_t, then the immediate reward r_t is as shown in Equation (3).

$$r_t = \begin{cases} R_1 & c_t \in \{1, 2, \ldots, C-1\},\ a_t = c_t \\ R_2 & a_t = 0,\ c_t = 0 \\ -R_3 & otherwise \end{cases} \quad (3)$$

(1) Correctly detect the attack sample: If the sample is indeed attack sample $c_t \in \{0, 1, \ldots, C-1\}$ and its action matches the true class ($a_t = c_t$), then $r_t = R_1$, where $R_1 > 0$, representing the maximum positive reward.

(2) Correctly classifying a normal sample: If the sample is indeed normal ($c_t = 0$) and the action matches the true category ($a_t = 0$), then $r_t = R_2$, where $0 < R_2 < R_1$, representing a moderate positive reward.

(3) Misclassification: In addition to the two cases mentioned above, instances where a normal sample is incorrectly classified as an attack, an attack sample is incorrectly classified as normal, or the attack type is misidentified are all assigned a negative reward of $r_t = -R_3$, where $R_3 > 0$. Misclassification gives rise to two types of risks: false negatives that lead to security incidents and false positives that cause production disruptions. Both require the use of negative rewards to constrain the agent's erroneous decisions, and the value of R_3 can be dynamically adjusted based on the severity of the risk associated with each type of error. In this method, set $R_1 = 10$, $R_2 = 4$, and $R_3 = 8$.

(4) Discount factor

The discount factor $\gamma \in (0, 1)$ is used to quantify the trade-off between the immediate safety benefits of the current detection decision and the potential risks at a future time. In industrial IoT intrusion detection scenarios, attacks typically exhibit persistent and phased evolution; a false positive at a given moment may accumulate and amplify over multiple subsequent time steps, leading to more severe security risks. Therefore, a discount factor is introduced to capture the long-term impact of detection decisions on future state sequences, enabling the agent to optimize immediate detection performance while also focusing on overall safety gains across time scales.

Traditional MDPs typically include explicit state transition probabilities $P(s_{t+1}|s_t, a_t)$, which describe the probability that the environment will evolve to the next state given a particular state and action. In the methods described in this chapter, the state is defined as $s_t = [x_{t-L+1}, x_{t-L+2}, \ldots, x_t]$, which is used to characterize the local context of the most recent k time steps. Accordingly, the state transition corresponds to a shift of $s_{t+1} = [x_{t-L+2}, x_{t-L+3}, \ldots, x_{t+1}]$ in the sliding window. Since this paper constructs state sequences in chronological order on an offline dataset, s_{t+1} is exogenously determined by the data sequence and is independent of the action sequence a_t; therefore, there is no need to explicitly model transition probabilities.

In the MDP model presented in this paper, the agent's actions do not directly influence the movement of the sliding window; state transitions are determined exogenously by the temporal sequence of the dataset. However, the value of the A3C framework lies not in its ability to directly control the state of the environment through actions, but in its capacity to model long-range dependencies in time series: through a multi-step cumulative advantage function, the reward from a current observation decision can be propagated backward across multiple future time steps. In industrial IoT scenarios, cyberattacks exhibit distinct phased evolution patterns; security gaps identified at any given moment may lead to cumulative security risks in the future. Therefore, even if the action does not alter the

state transition probabilities, A3C can still effectively capture the risk propagation effects of detection decisions over time.

3.3. Design of the APA3CID Method

This paper formalizes the intrusion detection process as an MDP, clearly defining states, actions, and rewards, and unifies detection and classification under a single sequential decision-making optimization objective. Building on this foundation, this section develops an A3C-based deep reinforcement learning intrusion detection agent that learns and improves its intrusion detection and attack classification capabilities through interactive sampling and policy optimization of industrial IoT traffic sample sequences. The agent consists of a shared global Actor-Critic network and multiple parallel Worker subnetworks. The structure of the APA3CID agent is shown in Figure 3.

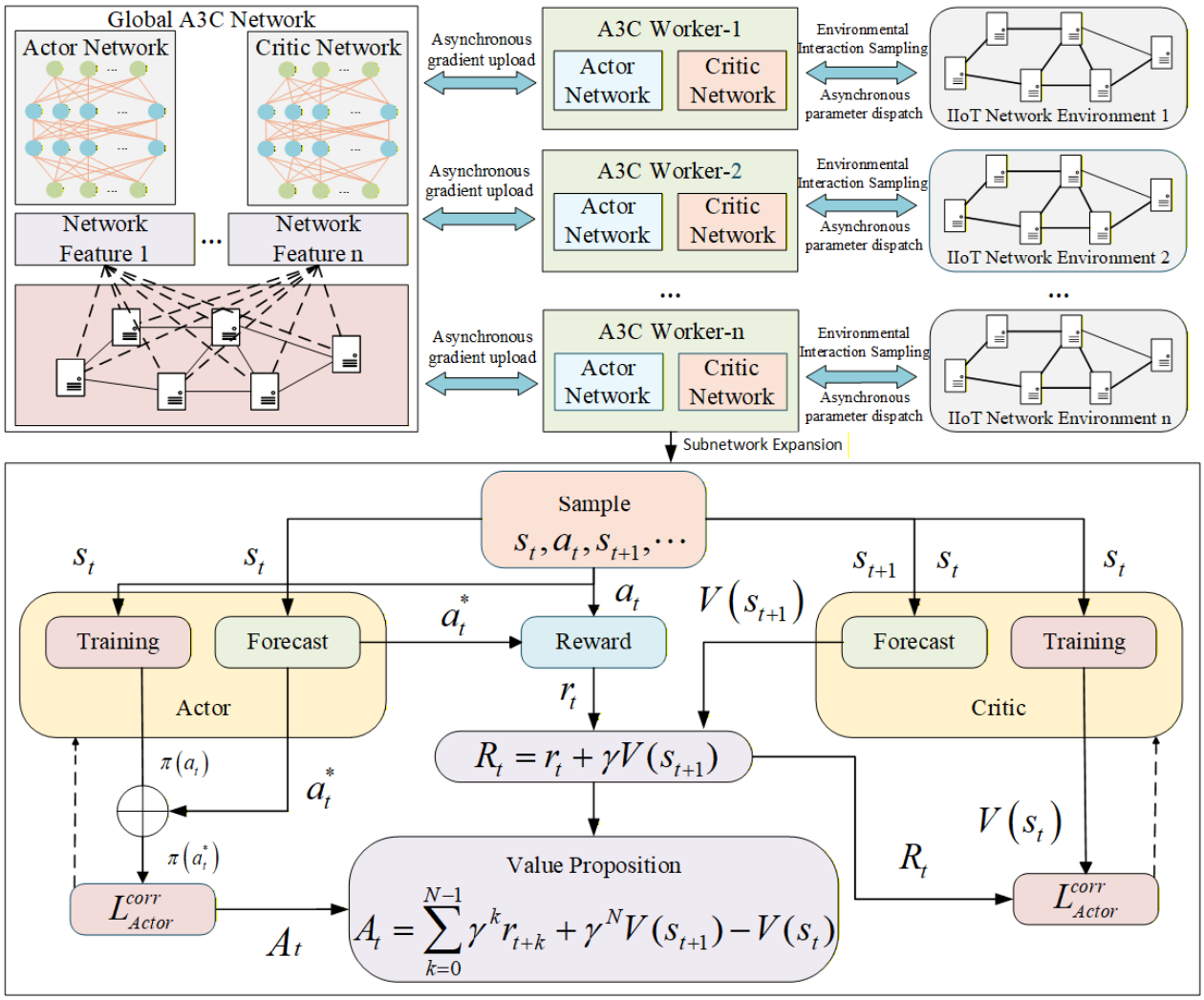

Figure 3. Structure of the APA3CID Agent.

