Article

Leveraging Swarm Intelligence for Invariant Rule Generation and Anomaly Detection in Industrial Control Systems

Yunkai Song, Huihui Huang, Hongmin Wang, Qiang Wei *

School of Cyberspace Security, Information Engineering University, Zhengzhou 450000, China; 08183119@cumt.edu.cn (Y.S.); huanghui.2513@163.com (H.H.); 13461658194@163.com (H.W.)
* Correspondence: prof_weiqiang@163.com

Abstract: Industrial control systems (ICSs), which are fundamental to the operation of critical infrastructure, face increasingly sophisticated security threats due to the integration of information and operational technologies. Conventional anomaly detection techniques often lack the ability to provide clear explanations for their detection, and their inherent complexity can impede practical implementation in the resource-constrained environments typical of ICSs. To address these challenges, this paper proposes a novel approach that leverages swarm intelligence algorithms for the extraction of numerical association rules, specifically designed for anomaly detection in ICS. The proposed approach is designed to effectively identify and precisely localize anomalies by analyzing the states of sensors and actuators. Experimental validation using the Secure Water Treatment (SWaT) dataset demonstrates that the proposed approach can detect over 84% of attack instances, with precise anomaly localization achievable by examining as few as two to six sensor or actuator states. This significantly improves the efficiency and accuracy of anomaly detection. Furthermore, since the method is based on the general control dynamics of ICSs, it demonstrates robust generalization, making it applicable across a wide range of industrial control systems.

Keywords: industrial control systems; anomaly detection; numerical association rules; swarm intelligence algorithms; security enhancement

Citation: Song, Y.; Huang, H.; Wang, H.; Wei, Q. Leveraging Swarm Intelligence for Invariant Rule Generation and Anomaly Detection in Industrial Control Systems. *Appl. Sci.* **2024**, *14*, 10705. https://doi.org/10.3390/app142210705

Academic Editor: Grigorios Beligiannis

Received: 19 October 2024
Revised: 7 November 2024
Accepted: 13 November 2024
Published: 19 November 2024

1. Introduction

1.1. Background

With the advancement of Industry 4.0, industrial control systems have evolved from closed systems into complex, interconnected frameworks. While this transition has enhanced production efficiency and flexibility, it has also significantly expanded the attack surface [1], allowing for security risks from information technology (IT) to be transmitted to operational technology (OT), thereby directly threatening the safety of industrial facilities. For instance, the "Stuxnet" virus attack on Iran's Bushehr nuclear power plant in 2010 led to the damage of centrifuges [2].

A typical ICS architecture comprises key components including sensors and actuators at the physical layer, Programmable Logic Controllers (PLCs) and Remote Terminal Units (RTUs) at the control layer, and Supervisory Control and Data Acquisition (SCADA) and Human–Machine Interface (HMI) systems at the supervisory control layer. Figure 1 illustrates a generic ICS architecture. In this setup, physical layer devices receive commands through the control layer and provide feedback on physical process states, while the supervisory control layer is responsible for high-level monitoring and decision-making. Anomaly detection mechanisms are typically implemented at this layer, monitoring sensor data and actuator status to ensure the normal operation of the physical processes within the ICS.

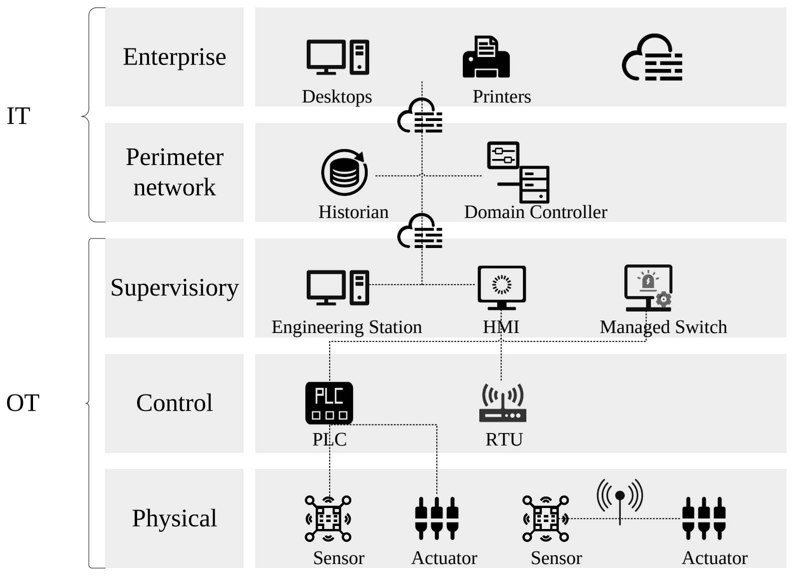

Figure 1. General architecture of ICS.

1.2. Related Work and Problem Statement

Current research on ICS anomaly detection includes statistical analysis, machine learning, and deep learning approaches [3]. Figure 2 illustrates the mainstream research approaches in the field of ICS anomaly detection.

Traditional anomaly detection methods rely on mathematical models constructed by domain experts to reflect the normal behavior of a system, such as entropy-based [4], Kalman filtering [5,6], hidden Markov models [7], and maximum likelihood estimation [8] methods. These methods capture anomalies by analyzing deviations from normal behavior. Ke Liu et al. [4] developed an intrusion detection method based on entropy, using a dynamic threshold adjustment mechanism to predict anomalies. However, these methods often assume that all system states are measurable, an assumption that is difficult to satisfy in real-world scenarios. Additionally, these algorithms depend heavily on expert knowledge, making the model-building process cumbersome and time-consuming.

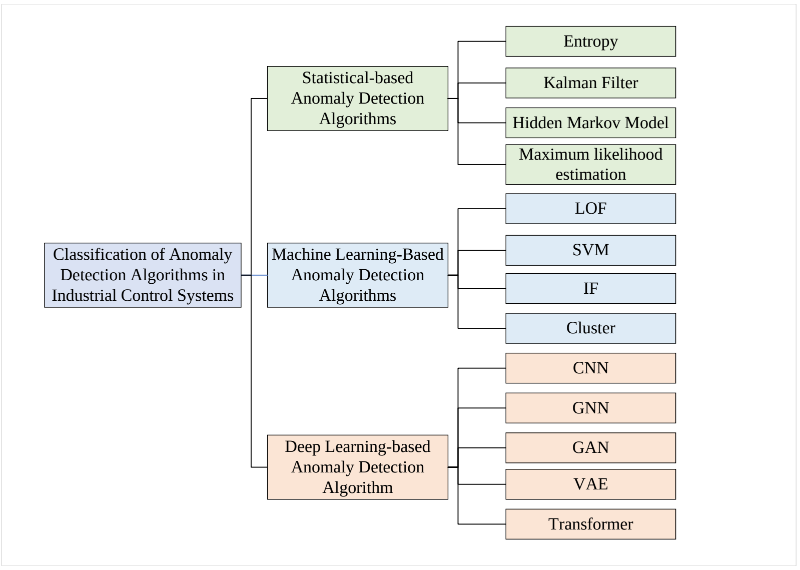
Classification of Anomaly Detection Algorithms in Industrial Control Systems
Statistical-based Anomaly Detection Algorithms
Machine Learning-Based Anomaly Detection Algorithms
Deep Learning-based Anomaly Detection Algorithm
Entropy
Kalman Filter
Hidden Markov Model
Maximum likelihood estimation
LOF
SVM
IF
Cluster
CNN
GNN
GAN
VAE
Transformer

Figure 2. Main research methods for anomaly detection in ICS.

With the rise of machine learning techniques, methods such as Local Outlier Factor (LOF) [9], Isolation Forest (IF) [10,11], Support Vector Machine (SVM) [12], and clustering [13] have shown significant advantages in anomaly detection. These methods do not rely on specific expert knowledge but face challenges when dealing with high-dimensional data, requiring large amounts of labeled data in complex ICSs.

Deep learning models, with their powerful modeling capabilities, have become a hotspot in the anomaly detection domain, including prediction-based and reconstruction-based models. Prediction-based models, such as Convolutional Neural Networks (CNNs) [14,15], Long Short-Term Memory networks (LSTMs) [16], Graph Neural Networks (GNNs) [17,18], and Transformers [19], identify significant prediction errors as anomalies by predicting future states or the next value in sequences. SiET [20] integrates Graph Neural Networks into Anomaly Transformers to detect anomalies by leveraging the high-dimensional correlations between time points. On the other hand, reconstruction-based models, such as Autoencoders (AEs) and Generative Adversarial Networks (GANs), identify anomalies by learning latent representations of data and attempting to reconstruct input data, with larger reconstruction errors indicating the presence of anomalies [21,22]. In [23], the method models normal behavior patterns by learning the graph structures within and between signals, and it detects anomalies using multi-view contrastive learning and adaptive data augmentation within a reconstruction-based approach. MAD-GAN [24] combines GAN and Long Short-Term Memory Recurrent Neural Networks to generate anomaly scores for detection. While deep learning-based anomaly detection methods exhibit superior performance, their inherent black-box nature limits anomaly explanations. In complex ICS, operators may need to investigate numerous sensors to locate the anomaly, impacting system response time. Moreover, ICS often have strict constraints on memory and hardware resources, constraining the direct application of complex models.

