Article

Research on a PSO-H-SVM-Based Intrusion Detection Method for Industrial Robotic Arms

Yulin Zhou, Lun Xie * and Hang Pan

School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing 100083, China; g20198821@xs.ustb.edu.cn (Y.Z.); b20180306@xs.ustb.edu.cn (H.P.)
* Correspondence: xielun@ustb.edu.cn; Tel.: +86-13681560734

Featured Application: This research is mainly used for intrusion detection against physical process logic attacks that industrial robotic arms may be subject to, and the establishment of response mechanisms.

Abstract: The automation and intelligence of industrial manufacturing is the core of the fourth industrial revolution, and robotic arms and proprietary networked information systems are an integral part of this vision. However, with the benefits come risks that have been overlooked, and robotic arms have become a heavily attacked area. In order to improve the security of the robotic arm system, this paper proposes an intrusion detection method based on a state classification model. The closure operation process of the robotic arm is divided into five consecutive states, while a support vector machine based on the particle swarm optimization algorithm (PSO-H-SVM) classifies the operation state of the robotic arm. In the detection process, the classifier predicts the operation state of the robotic arm in real time, and the detection method determines whether the state transfer meets the logical requirements, and then determines whether the intrusion occurs. In addition, a response mechanism is proposed on the basis of the intrusion detection system to make protection measures for the robotic arm system. Finally, a physical experiment platform was built to test the intrusion detection method. The results showed that the classification accuracy of the PSO-H-SVM algorithm reached 96.02%, and the detection accuracy of the intrusion detection method reached 90%, which verified the effectiveness and reliability of the intrusion detection method.

Keywords: security; robotic arm system; PSO-H-SVM; intrusion detection; physical process logic attack

check for updates

Citation: Zhou, Y.; Xie, L.; Pan, H. Research on a PSO-H-SVM-Based Intrusion Detection Method for Industrial Robotic Arms. *Appl. Sci.* **2022**, *12*, 2765. https://doi.org/10.3390/app12062765

Academic Editor: Pericle Perazzo

Received: 8 February 2022
Accepted: 2 March 2022
Published: 8 March 2022

1. Introduction

With the rapid development of various fields such as network technology, communication technology, hardware and manufacturing, mankind has ushered in the fourth industrial revolution, and we are already enjoying its fruits. In particular, automation is the key development direction of industrial manufacturing, involving a number of areas such as robotics, the Internet of Things and artificial intelligence [1]. Among them, the robot is an indispensable piece of equipment in the entire link; through their study, it can be found that the manufacturing industry has become more and more intelligent in recent years, and the industrial robotic arm has played an irreplaceable role in this process. Industrial robotic arms play a major role in various fields such as manufacturing, assembly and aerospace, and have become an integral part of the industrial field. In the process of operation, they can improve efficiency, reduce costs and minimize personnel safety issues [2–4].

Traditional robotic arm systems are only used in closed operating environments and mainly rely on operators to operate, and only act as independent workstations. However, with the development of communication networks and related hardware, robotic arm systems have started to become networked [5], forming smart factories. The entire production system that relies on robotic arms is becoming more and more open, so cyber-physical security for robotic arm systems is of vital importance. The security research of robotic arm systems can effectively respond to possible threats and make the next step when a security breach is created [6].

As attacks against robotic arm systems are intensively studied, the current attacks may involve protocol attacks, sensor data attacks and control system attacks. For example, in [7], Li. et al. performed a data logic attack on the EtherCAT communication protocol of a heavy-duty industrial robotic arm, mainly targeting the data packets between the control system and the actuator or between the sensor and the control system, by reordering, dropping or delaying the protocol packets to break the data logic of the communication protocol. This attack can seriously damage the operation state of the robotic arm. In [8], Kim et al. proposed a sensor attack on mobile robots, using Global Positioning System (GPS) spoofing to interfere with GPS sensor readings and gyroscope sensing data spoofing through acoustic noise, ultimately leading to the crash of the unmanned aerial vehicle. In [9], Jeong et al. analyzed the vulnerabilities of the robot operating system (ROS), and they included an insufficient ROS authentication scheme, a ROS bag replay attack, service hijacking, etc., for which if the ROS is attacked using malware, then the control data can be tampered with, causing damage to the control system. Meanwhile, the damage caused by targeted attacks covers both physical and network domains [10], and attacks on the network or protocols will cause damage to the network domain; attacks on the control system, such as tampering with instructions or sensing data, control data spoofing attacks, etc., will not affect the network domain, but will most likely cause physical damage or even personnel damage with unimaginable consequences. Nowadays, there is a wide range of research on intrusion detection for information-physical systems, covering power systems, industrial production systems and remote-control systems [11–14] these intrusion detection methods serve as important references in building the security system of robotic arms. The security of robotic arm systems is also being studied through different aspects, with some researchers having established a dynamic model of the robotic arm for fault diagnosis and isolation [15,16], as well as adding some defensive measures against threats from external attacks through the design process of the robotic arm [17,18]. In [19,20], rule-based and estimation-based methods were used to detect the anomalies of the robotic arm.