As shown in Figure 3, the global network consists of two components—the Actor and the Critic—which are used to maintain a unified policy and state-value function. Multiple parallel workers independently perform decision learning by dividing the chronologically ordered training sequence into M non-overlapping, contiguous segments of similar length. In each worker, the agent constructs the current state s_t using a sliding window, takes the sample x_t at the end of the window as the target, outputs a probability distribution of detection actions via the actor network, and performs the corresponding attack classification;

the environment returns an immediate reward r_t based on the true label, while the next state s_{t+1} is determined by the traversal order of the dataset. After sampling N consecutive time steps, the Worker calculates the N-step return using the state values estimated by the Critic network, and then constructs a reward function to update the parameters of the local Actor and Critic networks. At the same time, the gradients computed by each worker are synchronized asynchronously to the global network, enabling the sharing and updating of policy parameters. This reduces sample correlation and improves model convergence stability while ensuring training efficiency. For each state s_t, the agent determines a specific action based on the action policy function $\pi(a_t \mid s_t, \theta_i)$, which is shown in Equation (4).

$$\pi(a_t \mid s_t, \theta_i) = \frac{\exp(z_{t,a_t})}{\sum_{c=1}^{C} \exp(z_{t,c})} \tag{4}$$

Here, $z_{t,c}$ is the unnormalized score for the cth action category output by the Actor for state s_t; z_{t,a_t} is the score corresponding to action a_t; C is the total number of action categories; $\exp(\cdot)$ is the exponential function; and $\sum_{c=1}^{C} \exp(z_{t,c})$ is used for normalization to ensure that the sum of all action probabilities equals 1.

In A3C, $V(s_t)$ is the state value function, representing the expected long-term cumulative reward that an agent can obtain by following the current policy π when in state s_t at time t. Its core function is to quantify the safety value of the current state, providing a foundation for advantage function computation and policy optimization, as shown in Equation (5).

$$V(s_t) = \mathrm{E}_{\pi(s_t)}[r_t + \gamma V(s_{t+1})] \tag{5}$$

Here, $\mathrm{E}_{\pi(s_t)}$ represents the expected value under strategy $\pi(a|s, \theta)$, reflecting the statistical average of the rewards, and $\gamma \in (0, 1)$ is the discount factor, which balances immediate rewards against long-term rewards.

R_t is the action-value function, which quantifies the long-term cumulative reward obtained by following the current policy π when in state s_t at time t and executing action a_t, as shown in Equation (6).

$$R_t = r_t + \gamma V(s_{t+1}) \tag{6}$$

Given that Industrial Internet of Things (IIoT) traffic exhibits strong temporal correlations and phased evolution, single-step returns can easily lead to value estimation biases and update noise; therefore, N-step sampling is employed to accumulate return information over a longer time span, enabling more stable learning of the policy and value function. The multi-step cumulative advantage is calculated using reward r_t, as shown in Equation (7).

$$A_t = \sum_{k=0}^{N-1} \gamma^k r_{t+k} + \gamma^N V(s_{t+N}) - V(s_t) \tag{7}$$

Here, A_t represents the N-step ahead value, N is the preset number of sampling steps, k is the time step offset, γ is the discount factor, and γ^k and γ^N are the discount weights for the kth-step reward and the Nth-step state value, respectively. The sliding window state is used to characterize short-term traffic and attack evolution patterns, while the N-step sampling mechanism is used to assess the impact of current detection decisions on future multi-step security risks; together, they model the IIoT attack process across different time scales. The global iterative training reward is shown in Equation (8).

$$\mathrm{Reward}^{(i)} = \frac{1}{M \cdot T_{local}} \sum_{m=1}^{M} \sum_{t=1}^{T_{local}} \left(\sum_{k=0}^{N-1} \gamma^k r_{t+k}^{(m)} \right) \tag{8}$$

Here, Reward$^{(i)}$ represents the average reward value for the ith iteration, M represents the number of subnetworks, and T_{local} represents the number of local iterations for each subnetwork.

3.4. Adaptive Search

To prevent the policy from converging prematurely to a local optimum, an entropy regularization term is typically introduced into the actor's objective function to encourage exploration. However, a fixed entropy coefficient struggles to strike a balance throughout the entire training process; insufficient exploration in the early stages of training can lead to convergence on a suboptimal policy, while excessive exploration in the later stages can make it difficult for the policy to converge stably, introducing unnecessary fluctuations in false positives. To this end, this paper proposes an adaptive exploration mechanism that takes into account both the training phase and sample uncertainty to dynamically adjust the entropy coefficient.

By extending the entropy coefficient from a constant to a function $\beta(t)$ that varies with the training process, we achieve a training schedule characterized by aggressive exploration in the early stages and strong convergence in the later stages. The entropy coefficient function is shown in Equation (9).

$$\beta(t) = \max(\beta_{\min}, \beta_0 \cdot \alpha^t) \tag{9}$$

Here, $\beta(0)$ is the initial entropy coefficient, $\beta_{\min}$ is the lower bound of the entropy coefficient, $\alpha \in (0,1)$ is the annealing rate, and t is the number of global update steps. As training progresses, $\beta(t)$ gradually decreases, causing the strategy to transition from early exploration to later deterministic decision-making, which helps improve convergence stability and suppress fluctuations in false positives.

Outside the training phase, for states where the model exhibits high uncertainty, exploration should be increased to avoid making erroneous decisions driven by overconfidence in handling difficult samples. This paper adopts policy entropy as a measure of uncertainty and normalizes it as shown in Equation (10).

$$U(s_t) = \frac{H(\pi(\cdot \mid s_t))}{\log|\mathrm{A}|} \in [0,1] \tag{10}$$

Here, $U(s_t)$ is the uncertainty metric at time t, $H(\pi(\cdot \mid s_t))$ represents the entropy of the actor's action distribution, and $\log|\mathrm{A}|$ is the logarithm of the size of the action space. When the strategy distribution is relatively uniform and entropy is high, β_t is increased to enhance exploration. When the strategy distribution is relatively well-defined and the entropy is low, β_t approaches $\beta(t)$, which promotes convergence. The state-adaptive entropy coefficient is given by Equation (11).

$$\beta_t = \beta(t) \cdot (1 + U(s_t)) \tag{11}$$

3.5. Asynchronous Lag Correction

In A3C's asynchronous parallel training, each worker interacts with the environment using local parameters to sample trajectories and periodically synchronizes gradients asynchronously to the global network. Because the global network continues to evolve during the update process involving multiple workers, the policy used by the workers for sampling lags behind the current global target policy, thereby introducing off-policy bias. In long-sequence traffic scenarios within the IIoT, attack chains exhibit phased evolution and strong temporal correlations; class imbalance and cost-sensitive rewards amplify the instability caused by high-probability shifts, leading to cumulative bias.

To mitigate the lag bias in asynchronous policy learning, this paper introduces truncated importance sampling to correct the policy gradient and value objective without altering the original asynchronous training framework. The importance ratio is given by Equation (12).

$$\rho_t = \frac{\pi(a_t \mid s_t, \theta_i)}{\mu(a_t \mid s_t, \theta_i^-)} \tag{12}$$

Here, $\mu(a_t \mid s_t, \theta_i^-)$ represents the policy probability of the old parameter θ_i^- generated by the local actor at the worker's sampling time, and $\pi(a_t \mid s_t, \theta_i)$ represents the target policy probability of the current global actor.