Existing deep learning methods based on prediction and reconstruction often struggle to handle "stealth attacks", where attackers gradually inject small, imperceptible errors [25,26]. These attacks manifest as low-frequency and subtle disturbances that are frequently overlooked by traditional anomaly detection methods. However, over time or through cumulative effects, these disturbances can lead to severe system failures. Due to their covert nature, such attacks require more sensitive and precise detection mechanisms. The performance and robustness of current methods in dealing with these attacks still need improvements.

1.3. Contribution

To address these issues, this paper utilizes swarm intelligence algorithms to mine numerical association rules from ICS and applies these rules for anomaly detection. This approach effectively identifies anomalies and highlights deviations from normal patterns, providing clear explanations of anomalies to experts. Consequently, this significantly enhances the diagnostic speed and response efficiency of experts when encountering system anomalies. The content extracted is as follows:

1. This study introduces a numerical association rule method based on swarm intelligence algorithms for anomaly detection in ICSs, demonstrating broad applicability and strong generalization across diverse environments.
2. A feature selection and task partitioning strategy optimizes the search space for swarm intelligence algorithms, enhancing the efficiency of processing large-scale ICS datasets and ensuring high-quality invariant rule generation.
3. An implemented anomaly detection system based on the generated invariants achieves over 84% accuracy on the SWaT dataset, providing precise identification of operational deviations and intuitive explanations for anomalies.

The remainder of this paper is structured as follows: Section 2 provides a brief introduction to numerical association rule mining and swarm intelligence algorithms; Section 3 introduces the methodology design; Section 4 presents the experimental results; and the final section, Section 5, summarizes the paper and outlines future perspectives.

2. Numerical Association Rule Mining and Swarm Intelligence Algorithms

2.1. Numerical Association Rule Mining

Association rule mining [27] aims to uncover hidden relationships between attributes within transactional databases. Traditional association rule mining algorithms [28–30] primarily target discrete attribute data. When dealing with numerical attributes, discretization strategies are often required [31–33], which inevitably leads to information loss. With the advent of swarm intelligence and evolutionary algorithms, methods capable of directly handling both discrete and continuous attributes have emerged, leading to the development of numeric association rule mining (NARM) [34].

Formally, consider an ICS equipped with m sensors and n actuators. Let the data log $\mathcal{D}^{1:T} = \{d^1, d^2, \ldots, d^T\}$, where each record d_t consists of two parts: a state vector $x_t \in \mathbb{R}_m$, representing the readings of all sensors at time t; and a control vector $u_t \in \mathcal{K}_n$, denoting the operational states of the actuators at time t, where $\mathcal{K}_n$ is the set of control action categories. Further, construct an item set $I = \{i_1, i_2, \ldots, i_{m+n}\}$, where each signal $d_t \in \mathcal{D}^{1:T}$ corresponds to a subset of the item set I. An association rule can be defined as an implication relationship, formally expressed as follows:

$$X \Rightarrow Y \tag{1}$$

where X and Y are non-empty and mutually exclusive subsets of I, i.e., $X \subset I \wedge Y \subset I \wedge X \cap Y = \oslash$.

The practical utility of association rules is measured using metrics such as support, confidence, interest, and comprehensibility. Support reflects the frequency with which (X) and (Y) co-occur, defined as follows:

$$supp(X \Rightarrow Y) = \frac{n(X \cap Y)}{|D|} \tag{2}$$

where $n(X \cap Y)$ denotes the number of transactions that contain both X and Y, and $|D|$ represents the total number of transactions in the dataset D.

Confidence quantifies the probability of Y occurring given X, defined as follows:

$$conf(X \Rightarrow Y) = \frac{n(X \cap Y)}{n(X)} \tag{3}$$

A confidence of 1 indicates that Y always occurs if X occurs. Strong association rules generally require both high support and high confidence.

Interestingness measures the novelty and importance of the rule, defined as follows:

$$inte(X \Rightarrow Y) = \frac{supp(X \Rightarrow Y)}{supp(X)} \cdot \frac{supp(X \Rightarrow Y)}{supp(Y)} \cdot \left(1 - \frac{supp(X \Rightarrow Y)}{|D|}\right) \tag{4}$$

Comprehensibility assesses the intuitiveness and understandability of the rule by quantifying the simplicity of the antecedent and consequent, defined as follows:

$$comp(X \Rightarrow Y) = \frac{log(1 + |Y|)}{log(1 + |X \cup Y|)} \tag{5}$$

Invariant rules can be defined as strong association rules. In the context of ICSs, they reflect the essential characteristics of a normal system operation, manifesting as patterns of state transitions among sensors and actuators. Any physical process violating these rules would be considered an anomaly, formally represented as follows:

$$I_1 \Rightarrow I_2 \text{ where } I_1, I_2 \subset D \wedge I_1 \cap I_2 = \varnothing \wedge \frac{\sup(I_1 \cup I_2)}{\sup(I_1)} \geq 1 \tag{6}$$

In which I_1 and I_2 are the antecedent and consequent of the rule, respectively, and their relationship strictly meets specific conditions, that is, in all data logs, whenever I_1 occurs,

I_2 must necessarily occur, ensuring that the rules with strong associative relationships are selected, which can minimize the risk of false positives to the greatest extent.

2.2. *Swarm Intelligence-Based Algorithms*

Swarm intelligence [35] represents a computational paradigm inspired by the collective behavior of simple entities in nature. These algorithms simulate the complex group behaviors that emerge from local interactions among simple individuals. Typical algorithms include Differential Evolution (DE) [36], Particle Swarm Optimization (PSO) [37], and Bat Algorithms (BA) [38]. The core concept of swarm intelligence algorithms is to guide the collective behavior of numerous simple agents (referred to as "agents" or "particles") through basic rules to address complex optimization problems.

Swarm intelligence algorithms focus on the emergent behaviors of multiple interacting agents following simple rules. While each agent may be considered non-intelligent on its own, the collective system can exhibit self-organizing behavior, functioning akin to a form of collective intelligence.

Typically, swarm intelligence algorithms consist of the following key components:

- **Agents:** Agents are the basic units of the algorithm, moving within the search space and interacting based on the state of the environment and other agents.
- **Search Space:** This is the solution space explored by the algorithm, within which agents move to find optimal solutions.
- **Variation Operators:** These operators are used to update the state of the agents, which can be rule-based or equation-based.
- **Evaluation Function:** Used to evaluate the fitness of the agents, determining which agents survive to the next iteration.
- **Self-Organization:** Interactions among agents lead to self-organizing behavior, guiding the population toward the optimization objective.

By simulating the interactions among simple individuals, swarm intelligence algorithms demonstrate powerful capabilities in solving complex problems. Their characteristics of self-organization, parallelism, and adaptability make them effective tools for addressing various optimization challenges. A general pseudocode for swarm intelligence algorithms is provided in Algorithm 1.

Algorithm 1 General Pseudocode for Swarm Intelligence Algorithms

```
1:  procedure EnhancedSwarmIntelligence
2:      Initialization:
3:          Set the search space $S$ (the area in which agents search for solutions)
4:          Initialize the population of agents agents (each agent has an initial position within $S$)
5:          Define the fitness evaluation function $f(\cdot)$ (assesses the quality of a solution)
6:      while termination condition is not met do
7:          for each agent $\mathbf{agent}_i$ in agents do
8:              Compute the fitness value $f(\mathbf{position}_i)$ of the agent $\mathbf{agent}_i$'s position $\mathbf{position}_i$
9:          end for
10:         for each agent $\mathbf{agent}_i$ in agents do
11:             Update the agent's position $\mathbf{position}_i$ using variation operators
12:             Consider the influence of neighboring agents $\mathbf{neighbors}_i$ when updating $\mathbf{position}_i$
13:         end for
14:         Select the most fit agents based on their fitness values for the next iteration
15:         Record the best solution found so far and its corresponding fitness value
16:     end while
17:     Output the best solution and its fitness value
18: end procedure
```

3. Methodology

In this section, we present a framework for systematically deriving invariant rules from ICS data logs and utilizing them for anomaly detection. Our proposed method addresses the limitations of existing approaches by integrating swarm intelligence algorithms with advanced feature selection and task segmentation techniques. The framework consists of two primary components:

1. Generating Invariant Rules Using Swarm Intelligence Algorithms: We employ swarm intelligence algorithms to efficiently extract invariant rules from ICS data logs. These algorithms are powerful tools for solving optimization problems and allow us to uncover meaningful patterns and relationships from large datasets.
2. Utilizing Invariant Rules for Anomaly Detection in ICS: Once the invariant rules are generated, they are used for anomaly detection in ICS. This section details the application of these rules to analyze ICS data, enabling the timely identification and response to deviations from normal operational patterns.

The design of the anomaly detection model based on association rules is illustrated in Figure 3.