A summary of the literature reveals that not enough attention has been paid to the security of the physical process of the robotic arm. If an attacker intercepts the data of the robotic arm operation process and completes a logical state analysis, the control data of the wrong logical state is injected in the intrusion process for the purpose of damaging the physical process logic. However, detection methods based on data range anomalies cannot detect such attacks and can cause significant damage to the physical domain. Machine learning methods are also used in the intrusion detection of robotic arms [21,22]. In summary, based on the factors that have less security analysis regarding the physical process logic of the robotic arm, and the increasingly widespread use of machine learning methods in intrusion detection algorithms, this paper constructs an intrusion detection method for physical process logic attacks on robotic arms through a machine learning-based approach, designed to ensure the stable and safe operation of robotic arm systems. The main contributions of this paper are summarized as follows:

- The intrusion detection model for hierarchical support vector machines is established based on the particle swarm optimization algorithm (PSO-H-SVM) and response mechanism.
- The composition of the robotic arm system based on the EtherCAT protocol and the logical attack of the physical process are analyzed.
- A physical experiment platform is built to test the performance of the established intrusion detection model and verify its effectiveness.

The rest of the paper is organized as follows: Section 2 describes the related work. In Section 3, an intrusion detection method based on machine learning is given to classify the real-time operational state of the robotic arm by PSO-H-SVM, to determine whether it conforms to the logical state of the physical process and thus performs intrusion detection.

Section 4 describes the components of the robotic arm system and data acquisition and processing, and presents possible logical attacks on the physical process of the robotic arm. Finally, the intrusion detection method is evaluated based on a physical experimental platform to verify the effectiveness of the method. In Section 5, the methods proposed in this paper are discussed and summarized.

2. Related Work

With the widespread use of robotic arms, their security issues are also receiving more and more attention. Research on intrusion detection technology for robotic arms is the most critical, effective intrusion detection method to better protect the system, but attack and defense have always been an inseparable whole. Therefore, although. many scholars study how to attack, their purpose is not to promote destruction, but to prepare better protection methods. Pang et al. [23] designed a method for a two-channel false data injection attack on a controller and sensor, where the controller and sensor are injected with false data and the detector finds it difficult to detect the anomaly, making the attack more stealthy; a Kalman-filter-based compensation is proposed for the delay generated by the attack. Clark et al. [24] proposed a malicious attack on the machine learning strategy of a robot system, where the target is indirectly attacked by being forced to deviate from what it has learned through Q-learning algorithms. However, if the sensor data are detected, the attack may be exposed and thus lacks stealth. Li et al. [25] designed a two-loop covert attack on an industrial control system, using least squares support vector machines to construct a covert attack with an attack loop and a covert loop, and verified the effectiveness of the covert attack with a PLC system. Khojasteh et al. [26] investigated a learning-based attack method to estimate the dynamics of a system through a nonlinear Gaussian process-based learning algorithm that attacked the control policy. Zhao et al. [27] used subspace recognition technology to propose an attack method based on wrong data injection. The target of the attack was the state estimation error, and the coding matrix was used to detect the attack. However, it does not apply to distributed data systems.

From some of the attack methods described above, it is not difficult to find that attacks can cause a lot of damage once they are successful, so many scholars have studied intrusion detection methods based on different approaches. Hector et al. [28] provided a design solution for a security trigger on a robotic arm that uses the torque tolerance error of the robotic arm to detect anomalies, and considers a security threat as having arisen when the error between the expected value and the true value is greater than a threshold value relative to the expected value; it also develops resilient security measures that allow the robotic arm to operate safely in the home position when anomalies are detected. This one security trigger is universal for the intrusion detection of robotic arms. Tang et al. [29] studied the tracking control security of robotic arms, considered malicious DOS attacks, made a description for their attack frequency based on their physical and duration characteristics, and built a hybrid model to detect the attacks. Hong et al. [30] proposed an integrated anomaly detection method for hosts and networks based on a robotic arm system to detect attacks at both the application and network layers. Khaitan et al. [31] proposed a security detection method based on different indicators of integrated circuits for different levels of security risks for robots; this intrusion detection method can have a good defensive effect on the attacks proposed in [32], guaranteeing the safe operation of robots. Zhou et al. [33] designed a layer-by-layer defense framework centered on a secure network architecture, secure industrial network protocols, secure control systems and secure physical processes, to achieve an acceptable level of security risk and ensure the stable operation of the system. The advantage of this framework is its faceted defense and better security results. In [34], a multi-model-based anomaly intrusion detection method was also proposed to detect anomalies in the system from the multiple perspectives of communication models, task models, resource models and control data flow models, and to classify alarms after anomalies are detected. Hidden Markov classifiers are also constructed to distinguish between anomalies and faults, due to the different handling of system anomalies and faults.