Given that an excessively large ratio may introduce high variance, the truncated form $\overline{\rho}_t = \min(\rho_{\max}, \rho_t)$ is adopted. APA3CID builds upon the Actor-Critic dual-network architecture of A3C, optimizing the parameter update logic based on MDP modeling results to enable collaborative learning of detection strategies and state values. The actor policy network is responsible for outputting a probability distribution $\pi(a_t \mid s_t, \theta_i)$ over the action space. Its parameters are updated to maximize cumulative advantage by weighting the advantage term in the original policy gradient, resulting in the corrected loss function for the actor network as shown in Equation (13).

$$L_{Actor}^{corr} = -\mathrm{E}[\overline{\rho}_t \log \pi(a_t \mid s_t, \theta_i) \cdot A_t + \beta_t \cdot H(\pi(a_t \mid s_t, \theta_i))] \tag{13}$$

Here, L_{Actor}^{corr} represents the calibrated loss value of the Actor policy network; E is the statistical average over the batch data; $\pi(a_t|s_t, \pi_i)$ is the probability that the ith subnetwork's Actor parameter θ_i performs action a_t in state S_t at time t; $\log \pi(\cdot)$ is the natural logarithm of the policy probability; after updating the subnetwork parameters θ_i via gradient descent, they are asynchronously synchronized to the main network parameters θ, ensuring that the global policy aggregated by the main network covers the attack recognition capabilities of each sub-scenario.

Regarding the update of the value network, to ensure consistency between the Critic's learning objective and the corrected policy, a truncation coefficient $c_t = \min(c_{\max}, \rho_t)$ is introduced to robustly correct the TD error, and the corrected TD error is constructed as shown in Equation (14).

$$\delta_t^{corr} = \overline{\rho}_t \left(\sum_{k=0}^{N-1} \gamma^k r_{t+k} + \gamma^N V(s_{t+N}, \varphi_i) - V(s_t, \varphi_i) \right) \tag{14}$$

where r_{t+k} is the reward at time $t+k$, s_{t+N} represents the state representation at time $t+N$, V_i represents the parameters of the ith subnetwork (Critic value network), and $V(s_{t+N}, \varphi_i)$ and $V(s_t, \varphi_i)$ represent the value estimates for the future and current states at times $t+N$ and t, respectively. By minimizing the squared error between the N-step cumulative reward and the current state value, the Critic value network can accurately learn the mapping relationship between state and safety value, providing a reliable value benchmark for the Actor network and preventing decision errors caused by value estimation biases.

The Critic value network is responsible for estimating the state value $V(s_t, \phi_i)$. Parameter updates aim to minimize the error between the cumulative reward corresponding to the N-step ahead advantage and the predicted value. The loss function for the Critic value network is defined as shown in Equation (15).

$$L_{critic}^{corr} = \frac{1}{2}\mathrm{E}[(\delta_t^{corr})^2] \tag{15}$$

Here, L_{critic}^{corr} represents the loss value of the Critic network after calibration, and E represents the statistical mean of the batch data. The loss function of the Critic network for the ith global iteration is shown in Equation (16).

$$L_{\text{Critic}}^{(i)} = \frac{1}{M \cdot T_{local}} \sum_{m=1}^{M} \sum_{t=1}^{T_{local}} \frac{1}{2} \left(\sum_{k=0}^{N-1} \gamma^k r_{t+k}^{(m)} + \gamma^N V\left(s_{t+N}^{(m)}\right) - V\left(s_t^{(m)}\right) \right)^2 \tag{16}$$

The APA3CID algorithm is shown in Algorithm 1. First, the dataset is preprocessed, and then the GSLDWOA algorithm is applied for feature selection to eliminate redundant and irrelevant features, thereby reducing the dimensionality of the data. By incorporating an adaptive exploration mechanism and an asynchronous lag correction strategy, the model training and testing times are significantly reduced, resulting in improved detection rates and accuracy.

Algorithm 1. APA3CID: An Intrusion Detection Algorithm Based on Feature Optimization and Asynchronous Actor-Critic Learning

Input: Training dataset D_{train}, Test dataset D_{test}
Output: Test results for test dataset D_{test}
Data Preprocessing
Initialize the global actor network policy π_{glob}, the maximum number of iterations T_{glob} for the main network, the main network iteration counter $t_{glob} = 0$, M subnetworks, the maximum number of iterations T_{local} for each subnetwork, and the local step counter $t = 0$
while $t_{glob} < T_{glob}$ do
  for $i = 1$ to M in parallel do
    $\theta_i^- \leftarrow \theta$, $\varphi_i \leftarrow \varphi$
    while $t < T_{local}$ do
      for $j = 0$ to $N - 1$ do
        $a_{t+j} \sim \mu(a_t | s_{t+j}, \theta_i^-)$
        The environment returns a reward r_{t+j} and yields the next state s_{t+j+1}
      end for
      $A_t = \sum_{k=0}^{N-1} \gamma^k r_{t+k} + \gamma^N V(s_{t+N}) - V(s_t)$
      $\beta_t = \beta(t) \cdot (1 + U(s_t))$
      $\overline{\rho}_t = \min(\rho_{\max}, \rho_t)$
      $L_{Actor}^{corr} = -\mathrm{E}[\overline{\rho}_t \log \pi(a_t \mid s_t, \theta_i) \cdot A_t + \beta_t \cdot H(\pi(a_t \mid s_t, \theta_i))]$
      $L_{critic}^{corr} = \frac{1}{2}\mathrm{E}\left[\left(\overline{\rho}_t \left(\sum_{k=0}^{N-1} \gamma^k r_{t+k} + \gamma^N V(s_{t+N}, \varphi_i) - V(s_t, \varphi_i) \right) \right)^2 \right]$
      θ_i: $\theta_i \leftarrow \theta_i - \eta * \nabla \theta_i L_{Actor}^{corr}$
      V_i: $\varphi_i \leftarrow \varphi_i - \eta * \nabla \varphi_i L_{critic}^{corr}$
      $t = t + N$
      $t_{glob} = t_{glob} + 1$
    end while
  end parallel for
end while
return Test results for test dataset D_{test}

4. Experimental Design and Discussion of Results

4.1. Experimental Environment and Dataset

(1) Experimental Environment

To verify the effectiveness and performance of the proposed APA3CID algorithm, this chapter establishes a standardized experimental environment to conduct evaluation experiments. On the hardware side, it features an Intel (R) Core (TM) i7-13700K processor to ensure efficient data processing, 32 GB of DDR5 4800 MHz memory to handle large-scale data sets, and an NVIDIA GeForce RTX 4080 16 GB discrete graphics card. On the software side, the experimental setup consists of a 64-bit Windows 11 Pro operating system. The development environment was built using the Python 3.9.16 programming language. TensorFlow 2.13.0 was selected as the deep learning framework to implement the construction and training of the APA3CID model. The Scikit-Learn 1.3.0 library was used to perform data preprocessing and calculate model evaluation metrics, while NumPy 1.26.0 was employed to enable efficient numerical computations. Pandas 2.1.0 was used to read, perform basic cleaning, and convert the dataset. The key training parameter settings for APA3CID are shown in Table 3.

Table 3. APA3CID Training Parameters.