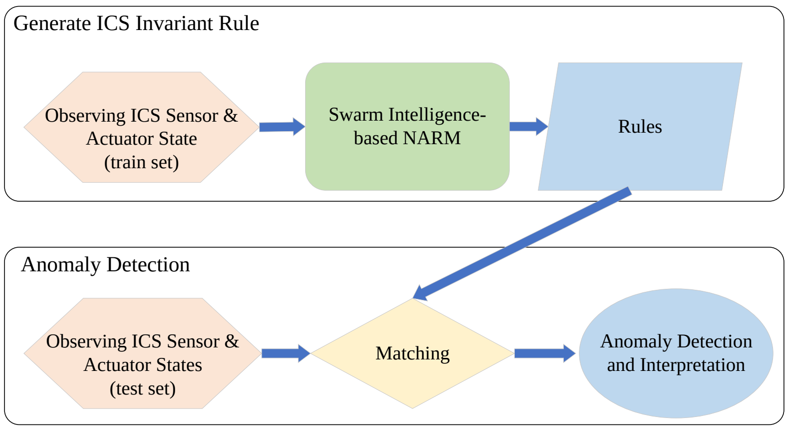
Generate ICS Invariant Rule
Observing ICS Sensor & Actuator State (train set)
Swarm Intelligence-based NARM
Rules
Anomaly Detection
Observing ICS Sensor & Actuator States (test set)
Matching
Anomaly Detection and Interpretation

Figure 3. Anomaly detection design.

To clearly present our contributions, we detail the proposed method in the following subsections.

3.1. Generating Invariant Rules Using Swarm Intelligence Algorithms

To efficiently extract invariant rules from ICS data logs, we employ swarm intelligence algorithms. These algorithms are powerful tools for solving optimization problems, and in the context of ICS, they allow us to uncover meaningful patterns and relationships from large datasets. The process of generating invariant rules is modeled as a single-objective, continuous optimization task, which involves optimizing three key components:

- **Feature Selection and Task Partitioning:** To improve efficiency and avoid inefficiency due to a large search space, we use feature selection through correlation analysis and task partitioning. This strategy refines the search process by breaking down the optimization task into more manageable sub-tasks, enhancing search efficiency.
- **Solutions Representation in the Search Space:** Solutions are encoded as real-valued vectors in the search space. A genotype/phenotype mapping mechanism transforms these encoded solutions into the phenotype space, which better represents the actual problem.
- **Fitness Function Evaluation:** A fitness function in the phenotype space evaluates the quality of the solutions, guiding the optimization process.

The flowchart for generating invariant rules using swarm intelligence algorithms is shown in Figure 4.

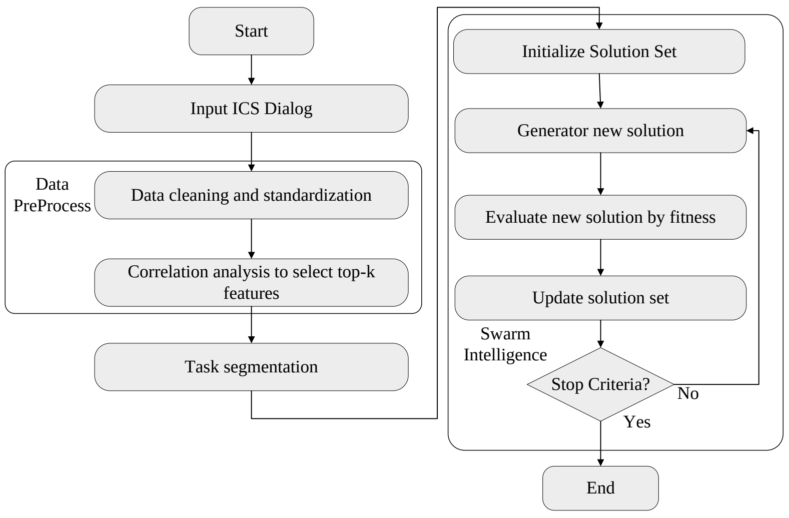

Figure 4. Flowchart of invariant rule generation based on swarm intelligence algorithms.

3.1.1. Feature Selection and Task Segmentation

Given the large scale and complexity of ICS datasets, as well as the challenges associated with the extensive solution space encountered by traditional swarm intelligence algorithms in numeric association rule mining, we employ feature selection and task segmentation. Specifically, each feature is treated as a target feature, and machine learning algorithms are utilized to identify features that are highly correlated with the target feature. This approach decomposes the large-scale mining task into multiple smaller, more manageable sub-tasks, thereby enhancing both computational efficiency and interpretability of the algorithm.

Let $F = \{F_1, F_2, \ldots, F_p\}$ be the set of all features, where p is the total number of features. The strength of the correlation between features is calculated using the Pearson correlation coefficient:

$$r_{F_i,F_j} = \frac{\sum_{k=1}^{n} (x_{ik} - \bar{x}_i)\left(x_{jk} - \bar{x}_j\right)}{\sqrt{\sum_{k=1}^{n} (x_{ik} - \bar{x}_i)^2}\sqrt{\sum_{k=1}^{n} \left(x_{jk} - \bar{x}_j\right)^2}} \tag{7}$$

where $r_{F_iF_j}$ represents the correlation coefficient between feature F_i and F_j, x_{ik} is the value of feature F_i in the k-th data record, $\bar{x}_i$ is the mean value of feature F_i, and n is the number of data records.

To ensure that the segmented sub-tasks possess clear correlations, a threshold θ must be defined to select features F_j that exhibit a significant correlation with the target feature F_i. Specifically, only features F_j for which the following condition holds:

$$\{F_j \mid |r_{F_i,F_j}| > \theta, F_i \neq F_j\} \tag{8}$$

are considered to have a strong correlation with F_i and are included in further analysis.

Having selected the set of features correlated with F_i, the original invariant rule mining task T can be divided into several smaller tasks T_k, each involving a subset of correlated features:

$$T_i = \{F_i, F_{j_1}, F_{j_2}, \ldots, F_{j_l}\} \tag{9}$$

where l is the number of selected features, and $l < p$. This method allows us to effectively reduce the complexity of the mining task by focusing on the most relevant features, thus improving the efficiency and accuracy of the mining process and ensuring that the mined rules have a higher explanation and generalization capability.

To illustrate this process, we provide Algorithm 2, which details the steps involved in feature selection and task segmentation.

In Algorithm 2, the input is the set of all features F and the correlation threshold θ. The output is a list of sub-tasks T, where each sub-task contains a set of highly correlated features. The algorithm first initializes a correlation matrix R and then calculates the Pearson correlation coefficients for all pairs of features. It then selects features that are highly correlated with each target feature and forms sub-tasks based on these correlated features. This approach ensures that the sub-tasks are manageable and computationally efficient.

Algorithm 2 Feature Selection and Task Segmentation

```
Require:
 1: $F$: Set of all features $\{F_1, F_2, \dots, F_p\}$, where $p$ is the total number of features.
 2: $\theta$: Correlation threshold to filter highly correlated features.
Ensure:
 3: $T$: List of sub-tasks, where each sub-task contains a set of highly correlated features.
 4: $p \leftarrow \text{length}(F)$                         ▷ Obtain the total number of features
 5: $R \leftarrow \text{zeros}(p, p)$                      ▷ Initialize the correlation matrix with zeros.
 6: for $i = 1$ to $p$ do
 7:     for $j = i$ to $p$ do
 8:         $R[i][j] \leftarrow \text{PearsonCorrelation}(F[i], F[j])$
 9:         $R[j][i] \leftarrow R[i][j]$                    ▷ The correlation matrix is symmetric.
10:     end for
11: end for
12: $T \leftarrow []$                                  ▷ Initialize an empty list to store sub-tasks.
13: for $i = 1$ to $p$ do
14:     $correlated\_features \leftarrow [F[i]]$            ▷ Start with the target feature.
15:     for $j = 1$ to $p$ do
16:         if $i \neq j$ and $|R[i][j]| > \theta$ then
17:             $correlated\_features.\text{append}(F[j])$   ▷ Add feature $F_j$ if it is highly correlated with $F_i$.
18:         end if
19:     end for
20:     $T.\text{append}(correlated\_features)$             ▷ Add the sub-task for $F_i$ to the list.
21: end for
22: return $T$                                         ▷ Return the list of sub-tasks.
```

3.1.2. Representation of Solutions and Numerical Association Rule Generation

In the representation of solutions within the search space of swarm intelligence algorithms, individual solutions appear as real-valued vectors, where t denotes the iteration number, and i is the index of the solution. The structure of the vector is as follows:

$$\mathbf{x}_i^{(t)} = (\dots, \underbrace{\left\langle x_{i,j,1}^{(t)}, x_{i,j,2}^{(t)}, x_{i,j,3}^{(t)} \right\rangle}_{Attr_{i,j}^{(num)}}, \dots, \underbrace{\left\langle x_{i,j',1}^{(t)}, x_{i,j',2}^{(t)} \right\rangle}_{Attr_{i,j'}^{(cat)}}, \dots) \tag{10}$$

Here, j and j' represent the j-th numerical attribute and the j'-th categorical attribute, respectively. n indicates the number of numerical attributes, and m is the number of categorical attributes.

Each numerical attribute $Attr_{i,j}^{(num)}$ is represented by a triple $\left\langle x_{i,j,1}^{(t)}, x_{i,j,2}^{(t)}, x_{i,j,3}^{(t)} \right\rangle$, where $x_{i,j,1}^{(t)}$ and $x_{i,j,2}^{(t)}$ denote the lower and upper bounds of the specific attribute values, and $x_{i,j,3}^{(t)}$ is a threshold indicating whether the feature is included in the rule.

The numerical attributes are calculated according to the following formula:

$$Attr_{i,j}^{(num)} = \begin{cases} NULL & \text{, if } rand(0,1) < x_{i,j,3}^{(t)} \\ [x_{i,j,1}^{(t)}, x_{i,j,2}^{(t)}] & \text{, otherwise} \end{cases} \tag{11}$$

Categorical attributes are simplified to a 2-tuple $\left\langle x_{i,j',1}^{(t)}, x_{i,j',2}^{(t)} \right\rangle$, including the specific candidate value $x_{i,j',1}^{(t)}$, the threshold $x_{i,j',2}^{(t)}$ indicating whether the feature is included in the rule.

The categorical attributes are calculated according to the following formula:

$$Attr_{i,j'}^{(cat)} = \begin{cases} NULL & \text{, if } rand(0,1) < x_{i,j',2}^{(t)} \\ x_{i,j',1}^{(t)} & \text{, otherwise} \end{cases} \tag{12}$$

According to the mapping rules of the solution vector, the antecedent and consequent are generated to form the rule $Ante \Rightarrow Conse$. The iterative process continues to optimize the solution vector and generate more high-quality invariant rules.

3.1.3. Fitness Function

In the process of generating invariant rules, the evaluation of the fitness function is a crucial step in determining the quality of the invariant rules [34]. The design of the fitness function directly impacts the performance of the algorithm and the quality of the final mined invariant rules. To comprehensively evaluate the invariant rules, the fitness function is typically constructed based on several metrics, including support, confidence, comprehensibility, and interestingness. Here, we use a weighted sum of support, confidence, and comprehensibility metrics:

$$f\left(\mathbf{x}_i^{(t)}\right) = \frac{\alpha \cdot \text{supp}\left(\mathbf{x}_i^{(t)}\right) + \beta \cdot \text{conf}\left(\mathbf{x}_i^{(t)}\right) + \gamma \cdot \text{comp}\left(\mathbf{x}_i^{(t)}\right)}{\alpha + \beta + \gamma} \tag{13}$$

where α, β, and γ are the respective weights. The objective is to maximize the fitness function.

The fitness function is designed to incorporate multiple metrics such as support, confidence, and comprehensibility. By maximizing the fitness function, the algorithm ensures the generation of high-quality invariant rules, which are essential for effective rule mining in ICS datasets.

3.2. Utilizing Invariant Rules for Anomaly Detection in ICS

After generating invariant rules using swarm intelligence algorithms, the next task is to apply these rules for real-time monitoring of an ICS to achieve effective anomaly detection. This section details the application of extracted invariant rules to analyze ICS data, enabling the timely identification and response to deviations from normal operational patterns, thus ensuring system stability and safety.

SCADA continuously collects data from various sensors and actuators within the ICS, including critical operational parameters such as pressure, temperature, liquid level, flow rate, and valve statuses. These data are compared against the set of invariant rules generated in the previous section. Each invariant rule is considered a physical law or state constraint that the system should adhere to under normal conditions. Any deviation from these invariant rules is flagged as an anomaly.

The method of anomaly detection based on invariant rules, due to its clarity and explanation, enables operators to quickly understand changes in the system's state and respond promptly. By comparing real-time data collected by the SCADA system with these invariant rules, this approach facilitates early detection and localization of anomalies, reducing the time required for fault diagnosis. Additionally, its ability to clearly distinguish between normal and abnormal states enhances the overall safety of the system, ensuring the continuity and reliability of industrial processes.

4. Experiments

The experiments were conducted on a personal computer equipped with the hardware and software configurations listed in Table 1. All experimental code was written in Python 3.9 and executed on the Windows 11 Pro 64-bit operating system. Data processing and feature extraction were performed using the Pandas and NumPy libraries, while visualization results were generated using the Matplotlib library. The implementation and optimization of swarm intelligence algorithms were carried out using the NiaARM and NiaPy libraries.

Table 1. Experimental setup and environment.

Component	Specification
CPU	Intel(R) Core(TM) i7-9700F @ 3.00GHz (Intel, Santa Clara, CA, USA)
RAM	32 GB
Operating System	Windows 11 Pro 64-bit
Programming Language	Python 3.9
Data Processing Libraries	Pandas 2.1.3
	NumPy 1.26.2
Visualization Library	Matplotlib 3.5.3
Swarm Intelligence Libraries	NiaARM 0.3.5
	NiaPy 2.0.5

4.1. Dataset

This experiment was conducted using the SWaT dataset [39]. The SWaT dataset originates from a downscaled model testbed designed to simulate a real-world water treatment plant, encompassing six critical stages of water treatment. The dataset comprises 11 days of operational data, with the first 7 days representing normal operation conditions collected in a physically isolated environment. The subsequent 4 days feature 41 distinct cyber-attack scenarios, which are categorized into three main types:

- **Single Stage Single Point Attacks:** These attacks target a single point within one stage of the water treatment process.
- **Single Stage Multi-Point Attacks:** These attacks involve multiple points within a single stage, affecting several components or sensors simultaneously.
- **Multi-Stage Single Point Attacks:** These attacks target a single point across multiple stages of the water treatment process.
- **Multi-Stage Multi-Point Attacks:** These attacks span multiple stages of the water treatment process, impacting various components and sensors across different stages.

Out of the 41 attack scenarios, 36 resulted in tangible physical impacts on the system. Data were recorded at a frequency of one sample per second, encompassing sensor readings from 25 continuous attributes and actuator states from 26 discrete attributes. The architecture of the SWaT testbed is illustrated in Figure 5.

For this experiment, the training set consisted of the normal operational data from the first seven days, capturing the system's typical operational conditions. The test set was derived from the data of the following four days, featuring a range of cyber-attack scenarios, to evaluate the model's performance and detection accuracy under various attack conditions.

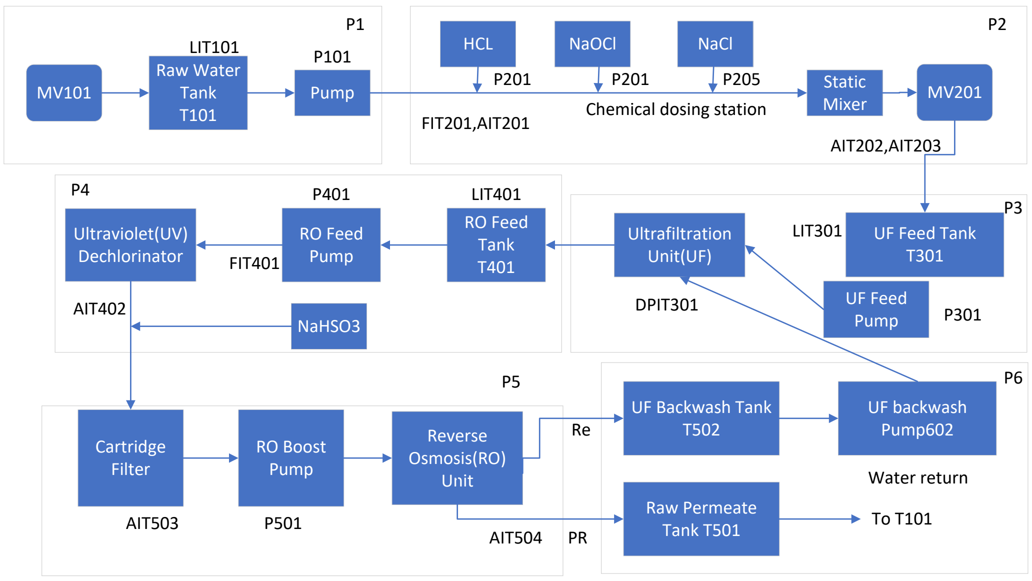

Figure 5. The figure illustrates the architecture of SWaT, which is designed to simulate a real-world water treatment plant. The testbed consists of six critical stages of water treatment, each with multiple sensors and actuators [39].

4.2. Numeric Association Rules Mining

4.2.1. Feature Selection and Task Segmentation

In the SWaT dataset, Pearson correlation coefficients between all features were computed using the method described in Section 3.1.1. The results are shown in Figure 6. Certain features exhibit a high degree of correlation; for instance, the Pearson correlation coefficient between the flow sensor FIT101 and the valve MV101 is as high as 97.12%. This strong dependency indicates that the valve MV101 regulates the inflow to the raw water tank, which is consistent with the interdependencies between sensors and actuators in industrial control systems environments.