Experimental Parameters	Parameter Value
Sliding Window L	15
Maximum number of iterations T_{glob} for the main network	100
Maximum number of iterations T_{local} for a subnetwork	1000
Number of subnetworks M	4
Number of sampling steps N	15
Discount factor γ	0.75
Initial entropy regularization coefficient β_0	0.01
Lower bound of the entropy coefficient $\beta_{\min}$	0.001
Attenuation coefficient α	0.999
Importance weight truncation upper limit threshold $\rho_{\max}$	2
Critic-side threshold $c_{\max}$ for clipping the upper bound of weights	1
Learning rate η	0.01

(2) X-IIoTID dataset

The X-IIoTID dataset [24] is an IIoT intrusion detection benchmark dataset released by Muna Al-Hawawreh's team in 2021. Its core value lies in being the first to systematically cover the full stack of IIoT activities and the entire attack lifecycle—from reconnaissance to ransomware—while being designed for a connection-agnostic, device-agnostic heterogeneous environment that closely mirrors real-world industrial scenarios such as smart manufacturing and energy monitoring. The dataset contains 820,834 samples, including 421,417 normal samples and 399,417 attack samples. It comprises 65 mixed features derived from network traffic, system status, logs, and application-layer protocols, and provides three levels of fine-grained labels. This makes it suitable for validating the capabilities of the APA3CID framework described in this paper for modeling the evolution of multi-stage attacks and processing high-dimensional heterogeneous features. The characteristics and types of the X-IIoTID dataset are shown in Table 4.

(3) WUSTL-IIoT-2021 dataset

The WUSTL-IIoT-2021 dataset [25] is an IIoT security dataset released by Washington University. It focuses on the detection of cyberattacks against industrial control systems and contains 1,194,464 records and 41 features, with 87,016 attack samples. and 1,107,448 normal samples. It encompasses both normal traffic and various attack types, including DoS, MITM, replay attacks, and malicious firmware injection, clearly illustrating the typical challenges

of attack sparsity and extreme class imbalance in industrial settings. This dataset validates the robustness of APA3CID in achieving low false-negative rates for covert attacks such as R2L and U2R. Table 5 lists 41 structured features covering multi-dimensional metrics such as traffic statistics, transmission quality, and temporal behavior. These features effectively capture the temporal patterns of IIoT-specific threats, such as DoS, MITM, and replay attacks, thereby providing the necessary data foundation for the modeling of multi-step cumulative advantage and sliding window states in this paper. Since the X-IIoTID dataset and the WUSTL-IIoT-2021 dataset do not provide pre-defined training and test sets, this paper divides the data into training and test sets in a 7:3 ratio. The features and types of the WUSTL-IIoT-2021 dataset are shown in Table 5.

Table 4. Characteristics and Types of the X-IIoTID Dataset.

Feature Category	Number of Features	Data Types	Example Features
Network Traffic Characteristics	28	Continuous/ Discrete	Discrete:Src_IP, Protocol_Type, SYN_Flag_Count; Continuous: Flow_Duration, Flow_Bytes/s, Total_Fwd_Pkts
System Status Characteristics	15	Continuous	CPU_Usage, Memory_Usage, Disk_IO_Read, Device_Temperature
Log-derived features	12	Continuous/ Discrete	Discrete:Alert_Level, Rule_ID, Attack_Signature_Match; Continuous: Alert_Count, Rule_Confidence, Time_Since_Last_Alert
Characteristics of Application Layer Protocols	10	Continuous/ Discrete	Discrete:MQTT_Topic, CoAP_Method, WebSocket_Opcode; Continuous: App_Payload_Length, Payload_Entropy
Level 3 tag attributes	3	Discrete	Label_Level1 (0 = Normal/1 = attack), Label_Level2 (Attack Category Coding), Label_Level3 (Attack subtype code)

Table 5. Features and Types of the WUSTL-IIoT-2021 Dataset.

Feature Category	Number of Features	Examples of Data Types	Core Features
Traffic Statistics	9	Discrete: Spkts, Dpkts; Continuous: TBytes, Sbytes	Count the total number of packets and bytes in network traffic to identify traffic anomalies such as DoS floods
Speed and Load Characteristics	6	Continuous: Sload, Dload, Trate, Srate	Measuring traffic load per unit of time, a DoS attack causes a sudden spike in traffic rate or load
Transmission Quality Characteristics	7	Continuous: SrcJitAct, DstJitAct, Ploss, RTT	Indicates transmission stability; reconnaissance/DoS attacks can cause jitter and a significant increase in packet loss
Connection Properties	5	Discrete: Proto; Continuous: TcpRtt, Dur	Describe the basic properties of connections, distinguish between industrial protocols, and identify abnormally short connections
Temporal characteristics	4	Continuous: Idle, RunTime, mean, std	Recording changes in traffic over time reveals that the intervals between scan attacks follow a pattern
Characteristics of Characters and Services	10	Discrete:sTtl, dTtl; Continuous: SAppBytes, DAppBytes	Distinguish between application-layer and network-layer data, and identify malicious application-layer payloads

Both the X-IIoTID and WUSTL-IIoT-2021 datasets used in this paper are time-series industrial flow datasets. To prevent time-series leakage, the data was strictly partitioned

based on the chronological order of the sample timestamps, with the first 70% used as the training set and the remaining 30% as the test set. Neither random nor stratified partitioning was employed. All preprocessing, feature selection, and sliding window construction are performed exclusively on the training set. The encoding rules, normalization parameters, optimal feature subset, and window parameters are fixed before being applied to the test set, ensuring that no data is transferred across the training and test set boundaries. All experiments were repeated 30 times independently, and the final results are presented as mean values. A comparison of the model configurations is shown in Table 6.

Table 6. Comparison of Model Configuration Instructions.

Model Name	Key Features	Hyperparameter Settings
LR (Logistic Regression)	L2 regularization	C = 1.0, max_iter = 1000, solver = 'lbfgs'
RF (Random Forest)	100 decision trees	max_depth = 15, min_samples_split = 10, n_estimators = 100
XGBoost	Gradient Boosting Trees	n_estimators = 100, max_depth = 6, learning_rate = 0.1
CNN	Convolutional Neural Networks	Convolutional layers: 3 layers (32/64/128), kernel_size = 3; Fully connected layer: 128-dimensional; dropout = 0.3
LSTM	Long-Short Term Memory Networks	LSTM layers: 2 layers (64 × 64); Fully connected layer: 128-dimensional; dropout = 0.3
DQN	Deep Q-Network	Experience pool: 10,000; Target network updates: 100 steps; Exploration rate ε: 0.9 → 0.01
DDQN	Double-Depth Q-Network	Similar to DQN, it uses dual-network decoupling
A3C	Advantages of Asynchronous Actor-Critic	Number of subnetworks: 4; Number of steps: 5; Learning rate: 0.001
IA3C	Improvements to A3C	Entropy regularization coefficient: 0.01; other settings are the same as in A3C

4.2. Evaluation Criteria

In industrial IoT intrusion detection, a true positive (TP) is observed when an attack is correctly predicted, a true negative (TN) is observed when a normal event is correctly predicted, a false positive (FP) is observed when the model incorrectly predicts a positive event, and a false negative (FN) is observed when the model incorrectly predicts a negative event. There are various metrics for evaluating network intrusion detection algorithms. This method employs metrics based on confusion matrices to provide an intuitive reflection of model performance, including accuracy (Acc), precision (Pre), detection rate (DR), and F1-score [26].

(1) Accuracy

The *Acc* is used to measure the model's classification accuracy across all samples, including correctly predicted positive and negative samples, as shown in Equation (17).