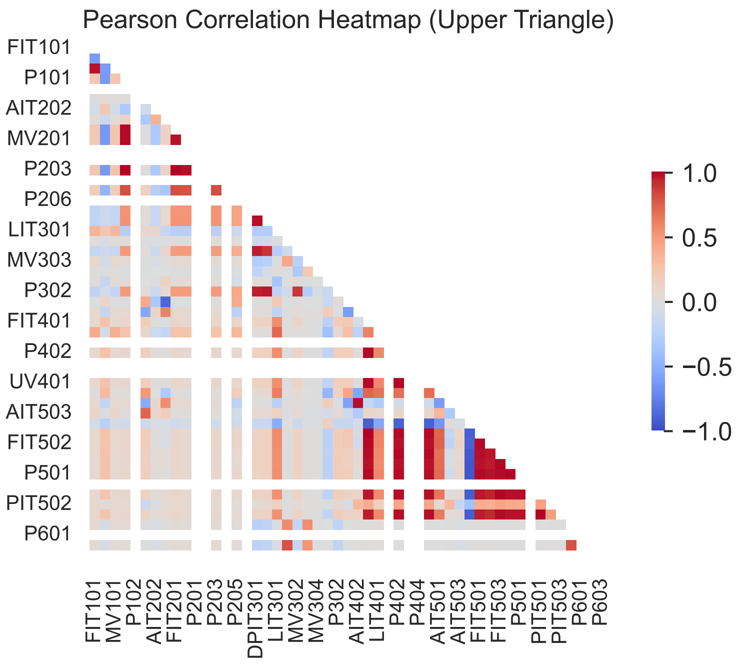

Figure 6. Correlation relationships between features.

To reduce the search space for swarm intelligence algorithms in mining invariant rules for ICSs and enhance search efficiency, the mining task was decomposed into smaller tasks based on feature correlations. To determine an appropriate threshold for this decomposition, we conducted an experiment where the Pearson correlation coefficient threshold, denoted by θ, was varied from 0 to 0.9 in increments of 0.1. The evaluation metrics used in this experiment were the number of rules mined and the time cost of the mining process.

The experimental results, illustrated in Figure 7, show that when $\theta = 0.3$, the number of rules mined is maximized while the time cost remains relatively low. This suggests that setting $\theta = 0.3$ strikes a balance between the comprehensiveness of the rule set and computational efficiency. Therefore, for each feature, its correlation with other features was calculated, and only those with a correlation coefficient greater than $\theta = 0.3$ with the target feature were retained. This approach enabled the large-scale mining task to be broken down into multiple smaller tasks, each focusing on different feature combinations, with each sub-task targeting a group of highly correlated features.

By adopting this strategy, we not only reduced the complexity of the problem but also ensured that the rules mined are more likely to capture meaningful and relevant patterns within the data, thereby improving the overall performance and interpretability of the model.

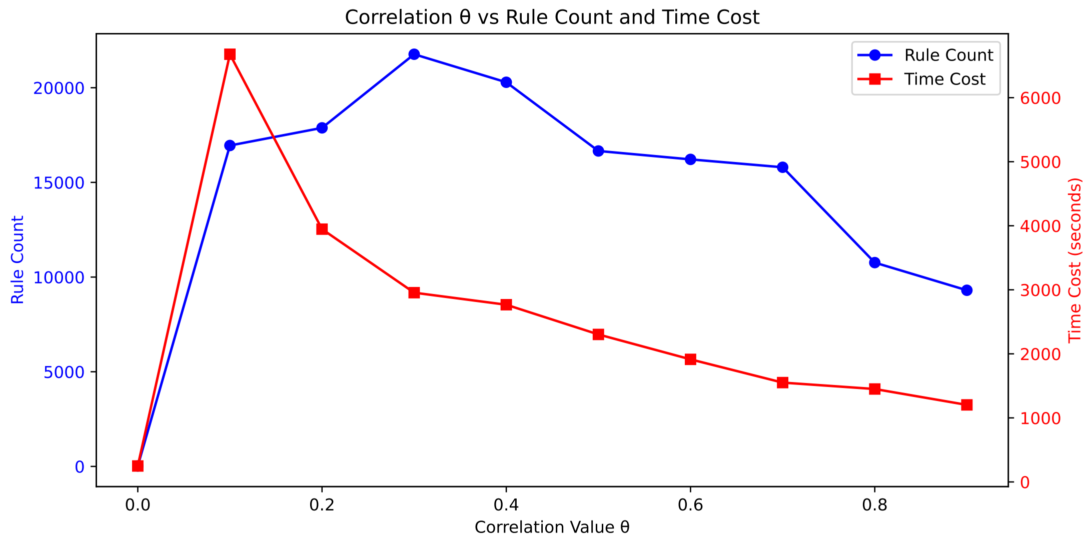

Figure 7. Effect of pearson correlation coefficient threshold (θ) on the number of rules mined and time cost.

4.2.2. Selection and Parameters Tuning of Swarm Intelligence Algorithms

In this study, we utilized three different swarm intelligence algorithms: DE [36], PSO [37], and BA [38]. All swarm intelligence algorithms in this study were implemented using the NiaPy [40] framework. This section aims to identify the most suitable swarm intelligence algorithm for invariant rule mining in ICS scenarios through parameter optimization.

PSO, DE, and BA are metaheuristic algorithms that depend on predefined parameters, and tuning parameters is crucial for algorithm performance. To determine the optimal parameter settings, we adopted the General Factorial Design method for systematic parameters tuning. The parameter settings for the PSO algorithm were set by prior research [41–43], ensuring a scientific and effective range of parameter. The parameter values for the BA and DE were derived from the default settings in the NiaPy framework and related studies [44,45]. The factors and levels are listed in Table 2.

Considering the inherent randomness of these algorithms, to ensure robust results, each parameter combination was independently run 20 times. The average results served as the basis for performance evaluation, reducing the impact of random factors. Ultimately, through a comprehensive analysis of the experimental data, Table 3 presents the optimal parameter combinations for each algorithm, which demonstrates the best rule mining performance across multiple tests.

Table 2. Parameter settings for different optimization methods.

Method	Factor	Level 1	Level 2	Level 3
PSO	w	0.4	0.8	–
	C_1	0.5	1	2
	C_2	0.5	1	2
	Population	20	30	50
		50	100	500
	Iteration	50	100	500
		50	100	500
DE	F	0.5	0.8	1
	CR	0.7	0.8	0.9
	Population	20	30	50
		50	100	500
	Iteration	50	100	500
		50	100	500
BA	Loudness	1	0.9	0.8
	Pulse Rate	1	0.9	0.8
	α	0.7	0.8	0.9
	γ	0.1	0.2	0.3
	Population	20	30	50
		50	100	500
	Iteration	50	100	500
		50	100	500

Table 3. Best parameter setting.

Method	Factor	Value
PSO	w	0.8
	C_1	2
	C_2	2
DE	F	1
	CR	0.8
BA	Loudness	1
	Pulse rate	1
	α	0.9
	γ	0.1
PSO, DE, BA	population	50
	Iteration	50

4.2.3. Results and Evaluation of Invariant Rule Generation

To comprehensively assess the performance of different swarm intelligence algorithms in the task of numerical association rule mining, we evaluated them using metrics such as fitness, support, interestingness, comprehensibility, and the number of rules generated. Table 4 presents the results of these evaluation metrics for each swarm intelligence algorithm under their optimal hyperparameter configurations, providing quantitative evidence for comparative analysis.

As shown in Table 4, PSO outperformed BA and DE in most metrics. Specifically, PSO achieved the highest average fitness (0.8573), support (0.7102), and interestingness (0.7468). These superior results can be attributed to PSO's global search capability and rapid convergence, which enable it to explore the solution space effectively and avoid local optima. The high support and interestingness indicate that the rules generated by PSO are both frequent and novel, reflecting the true behavior of the system.

Table 4. Evaluation metrics of swarm intelligence algorithms under optimal hyperparameters.

Algorithm	Avg (Fitness)	Avg (Support)	Avg (Interestingness)	Avg (Comprehensibility)	Total Rules
BA	0.7879	0.5814	0.5749	0.5417	22,847
DE	0.7403	0.4856	0.5064	0.5410	20,958
PSO	0.8573	0.7102	0.7468	0.5866	21,075

While PSO's comprehensibility (0.5866) was slightly lower than BA's (0.5417) but higher than DE's (0.5410), it still generated a reasonable number of rules (21,075), balancing quality and quantity. This suggests that PSO can produce high-quality, interpretable rules without generating excessive redundancy.

In contrast, BA performed well in fitness and support but was less effective in interestingness and comprehensibility. DE had relatively lower performance in all metrics except comprehensibility. These differences can be attributed to the distinct search mechanisms and optimization strategies of BA and DE.

Overall, PSO demonstrated superior performance in numerical association rule mining, particularly in terms of fitness, support, and interestingness. Its global search capability and rapid convergence make it a preferred choice for this task.

4.3. Anomaly Detection and Explanation

4.3.1. Evaluation Metrics for Anomaly Detection

Due to the issue of class imbalance in ICS datasets, relying solely on accuracy as an evaluation metric may be insufficient. Class imbalance refers to the disparity in sample sizes across different classes, which can lead to models achieving high accuracy simply by predicting the majority class while neglecting the performance on minority classes. Therefore, this study employs a range of evaluation metrics to provide a comprehensive assessment, including accuracy, precision, recall, and F1-score.