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{17}$$

(2) Precision

The *Pre* measures the proportion of samples classified as positive by the model that are actually positive. A high precision indicates that the model is highly accurate in identifying

positive samples and can effectively reduce the number of negative samples misclassified as positive, as shown in Equation (18).

$$Pre = \frac{TP}{TP + FP} \tag{18}$$

(3) Detection Rate

The DR, also known as recall, is a core metric for evaluating a classification model's ability to detect true positive samples, as shown in Equation (19).

$$DR = \frac{TP}{TP + FN} \tag{19}$$

(4) F1-score

The F1-score takes both precision and recall (detection rate) into account; it is the harmonic mean of precision and recall, and provides a more comprehensive reflection of a model's performance, as shown in Equation (20).

$$\text{F1-score} = 2 * \frac{PRE * DR}{PRE + DR} = \frac{2 * TP}{2 * TP + FP + FN} \tag{20}$$

4.3. Experimental Results

4.3.1. Ablation Experiments and Performance Evaluation

To eliminate the effects of randomness introduced by random initialization, data sampling, and reinforcement learning exploration strategies, all ablation models in this paper were tested across 30 independent experiments using different random seeds to ensure experimental stability and reproducibility. To validate the effectiveness of each core module of APA3CID, this experiment utilized the X-IIoTID dataset to conduct ablation tests on the five algorithms—Baseline A3C, Baseline A3C + GSLDWOA, baseline A3C + GSLDWOA + adaptive exploration, baseline A3C + GSLDWOA + adaptive exploration + asynchronous lag correction, and APA3CID. The results of the ablation experiments are shown in Table 7.

Table 7. Ablation Experiment Results.

Model Number	Model	Acc (Mean ± Std)	Pre (Mean ± Std)	DR (Mean ± Std)	F1-Score (Mean ± Std)	Train Time (s)
①	Baseline A3C	0.8540 ± 0.0062	0.8171 ± 0.0075	0.9462 ± 0.0041	0.8867 ± 0.0053	1635 ± 42
②	① + GSLDWOA	0.8667 ± 0.0051	0.8615 ± 0.0063	0.9532 ± 0.0038	0.8913 ± 0.0046	1356 ± 35
③	② + Adaptive exploration	0.8851 ± 0.0044	0.8653 ± 0.0057	0.9642 ± 0.0033	0.9030 ± 0.0041	1226 ± 31
④	③ + Asynchronous lag correction (Policy Only)	0.8959 ± 0.0039	0.8498 ± 0.0052	0.9686 ± 0.0029	0.9025 ± 0.0037	1093 ± 28
⑤	APA3CID	0.9217 ± 0.0028	0.8756 ± 0.0043	0.9794 ± 0.0024	0.9245 ± 0.0030	986 ± 24

As shown by the ablation experiment results based on the X-IIoTID dataset in Figure 4, APA3CID achieved an accuracy of 92.17%, a precision of 97.94%, and an F1-score of 92.45%, outperforming the other four modules. After incorporating the GSLDWOA feature selection, the baseline A3C model enhanced its global search capabilities through Gaussian variation and variable-step Levy flight, effectively reducing redundant and irrelevant features. Although there was no significant improvement in the model's DR and F1-score, training time was significantly reduced to 1356 s. The incorporation of an adaptive exploration mechanism enables the agent to maintain a higher level of exploration during the early stages of training, mitigating the risk of getting stuck in local optima and enhancing its ability to detect attacks from minority classes. As a result, the DR and F1-score improved to

96.42% and 90.30%, respectively. After introducing an asynchronous lag correction strategy, we adjusted the policy gradient and value updates by truncating the importance weights, resulting in more stable policy optimization. The DR improved to 96.86%, the F1-score reached 90.25%, and the training time was 1093 s.

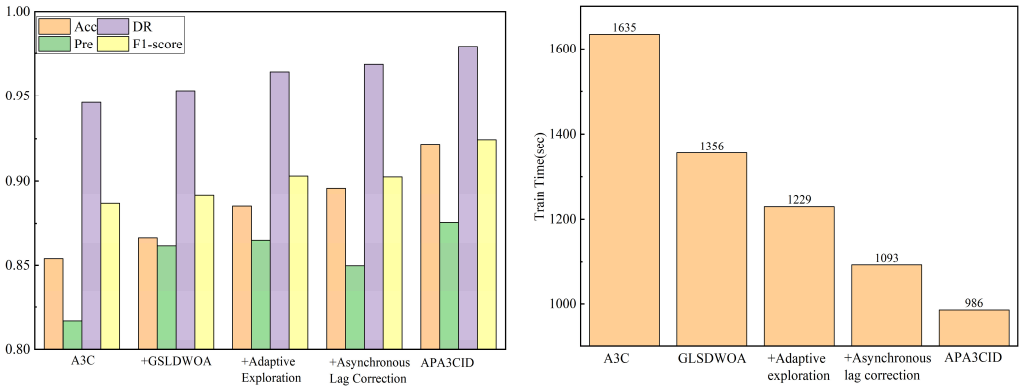

Figure 4. Results of the APA3CID algorithm ablation experiment.

As shown by the training curve results in Figure 5, APA3CID outperforms the comparison benchmarks in terms of both cumulative reward and convergence stability. The average rewards for all four methods increased with each iteration and eventually stabilized. Among them, APA3CID showed a faster increase in reward and achieved the highest steady-state value, indicating that it is better able to converge to an optimal strategy. The rapid decline and sustained lower level of the Critic Loss indicate that the error in the value function estimation is smaller and the gradient signal is more stable, which helps suppress oscillations during the policy update process and improves the convergence and robustness of training. After approximately 80 iterations, the results stabilized, further confirming that the proposed method converges well.

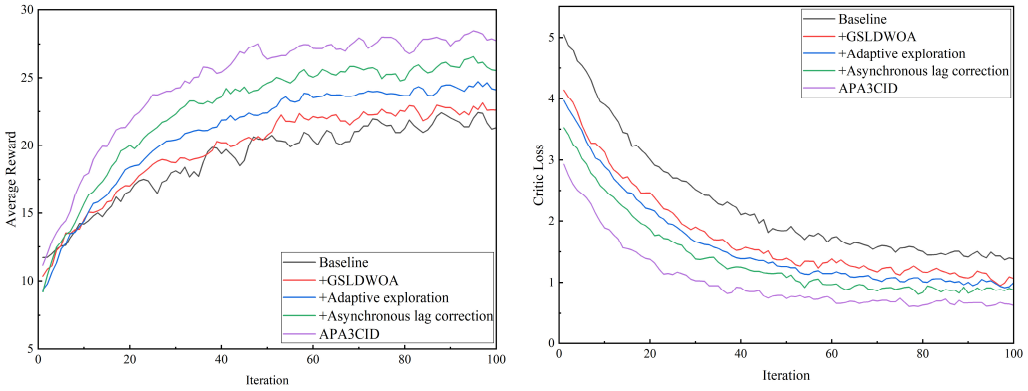

Figure 5. Training curve of APA3CID on the X-IIoTID dataset.

To evaluate the intrusion detection performance of APA3CID on the X-IIoTID dataset, this paper conducted a binary classification experiment on the test set to obtain a confusion matrix; the results are shown in Figure 6. In the binary classification confusion matrix, the model correctly identified 108,422 normal traffic samples, misclassified 16,721 as attacks, correctly identified 117,634 attack traffic samples, and misclassified only 2474 attack samples as normal. The confusion matrix uses varying shades of color within the same row to distinguish the classification performance of different labels, providing a visual representation of the APA3CID's intrusion detection capabilities.