Accuracy

$$\text{Accuracy} = \frac{TP + TN}{TP + FN + TN + FP} \tag{14}$$

Accuracy is defined as the proportion of correctly predicted instances out of the total number of instances. While accuracy is an intuitive and commonly used metric, it may not be the best choice when dealing with class imbalance, as a model can achieve high accuracy by predominantly predicting the majority class.

Recall

$$\text{Recall} = \frac{TP}{TP + FN} \tag{15}$$

Recall, also known as sensitivity or true positive rate, is the ratio of true positives (TP) to the total number of actual positives (TP + False Negatives, FN). In the context of ICS monitoring, a high recall indicates the effective identification of most true anomalies, which is crucial for ensuring system security.

Precision

$$\text{Precision} = \frac{TP}{TP + FP} \tag{16}$$

Precision, also known as positive predictive value, is the ratio of true positives (TP) to the total number of predicted positives (TP + False Positives, FP). In ICS scenarios, high precision ensures that when the model predicts an event as anomalous, the prediction is highly reliable, thereby reducing the impact of false alarms.

F1-Score

$$\text{F1-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{17}$$

The F1-score is the harmonic mean of precision and recall, providing a balanced measure of these two aspects. In ICS monitoring, the F1-score helps find a reasonable trade-off between the accuracy of anomaly detection and its comprehensiveness.

By employing these metrics, we can provide a more comprehensive evaluation of the model's performance in handling ICS data, particularly in detecting rare but critical anomalies. Additionally, this multi-dimensional evaluation approach helps reveal potential biases or weaknesses in the model, guiding further improvements.

4.3.2. Anomaly Detection Results

To validate the effectiveness of the discovered invariant rules in the anomaly detection task for ICSs, we evaluated the invariant rules on the test set. The evaluation employed recall, precision, and F1-score as primary metrics to comprehensively measure the detection performance of different algorithms. The specific evaluation results are presented in Table 5.

Table 5. Evaluation of invariant rule anomaly detection performance.

Algorithm	Accuracy	Recall	Precision	F1-Score
BA	0.9782	0.8239	0.9795	0.8950
DE	0.9811	0.8676	0.9603	0.9116
PSO	0.9804	0.8457	0.9773	0.9067

The evaluation results indicate that the DE algorithm excels in recall and F1-score due to its superior global search capability and local optimization performance, outperforming both the BA and PSO algorithm. However, in terms of precision, the PSO algorithm demonstrates slightly better performance.

Table 6 illustrates several typical invariant rules mined using the PSO algorithm. The recall and precision values indicate the detection effectiveness of the rules for specific attack types. These rules not only exhibit high predictive accuracy but also possess excellent interpretability, facilitating rapid identification of potential anomaly patterns by experts. In practice, utilizing multiple invariant rules in conjunction can yield improved detection results.

Table 6. Typical invariant rules mined using the particle swarm optimization algorithm.

Rule No.	Antecedent	Consequent	Attack Type	Recall	Precision
1	$167.305 < \text{LIT101} < 650.699$ and P101 = ON	P102 = OFF	1	0.8978	1
2	$246.053 < \text{AIT201} < 272.272$ and $229.611 < \text{LIT101} < 678.163$	P102 = OFF	2	0.8961	1
3	$153.781 < \text{AIT402} < 165.234$ and $262.233 < \text{AIT503} < 281.705$	$413.039 < \text{LIT301} < 1014.724$	7	0.9463	1
4	MV302 = OFF and $0.0 < \text{FIT301} < 2.359$ and MV304 = ON	$0.0 < \text{DPIT301} < 21.0993$	8	0.4917	1
5	$329.519 < \text{AIT203} < 336.848$ and $0.663 < \text{FIT501} < 1.707$	P501 = ON	22	0.8509	1

4.4. Comparison with Other Methods

To validate the effectiveness of our approach, we compared it against several representative methods recently applied to the SWaT dataset using the same evaluation metrics. Table 7 summarizes the comparison, which includes traditional baseline methods [9–11], deep learning-based baselines [19–21,23], and an invariant rule-based baseline [32]. Some experimental results are sourced from SiET [20]. For this comparison, we used the hyperparameters specified in the respective papers for each baseline model.

Table 7. Detection performance of different methods on the SWaT dataset.

Method	Precision	Recall	F1-Score
LOF	0.7544	0.6290	0.6860
IF	0.4929	0.4495	0.4702
DIF	0.935	0.835	0.882
Anomaly Transformer	0.8935	0.9292	0.9110
TranAD	0.976	0.6997	0.8151
MAD-GAN	0.9897	0.6374	0.7754
SiET	0.9258	0.9705	0.9476
Invariant rule	0.974	0.7657	0.8578
BA (ours)	0.9795	0.8239	0.8950
DE (ours)	0.9603	0.8676	0.9116
PSO (ours)	0.9773	0.8457	0.9067

As shown in Table 7, while our method exhibits slightly lower performance in recall and F1-score compared to some deep learning-based methods (e.g., SiET), it maintains high precision. This indicates that our method effectively reduces false positives while still identifying most anomalies.

Although deep learning-based anomaly detection methods excel in recall and F1-score, their black-box nature poses significant limitations in practical applications. Specifically, these models can detect anomalies but fail to provide specific reasons for the anomalies. As a result, operators must manually inspect numerous sensor or actuator states to identify potential sources of anomalies, which is both time-consuming and complex.

In contrast, our method not only accurately detects anomalies but also provides intuitive explanations. Once an anomaly is detected, our method can quickly pinpoint the specific cause and detail how it deviates from normal patterns. For instance, by analyzing a small number of key sensors (typically 2–6) or actuators, operators can rapidly determine the exact location and cause of the anomaly. This capability significantly enhances the efficiency of fault diagnosis, reducing the time and effort required to identify and address issues, thus ensuring the safe and stable operation of the system.

For example, one of the typical invariant rules we mined using the PSO algorithm is $810.1662341107707 < LIT101 < 1000.0$ and P101=ON $\Rightarrow$ MV101=OFF. Through the system's feedback mechanism, this rule eventually causes MV101 to close. Deviations from this established pattern may indicate an attack or a fault. In the SWaT system, network attacks on MV101 can be effectively identified by this rule. Specifically, in the test set, single stage single point attacks on MV101 are detected with high accuracy, as shown in Figure 8. The recall rate for this rule in detecting the single stage single point attack reached 89.78%, with a precision of 1.0. Furthermore, combining this rule with two other invariant rules for detecting the attack can improve the recall rate to 98.14%. Similarly, the invariant rules we discovered showed good detection performance for single stage multi-point attacks, multi-stage single point attacks, and multi-stage multi-point attacks. Figures 9, 10 and 11 illustrate the detection results for these attack types, respectively.

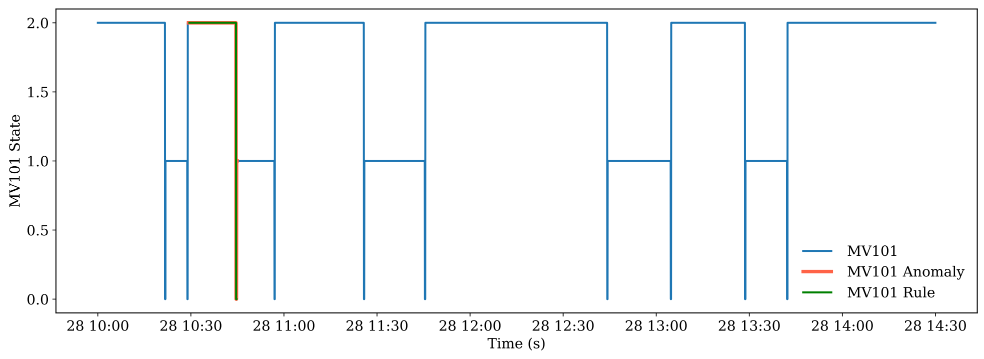

Figure 8. Detection performance of the invariant rule for single stage single point attacks on MV101. The figure shows the accuracy and recall rate in detecting anomalies, with a recall rate of 89.78% and a precision of 1.0.

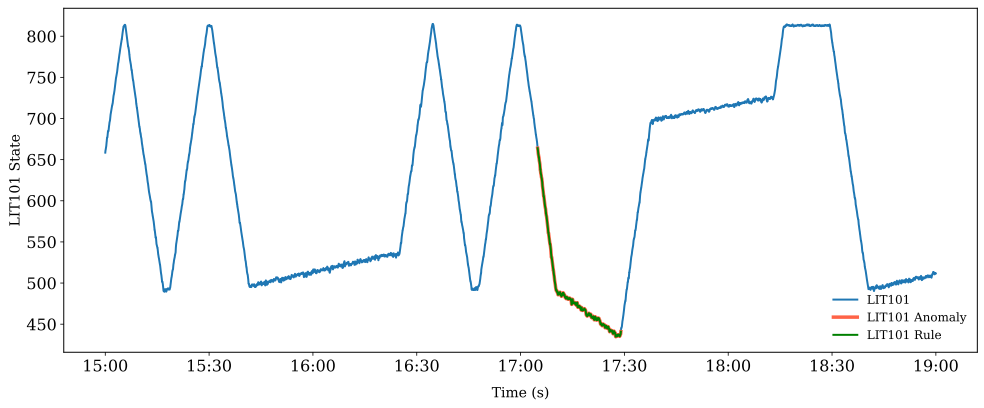

Figure 9. Detection performance of the invariant rules for multi-stage single point attacks on LIT101 and LIT301. The figure demonstrates the rule's ability to detect anomalies across multiple stages.

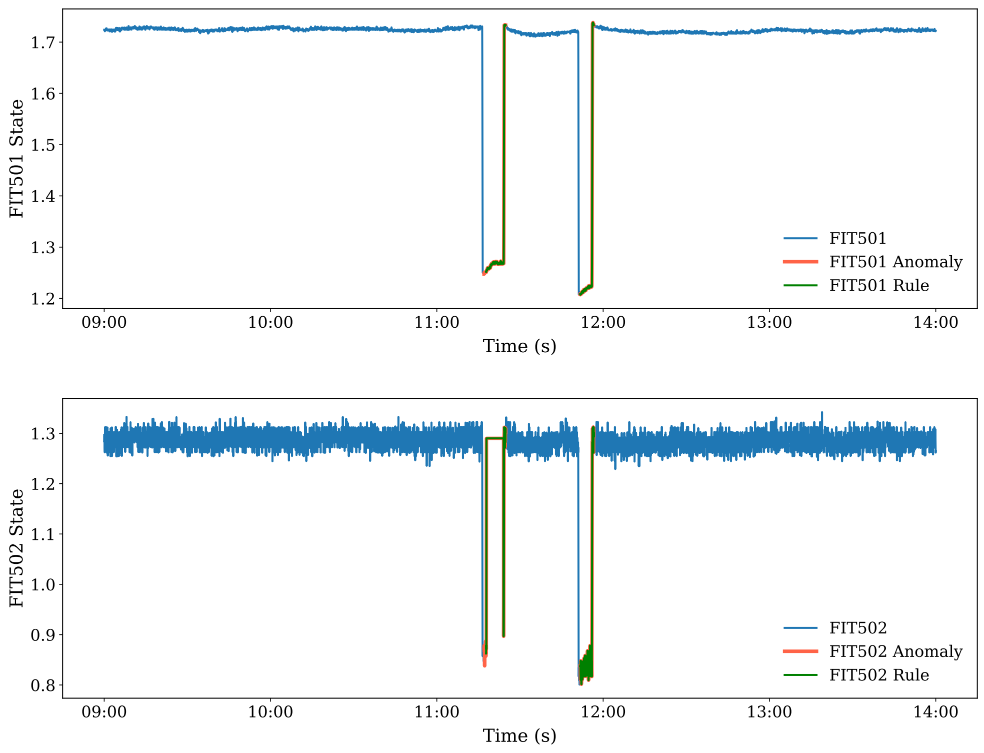

Figure 10. Detection performance of the invariant rules for single stage multi-point attacks on FIT501 and FIT502. The figure illustrates the effectiveness of the rule in identifying these types of attacks.

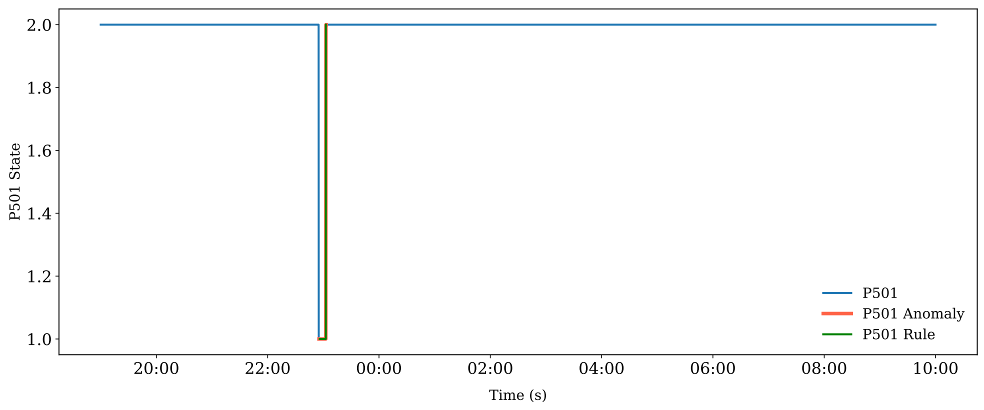

Figure 11. Detection performance of the invariant rules for multi-stage multi-point attacks on P501 and FIT502. The figure highlights the rule's effectiveness in identifying complex, multi-stage, and multi-point attack scenarios.

5. Conclusions

This study proposes a numerical association rule mining method based on swarm intelligence algorithms for anomaly detection in ICS. The method models the association rule mining task as a single-objective continuous optimization problem. By applying feature selection and task partition strategies, the method simplifies the management of complex tasks, improving search efficiency while ensuring the quality of the generated rules. The representation of solutions in the search space is optimized, which not only enhances computational efficiency but also ensures the high quality of the generated rules, ultimately enabling accurate identification of anomalous behavior in ICSs.

One significant advantage of this method is its ability to mine association rules across multiple subsystems, revealing the interactions between different components within the system. This cross-subsystem perspective allows for the identification of more complex anomaly patterns, which may only emerge through interactions between multiple subsystems and could be missed if individual subsystems were analyzed in isolation. Additionally, the generality of the method makes it adaptable to various types of ICS, requiring only the collection of log data during normal system operation for rule mining and anomaly detection. This characteristic gives the method strong applicability, allowing it to be deployed across diverse industrial control environments.

Experimental results demonstrate that the swarm intelligence algorithms, based on PSO, DE, and BA, can detect over 84% of attack instances and achieve a detection accuracy of more than 97%. The method not only exhibits high detection performance but also provides excellent interpretability, offering detailed analyses of the causes of anomalies, which helps operators quickly identify the source of issues.

However, there are still some limitations to the method. First, the quality of the data significantly impacts the performance of the method, as the size, completeness, and accuracy of the dataset directly affect detection results. Second, the method may struggle to detect certain complex attacks that have minimal impact on the physical state of the system, as these attacks may not cause significant anomalous behavior. Therefore, future research will explore the integration of deep learning and other technologies to further enhance the detection accuracy and robustness of the method, especially in dynamic and complex industrial environments.

Overall, the swarm intelligence-based anomaly detection method proposed in this study, by mining cross-subsystem association rules and exhibiting strong generality, offers an effective and reliable solution for ICS security monitoring. Future work could further incorporate additional technologies and optimization strategies to achieve more efficient and precise anomaly detection.

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

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original data presented in the study are openly available at https://github.com/shouwangshenyuan/NARM.git, accessed on 20 October 2024; see Section 3.1 for details.

Acknowledgments: We would like to express our gratitude to the anonymous referees for their valuable suggestions, which significantly contributed to the improvement of this work.

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

Abbreviations

ICSs	Industrial Control Systems
IT	Information Technology
OT	Operational Technology
PLCs	Programmable Logic Controllers
RTUs	Remote Terminal Units
SCADA	Supervisory Control and Data Acquisition
HMI	Human–Machine Interface
LOF	Local Outlier Factor
IF	Isolation Forest
SVMs	Support Vector Machines
CNNs	Convolutional Neural Networks
LSTMs	Long Short-Term Memory Networks
GNNs	Graph Neural Networks
AEs	Autoencoders
GANs	Generative Adversarial Networks
NARM	Numeric Association Rule Mining
DE	Differential Evolution
PSO	Particle Swarm Optimization
BA	Bat Algorithms
SWaT	Secure Water Treatment
TP	True Positives
FP	False Positives

References

1. Mekala, S.H.; Baig, Z.; Anwar, A.; Zeadally, S. Cybersecurity for Industrial IoT (IIoT): Threats, countermeasures, challenges and future directions. *Comput. Commun.* **2023**, *208*, 294–320. [CrossRef]
2. Falliere, N.; Murchu, L.O.; Chien, E. W32. stuxnet dossier. *White Pap. Symantec Corp. Secur. Response* **2011**, *5*, 29.
3. Canonico, R.; Sperlì, G. Industrial cyber-physical systems protection: A methodological review. *Comput. Secur.* **2023**, *135*, 103531. [CrossRef]
4. Liu, K.; Wang, M.; Ma, R.; Zhang, Z.; Wei, Q. Detection and localization of cyber attacks on water treatment systems: An entropy-based approach. *Front. Inf. Technol. Electron. Eng.* **2022**, *23*, 587–603. [CrossRef]
5. Ahmed, C.M.; Murguia, C.; Ruths, J. Model-based attack detection scheme for smart water distribution networks. In Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, Abu Dhabi, United Arab Emirates, 2–6 April 2017; pp. 101–113.
6. Li, S.; Hu, Y.; Zheng, L.; Li, Z.; Chen, X.; Fernando, T.; Iu, H.H.; Wang, Q.; Liu, X. Stochastic event-triggered cubature Kalman filter for power system dynamic state estimation. *IEEE Trans. Circuits Syst. II Express Briefs* **2018**, *66*, 1552–1556. [CrossRef]