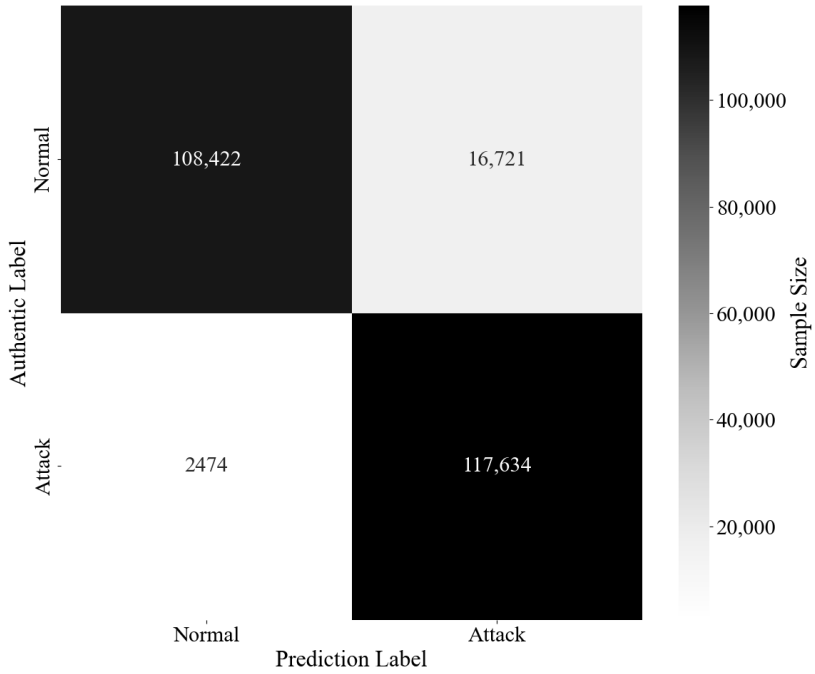

Figure 6. Binary classification confusion matrix for APA3CID based on the X-IIoTID dataset.

4.3.2. Comparative Experiment

(1) Results from the X-IIoTID dataset

This section uses the X-IIoTID dataset to compare the accuracy, precision, recall, and F1-score of traditional and mainstream intrusion detection models, including APA3CID, LR, RF, XGBoost, CNN, LSTM, DQN, DDQN [27], A3C, and IA3C [28].

As shown in the experimental results in Figure 7, compared to the other nine methods, APA3CID achieved the highest metrics across the board, with an accuracy of 92.17%, a precision of 87.56%, a recall of 97.94%, and an F1-score of 92.45%. Compared to A3C, the recall and F1-score improved by approximately 3.51% and 4.26%, respectively. Due to their limited feature representation capabilities, LR, RF, and XGBoost generally achieved accuracy rates below 80%. CNNs and LSTMs are essentially supervised static mapping models that lack mechanisms for dynamic interaction with the environment; their accuracy and precision are slightly higher. DQN, DDQN, A3C, and IA3C incorporate a reward mechanism, but they suffer from high estimation variance and limited exploration efficiency, making them prone to getting stuck in local optima.

(2) Results from the WUSTL-IIoT-2021 dataset

This section uses the WUSTL-IIoT-2021 dataset to compare the accuracy, precision, recall, F1-score, and training time of APA3CID against traditional and mainstream intrusion detection models such as LR, RF, XGBoost, CNN, LSTM, DQN, DDQN, A3C, and IA3C.

As shown in the experimental results in Figure 8, APA3CID achieved the highest metrics compared to the other nine methods, with a recall of 90.12%, a precision of 89.14%, a detection rate of 98.35%, and an F1-score of 93.32%. The detection rate and F1-score improved by approximately 7.46% and 6.76%, respectively, compared to IA3C, fully meeting the requirements of IIoT scenarios for low false negative rates. Due to difficulties in adapting to the complex data distributions of the IIoT, LR, RF, and XGBoost achieved an accuracy of only 65.32% and an F1 score as low as 65.61%. Thanks to improvements in automatic feature extraction, the F1 scores for CNN and LSTM increased to 79.36% and 81.57%, respectively. Methods such as DQN, DDQN, and DDPG lack scenario-specific adaptation to industrial protocols and device characteristics, resulting in relatively low policy learning efficiency.

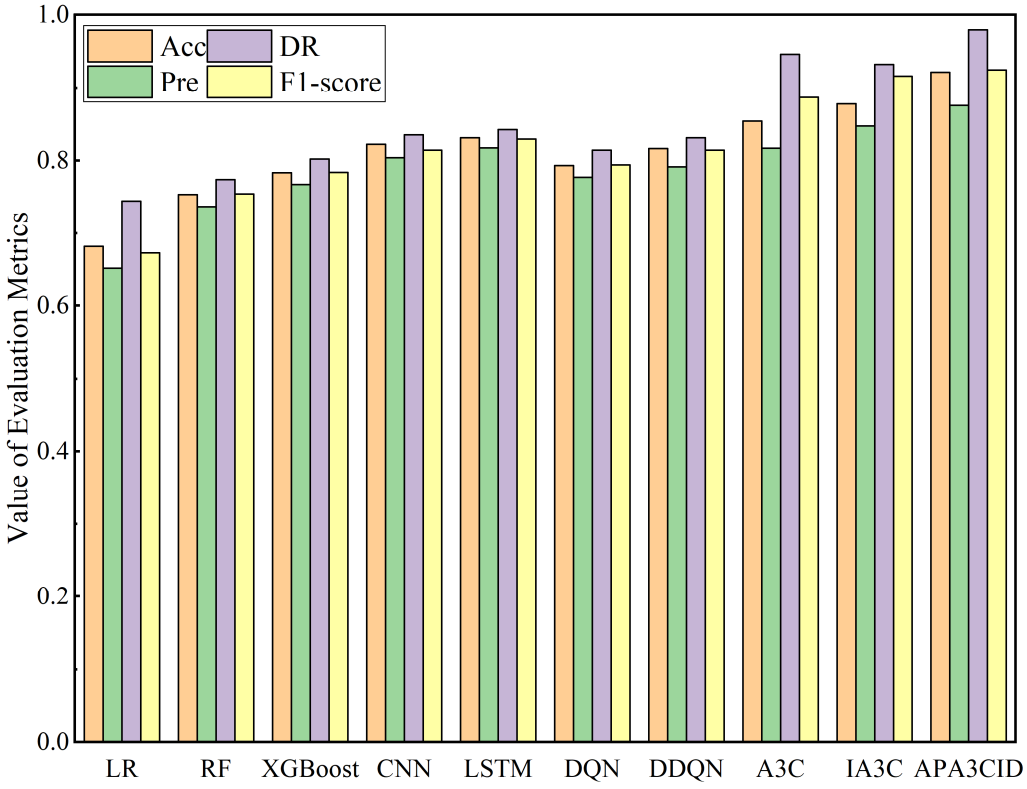

Figure 7. Performance comparison of APA3CID with other models on the X-IIoTID dataset.

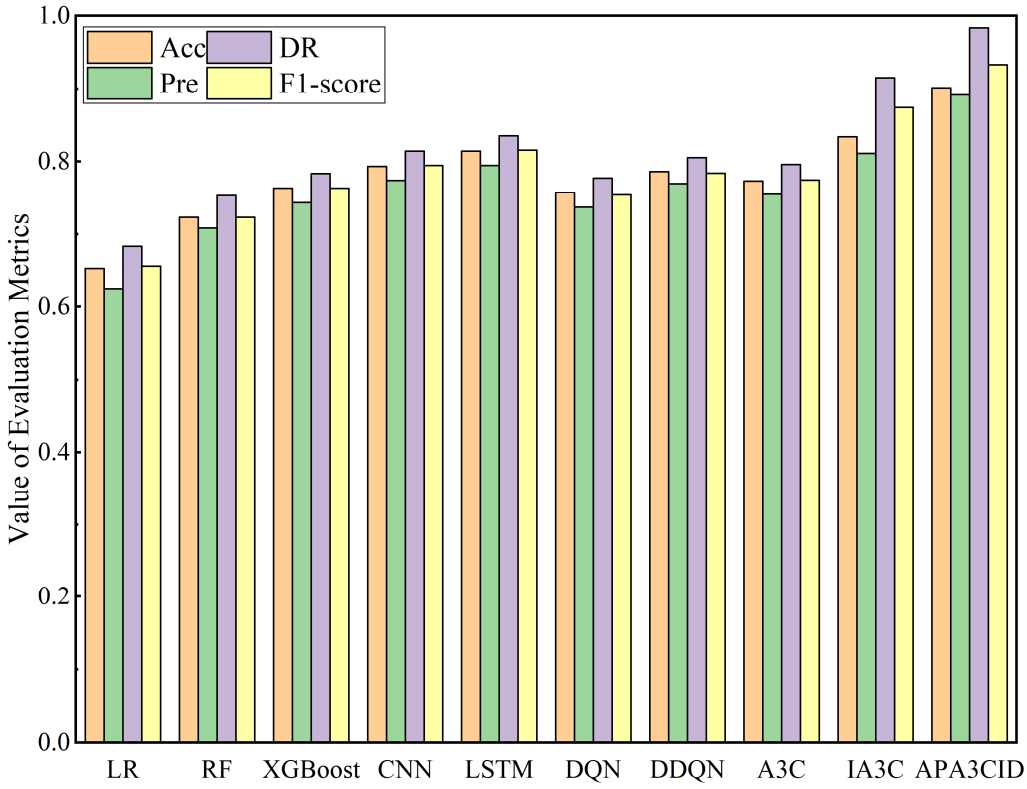

Figure 8. Performance comparison of APA3CID with other models on the WUSTL-IIoT-2021 dataset.

Figure 9 shows a comparison of the training times for APA3CID and other models on the WUSTL-IIoT-2021 dataset. The results indicate that, compared to general machine learning, deep learning, and reinforcement learning algorithms, APA3CID places greater emphasis on balancing detection performance and training efficiency. GSLDWOA feature

selection can eliminate redundant and irrelevant features, enhance the discriminative power of effective features, and thereby reduce the false negative rate. A3C-based adaptive exploration and asynchronous lag correction enhance the stability of policy updates, making the model more sensitive to attack samples and thereby improving DR. Consequently, the aforementioned feature selection and multi-network asynchronous optimization introduce additional computational overhead. Factors such as the high feature dimensionality of the WUSTL-IIoT-2021 dataset mean that the overhead incurred by feature selection and additional adjustments outweighs the benefits, resulting in training times that are slightly longer than those of the traditional A3C model.

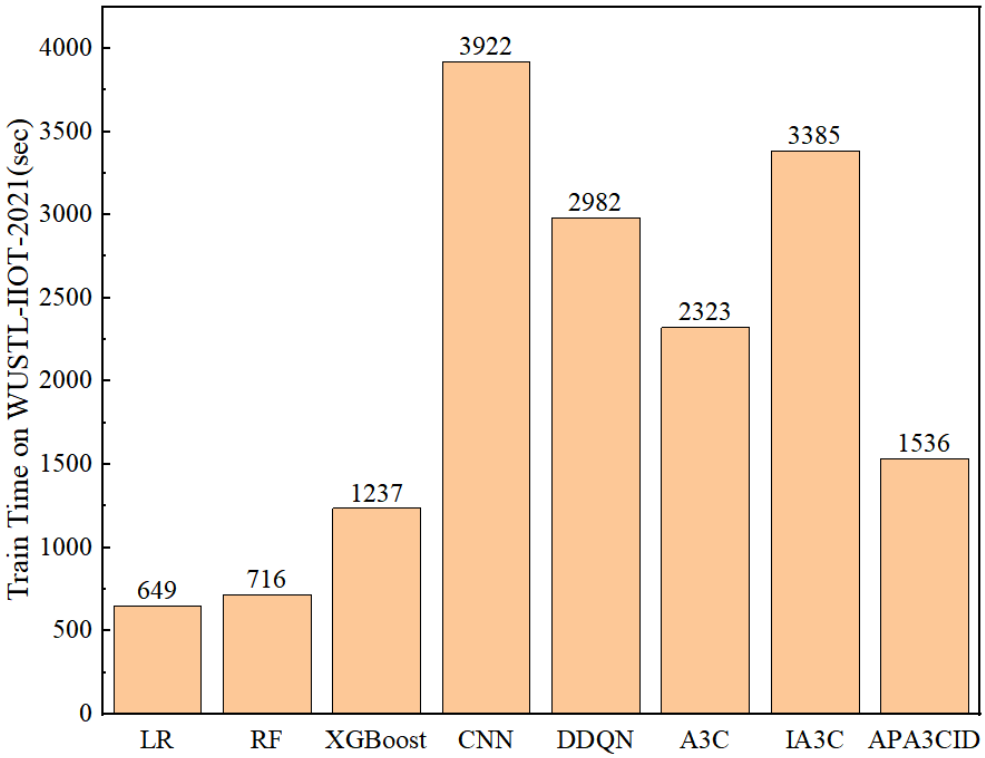

Figure 9. Comparison of training times for APA3CID and other models on the WUSTL-IIoT-2021 dataset.

To verify the deployment efficiency of APA3CID, we further tested its prediction time on public datasets. On the X-IIoTID dataset, which contains 246,251 samples, the model's overall prediction time was 2.91 s, corresponding to a prediction time of approximately 11.82 microseconds per sample. On the WUSTL-IIoT-2021 test set, which contains 332,234 samples, the total prediction time was 3.84 s, with a prediction time of approximately 11.56 microseconds per sample. In addition, this paper further supplements the core efficiency metrics related to model deployment:

(1) Model parameter size: The APA3CID model has approximately 1.2 million parameters, which is significantly fewer than those of typical deep learning detection models;

(2) Computational complexity: The computational load (FLOPs) for single-sample inference is approximately 0.8 MFLOPs, resulting in extremely low computational overhead;

(3) Memory usage: Peak memory usage during the inference phase is approximately 15 MB, which is well within the resource constraints of IIoT edge devices.

5. Conclusions

This paper addresses core challenges in IIoT intrusion detection, such as feature redundancy in high-dimensional data, poor adaptability to attack dynamics, and the difficulty of

balancing detection accuracy and efficiency. It proposes an intrusion detection algorithm based on feature optimization and asynchronous actor-critic learning. By establishing a processing workflow that combines data preprocessing with GSLDWOA feature selection, the algorithm accurately identifies key attack features and resolves the issue of noise interference in high-dimensional data; A modeling framework based on Markov processes is designed to address the difficulty traditional models face in adapting to the dynamic nature of industrial scenarios; adaptive exploration mechanisms and asynchronous lag correction strategies are introduced to enhance convergence stability and robustness, while reducing strategy bias caused by asynchronous updates. Experimental results show that APA3CID achieved F1 scores of 92.16% and 93.32% on the X-IIoTID and WUSTL-IIoT-2021 datasets, respectively, significantly outperforming comparison models such as LR, XGBoost, CNN, and the baseline A3C.

APA3CID has demonstrated strong performance in industrial IoT intrusion detection, but it still has its limitations. First, its performance is quite sensitive to reward design and key hyperparameters, so it may require re-tuning across different scenarios. At present, the system has primarily been validated for scenarios with a limited number of categories; further evaluation is needed to identify more complex, fine-grained multi-class attacks. In addition, we will further explore the practicality of the method described in this paper in continuous deployment industrial IoT environments, with a focus on concept drift detection and adaptive update mechanisms.

Author Contributions: Conceptualization, J.C., H.Y. and J.L.; methodology, J.C., H.Y. and J.L.; software, J.C., H.Y. and J.L.; validation, J.C., H.Y. and J.L.; formal analysis, R.L., W.H. and H.S.; investigation, J.C. and J.L.; resources, H.Y. and S.W.; data curation, R.L. and S.W.; writing—original draft preparation, J.C., H.Y. and J.L.; writing—review and editing, J.C., H.Y. and H.S.; supervision, J.C. and W.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Henan Provincial Department of Science and Technology, Key Research and Development Project (No. 251111210300), the Science and Technology Research Project of Henan Province (No. 252102210177, No. 262102210231), Shenzhen Municipal Science and Technology Innovation Commission Stabilization Support Program (No. 20231128083944001).