7. Stefanidis, K.; Voyiatzis, A.G. An HMM-based anomaly detection approach for SCADA systems. In Proceedings of the Information Security Theory and Practice: 10th IFIP WG 11.2 International Conference, WISTP 2016, Heraklion, Crete, Greece, 26–27 September 2016, Proceedings 10; Springer: Berlin/Heidelberg, Germany, 2016; pp. 85–99.
8. Li, Z.; Zhao, Y.; Botta, N.; Ionescu, C.; Hu, X. COPOD: Copula-based outlier detection. In Proceedings of the 2020 IEEE International Conference on Data Mining (ICDM), Sorrento, Italy, 17–20 November 2020; pp. 1118–1123.
9. Breunig, M.M.; Kriegel, H.P.; Ng, R.T.; Sander, J. LOF: Identifying density-based local outliers. In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, Dallas, TX, USA, 16–18 May 2000; pp. 93–104.
10. Liu, F.T.; Ting, K.M.; Zhou, Z.H. Isolation-based anomaly detection. *ACM Trans. Knowl. Discov. Data (TKDD)* **2012**, *6*, 1–39. [CrossRef]
11. Elnour, M.; Meskin, N.; Khan, K.; Jain, R. A dual-isolation-forests-based attack detection framework for industrial control systems. *IEEE Access* **2020**, *8*, 36639–36651. [CrossRef]
12. Erfani, S.M.; Rajasegarar, S.; Karunasekera, S.; Leckie, C. High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning. *Pattern Recognit.* **2016**, *58*, 121–134. [CrossRef]
13. Zong, B.; Song, Q.; Min, M.R.; Cheng, W.; Lumezanu, C.; Cho, D.; Chen, H. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April –3 May 2018.
14. Xie, X.; Wang, B.; Wan, T.; Tang, W. Multivariate abnormal detection for industrial control systems using 1D CNN and GRU. *IEEE Access* **2020**, *8*, 88348–88359. [CrossRef]
15. Luo, D.; Wang, X. Moderntcn: A modern pure convolution structure for general time series analysis. In Proceedings of the Twelfth International Conference on Learning Representations, Vienna, Austria, 7–11 May 2024.
16. Fährmann, D.; Damer, N.; Kirchbuchner, F.; Kuijper, A. Lightweight long short-term memory variational auto-encoder for multivariate time series anomaly detection in industrial control systems. *Sensors* **2022**, *22*, 2886. [CrossRef]
17. Deng, A.; Hooi, B. Graph neural network-based anomaly detection in multivariate time series. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual, 2–9 February 2021; Volume 35, pp. 4027–4035.
18. Zhou, B.; Dong, Y.; Yang, G.; Hou, F.; Hu, Z.; Xu, S.; Ma, S. A graph-attention based spatial-temporal learning framework for tourism demand forecasting. *Knowl. Based Syst.* **2023**, *263*, 110275. [CrossRef]
19. Xu, J. Anomaly transformer: Time series anomaly detection with association discrepancy. *arXiv* **2021**, arXiv:2110.02642.
20. Xiong, W.; Wang, P.; Sun, X.; Wang, J. SiET: Spatial information enhanced transformer for multivariate time series anomaly detection. *Knowl. Based Syst.* **2024**, *296*, 111928. [CrossRef]
21. Tuli, S.; Casale, G.; Jennings, N.R. Tranad: Deep transformer networks for anomaly detection in multivariate time series data. *arXiv* **2022**, arXiv:2201.07284. [CrossRef]
22. Madan, N.; Ristea, N.C.; Ionescu, R.T.; Nasrollahi, K.; Khan, F.S.; Moeslund, T.B.; Shah, M. Self-supervised masked convolutional transformer block for anomaly detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *46*, 525–542. [CrossRef] [PubMed]
23. Qin, S.; Chen, L.; Luo, Y.; Tao, G. Multi-view graph contrastive learning for multivariate time series anomaly detection in IoT. *IEEE Internet Things J.* **2023**, *10*, 22401–22414. [CrossRef]
24. Li, D.; Chen, D.; Jin, B.; Shi, L.; Goh, J.; Ng, S.K. MAD-GAN: Multivariate anomaly detection for time series data with generative adversarial networks. In Proceedings of the International Conference on Artificial Neural Networks, Munich, Germany, 17–19 September 2019; Springer: Berlin/Heidelberg, Germany, 2019; pp. 703–716.
25. Liu, Y.; Ning, P.; Reiter, M.K. False data injection attacks against state estimation in electric power grids. *ACM Trans. Inf. Syst. Secur. (TISSEC)* **2011**, *14*, 1–33. [CrossRef]
26. Deng, R.; Zhuang, P.; Liang, H. False data injection attacks against state estimation in power distribution systems. *IEEE Trans. Smart Grid* **2018**, *10*, 2871–2881. [CrossRef]
27. Agrawal, R.; Imieliński, T.; Swami, A. Mining association rules between sets of items in large databases. In Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, DC, USA, 26–28 May 1993; pp. 207–216.
28. Agrawal, R.; Srikant, R. Fast Algorithms for Mining Association Rules. In Proceedings of the 20th International Conference on Very Large Data Bases, Santiago de Chile, Chile, 12–15 September 1994.
29. Zaki, M.J. Scalable algorithms for association mining. *IEEE Trans. Knowl. Data Eng.* **2000**, *12*, 372–390. [CrossRef]
30. Han, J.; Pei, J.; Yin, Y. Mining frequent patterns without candidate generation. *ACM Sigmod Rec.* **2000**, *29*, 1–12. [CrossRef]
31. Yoong, C.H.; Palleti, V.R.; Maiti, R.R.; Silva, A.; Poskitt, C.M. Deriving invariant checkers for critical infrastructure using axiomatic design principles. *Cybersecurity* **2021**, *4*, 1–24. [CrossRef]
32. Feng, C.; Palleti, V.R.; Mathur, A.; Chana, D. A Systematic Framework to Generate Invariants for Anomaly Detection in Industrial Control Systems. In Proceedings of the NDSS, San Diego, CA, USA, 24–27 February 2019; pp. 1–15.
33. Maiti, R.R.; Yoong, C.H.; Palleti, V.R.; Silva, A.; Poskitt, C.M. Mitigating adversarial attacks on data-driven invariant checkers for cyber-physical systems. *IEEE Trans. Dependable Secur. Comput.* **2022**, *20*, 3378–3391. [CrossRef]
34. Fister, I., Jr.; Fister, I. A brief overview of swarm intelligence-based algorithms for numerical association rule mining. In *Applied Optimization and Swarm Intelligence*; Springer: Singapore, 2021; pp. 47–59.
35. Blum, C.; Li, X. Swarm intelligence in optimization. In *Swarm Intelligence: Introduction and Applications*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 43–85.

36. Storn, R.; Price, K. Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [CrossRef]
37. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95-International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948.
38. Yang, X.S. A new metaheuristic bat-inspired algorithm. In *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 65–74.
39. Mathur, A.P.; Tippenhauer, N.O. SWaT: A water treatment testbed for research and training on ICS security. In Proceedings of the 2016 International Workshop on Cyber-Physical Systems for Smart Water Networks (CySWater), Vienna, Austria, 11 April 2016; pp. 31–36.
40. Vrbančič, G.; Brezočnik, L.; Mlakar, U.; Fister, D.; Fister, I. NiaPy: Python microframework for building nature-inspired algorithms. *J. Open Source Softw.* **2018**, *3*, 613. [CrossRef]
41. Tahyudin, I.; Nambo, H. The combination of evolutionary algorithm method for numerical association rule mining optimization. In Proceedings of the Tenth International Conference on Management Science and Engineering Management, Langkawi, Malaysia, 27 December 2017; Springer: Berlin/Heidelberg, Germany, 2017; pp. 13–23.
42. Kuo, R.J.; Gosumolo, M.; Zulvia, F.E. Multi-objective particle swarm optimization algorithm using adaptive archive grid for numerical association rule mining. *Neural Comput. Appl.* **2019**, *31*, 3559–3572. [CrossRef]
43. Tahyudin, I.; Nambo, H. Improved optimization of numerical association rule mining using hybrid particle swarm optimization and cauchy distribution. *Int. J. Electr. Comput. Eng.* **2019**, *9*, 1359. [CrossRef]
44. Fister, I.; Iglesias, A.; Galvez, A.; Del Ser, J.; Osaba, E.; Fister, I. Differential evolution for association rule mining using categorical and numerical attributes. In Proceedings of the Intelligent Data Engineering and Automated Learning—IDEAL 2018: 19th International Conference, Madrid, Spain, 21–23 November 2018; Proceedings, Part I 19; Springer: Berlin/Heidelberg, Germany, 2018; pp. 79–88.
45. Fister Jr, I.; Podgorelec, V.; Fister, I. Improved nature-inspired algorithms for numeric association rule mining. In Proceedings of the Intelligent Computing and Optimization: Proceedings of the 3rd International Conference on Intelligent Computing and Optimization 2020 (ICO 2020), Koh Samui, Thailand, 17–18 December 2021; Springer: Berlin/Heidelberg, Germany, 2021; pp. 187–195.