Data Availability Statement: The data presented in this study are available on request from the corresponding author due to privacy.

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

References

1. Sisinni, E.; Saifullah, A.; Han, S.; Jennehag, U.; Gidlund, M. Industrial internet of things: Challenges, opportunities, and directions. *IEEE Trans. Ind. Inform.* **2018**, *14*, 4724–4734. [CrossRef]
2. Jamshidi, S.; Nikanjam, A.; Nafi, K.W.; Khomh, F.; Rasta, R. Application of deep reinforcement learning for intrusion detection in Internet of Things: A systematic review. *Internet Things* **2025**, *31*, 101531. [CrossRef]
3. Devendiran, R.; Turukmane, A.V. Dugat-LSTM: Deep learning based network intrusion detection system using chaotic optimization strategy. *Expert Syst. Appl.* **2024**, *245*, 123027. [CrossRef]
4. Zhang, C.; Jia, D.; Wang, L.; Wang, W.; Liu, F.; Yang, A. Comparative research on network intrusion detection methods based on machine learning. *Comput. Secur.* **2022**, *121*, 102861. [CrossRef]
5. Rahman, M.M.; Al Shakil, S.; Mustakim, M.R. A survey on intrusion detection system in IoT networks. *Cyber Secur. Appl.* **2025**, *3*, 100082. [CrossRef]
6. Jaw, E.; Wang, X. Feature selection and ensemble-based intrusion detection system: An efficient and comprehensive approach. *Symmetry* **2021**, *13*, 1764. [CrossRef]
7. Wang, R.; Zhang, Y.; Peng, L.; Fortino, G.; Ho, P.-H. Time-varying-aware network traffic prediction via deep learning in IIoT. *IEEE Trans. Ind. Inform.* **2022**, *18*, 8129–8137. [CrossRef]
8. Talpur, N.; Abdulkadir, S.J.; Hasan, M.H.; Alhussian, H.; Alwadain, A. A Novel Wrapper-Based Optimization Algorithm for the Feature Selection and Classification. *Comput. Mater. Contin.* **2023**, *74*, 5799–5820. [CrossRef]

9. Liu, H.; Zhou, M.C.; Liu, Q. An embedded feature selection method for imbalanced data classification. *IEEE/CAA J. Autom. Sin.* **2019**, *6*, 703–715. [CrossRef]
10. Emmanuel, I.; Sun, Y.; Wang, Z. A machine learning-based credit risk prediction engine system using a stacked classifier and a filter-based feature selection method. *J. Big Data* **2024**, *11*, 23. [CrossRef]
11. Gong, H.; Li, Y.; Zhang, J.; Zhang, B.; Wang, X. A new filter feature selection algorithm for classification task by ensembling pearson correlation coefficient and mutual information. *Eng. Appl. Artif. Intell.* **2024**, *131*, 107865. [CrossRef]
12. Liu, Z.; Thapa, N.; Shaver, A.; Roy, K.; Siddula, M.; Yuan, X.; Yu, A. Using embedded feature selection and CNN for classification on CCD-INID-V1—A new IoT dataset. *Sensors* **2021**, *21*, 4834. [CrossRef]
13. Huang, W.; Tian, H.; Wang, S.; Zhang, C.; Zhang, X. Integration of simulated annealing into pigeon inspired optimizer algorithm for feature selection in network intrusion detection systems. *PeerJ Comput. Sci.* **2024**, *10*, e2176. [CrossRef]
14. Mehedi, S.T.; Anwar, A.; Rahman, Z.; Ahmed, K.; Islam, R. Dependable intrusion detection system for IoT: A deep transfer learning based approach. *IEEE Trans. Ind. Inform.* **2022**, *19*, 1006–1017. [CrossRef]
15. Wazid, M.; Singh, J.; Pandey, C.; Sherratt, R.S.; Das, A.K.; Giri, D.; Park, Y. Explainable deep Learning-Enabled malware attack detection for IoT-Enabled intelligent transportation systems. *IEEE Trans. Intell. Transp. Syst.* **2025**, *26*, 7231–7244. [CrossRef]
16. Attique, D.; Hao, W.; Ping, W.; Javeed, D.; Kumar, P. Explainable and data-efficient deep learning for enhanced attack detection in iiot ecosystem. *IEEE Internet Things J.* **2024**, *11*, 38976–38986. [CrossRef]
17. Bella, K.; Guezzaz, A.; Benkirane, S.; Azrour, M.; Fouad, Y.; Benyeogor, M.S.; Innab, N. An efficient intrusion detection system for IoT security using CNN decision forest. *PeerJ Comput. Sci.* **2024**, *10*, e2290. [CrossRef]
18. Nie, L.; Ning, Z.; Wang, X.; Hu, X.; Cheng, J.; Li, Y. Data-driven intrusion detection for intelligent internet of vehicles: A deep convolutional neural network-based method. *IEEE Trans. Netw. Sci. Eng.* **2020**, *7*, 2219–2230. [CrossRef]
19. Liang, J.; Ma, M.; Tan, X. GaDQN-IDS: A novel self-adaptive IDS for VANETs based on Bayesian game theory and deep reinforcement learning. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 12724–12737. [CrossRef]
20. Duan, Q.; Al-Shaer, E.; Garlan, D. Self-Adaptive Dual-Layer DDoS Mitigation using Autoencoder and Reinforcement Learning. In *2025 IEEE/ACM 20th Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*; IEEE: New York, NY, USA, 2025; pp. 111–121.
21. Yu, S.; Wang, X.; Shen, Y.; Wu, G.; Yu, S.; Shen, S. Novel intrusion detection strategies with optimal hyper parameters for industrial internet of things based on stochastic games and double deep Q-networks. *IEEE Internet Things J.* **2024**, *11*, 29132–29145. [CrossRef]
22. Ren, K.; Zeng, Y.; Zhong, Y.; Sheng, B.; Zhang, Y. MAFSIDS: A reinforcement learning-based intrusion detection model for multi-agent feature selection networks. *J. Big Data* **2023**, *10*, 137. [CrossRef]
23. Huang, W.; Yu, H.; Ren, J.; Wang, K.; Guo, Y.; Jin, L. GSLDWOA: A Feature Selection Algorithm for Intrusion Detection Systems in IIoT. *Comput. Mater. Contin.* **2025**, *86*, 1–24. [CrossRef]
24. Al-Hawawreh, M.; Sitnikova, E.; Aboutorab, N. X-IIoTID: A connectivity-agnostic and device-agnostic intrusion data set for industrial Internet of Things. *IEEE Internet Things J.* **2021**, *9*, 3962–3977. [CrossRef]
25. Ismail, S.; Dandan, S.; Qushou, A. Intrusion Detection in IoT and IIoT: Comparing Lightweight Machine Learning Techniques Using TON_IoT, WUSTL-IIOT-2021, and EdgeIIoTset Datasets. *IEEE Access* **2025**, *13*, 73468–73485. [CrossRef]
26. Salih, A.A.; Abdulazeez, A.M. Evaluation of classification algorithms for intrusion detection system: A review. *J. Soft Comput. Data Min.* **2021**, *2*, 31–40. [CrossRef]
27. Lopez-Martin, M.; Carro, B.; Sanchez-Esguevillas, A. Application of deep reinforcement learning to intrusion detection for supervised problems. *Expert Syst. Appl.* **2020**, *141*, 112963. [CrossRef]
28. Zhou, K.; Wang, W.; Hu, T.; Deng, K. Application of improved asynchronous advantage actor critic reinforcement learning model on anomaly detection. *Entropy* **2021**, *23*, 274. [CrossRef]