Similarly, with the widespread use of machine learning and deep learning, researchers have introduced intelligent algorithms into intrusion detection methods with good results, making learning-based intrusion detection methods mainstream. Akpinar et al. [35] proposed a learning-based intrusion detection method for the EtherCAT protocol, where an attack vector is created based on real-time data and a support vector machine training model is used to perform anomaly detection, which is used to detect anomalies in the communication protocol. In [36], the authors further classified the protocols into four categories by classifying the various features of the protocols. They produced 16 events in the 4 categories for anomaly detection, and the genetic algorithm-based support vector (GA-SVM) machine demonstrated good detection results. Qiu et al. [37] investigated an anomaly detection method based on Bayesian networks that can detect an attack on a robotic arm and distinguish whether it is an attack from the physical or network domain. Maushart et al. [38] used deep belief networks to detect certain known attacks in robotic arm systems, especially denial of service attacks and spoofing attacks. Narayanan et al. [39] applied the one-class support vector machine (One-Class SVM) to the joint angle anomaly detection of the robotic arm, and introduced the concept of the tolerance envelope to train supervised learning models, both with good results.

In summary, although there are many intrusion detection methods that can be used for robotic arm security, most of them analyze the communication model and data flow but lack a security analysis of the physical model itself. The physical process logic (PPL) attack occurs during robotic arm operation, that is, by overwriting the normal control data with the wrong state control data, which may cause the robotic arm operation system to be disrupted. This attack occurs if the joint state data at each moment of detection do not detect an abnormality, and although it is often the case that there is no abnormality observed in the data, the state disorder can cause serious damage to the physical domain. So, for this situation, this study aims to detect whether there is any abnormality in the logic of the physical process of the robotic arm, so as to ensure the safety of the robotic arm operation process.

3. Intrusion Detection Method and Response Mechanism

This section describes the relevant algorithms and methods used for intrusion detection. For the physical process logic attack mentioned above, a robotic arm physical process anomaly detection method is designed. Before detection, the classifier of the operating state of the robotic arm is first constructed, and the classification model of the hierarchical support vector machine based on the particle swarm optimization algorithm (PSO-H-SVM) is trained with the data set. As a result, the classifier with the expected effect is obtained.

In the process of detection, the real-time joint data of the robotic arm is processed into feature vectors for the input of the classifier, and the operation state of the robotic arm is predicted in real-time using the PSO-H-SVM algorithm, which in turn detects whether the state transfer of the robotic arm satisfies the normal physical process logic in the robotic arm physical process anomaly detection method. If an intrusion is detected, the response mechanism is triggered to protect the robotic arm system; otherwise, the next set of data is detected. The working block diagram of the intrusion detection mechanism is shown in Figure 1.

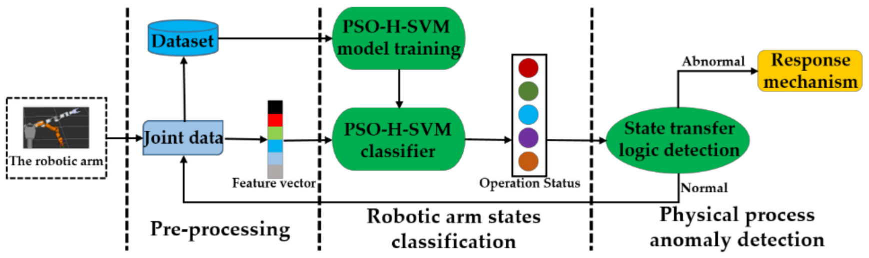


Figure 1. The whole process of the intrusion detection mechanism.

3.1. Particle Swarm Optimization Algorithm

Particle swarm optimization (PSO) is an iteration-based algorithm in which particles have only two attributes, velocit and position. Each particle gradually moves towards the position of the optimal solution during each iteration until all particles have completed the iteration and the optimal solution is found [40]. Assuming that the space of solutions is an n-dimensional space, then each particle is considered as a candidate solution to this n-dimensional search space, and each particle has memory. During each iteration, the particle updates the optimal position based on the calculated fitness value, and also updates the global optimal position based on the optimal positions of all particles in the particle swarm [41]. Assuming that there are m particles forming a particle swarm in an n-dimensional target search space, the iterative formulation of the algorithm is as follows:

$$V_i^{g+1} = \omega \cdot V_i^g + C_1 \cdot \mathrm{rand}_1 \cdot \left(\mathrm{pbest}_i^g - X_i^g\right) + C_2 \cdot \mathrm{rand}_2 \cdot \left(\mathrm{gbest}^g - X_i^g\right) \tag{1}$$

$$X_i^{g+1} = X_i^g + V_i^{g+1} \tag{2}$$

where, $V_i^g = (v_{i1}, v_{i2}, \cdots, v_{in}), i = 1, 2, \cdots, m$ is the velocity of the ith particle in the gth generation of the particle swarm; $X_i^g = (x_{i1}, x_{i2}, \cdots, x_{in}), i = 1, 2, \cdots, m$ is the position of the ith particle in the gth generation of the particle swarm; ω is a non-negative number and is the inertia weight; C_1 and C_2 are learning factors, also known as acceleration constants, representing the importance of the particle optimum and the global optimum; rand_1 and rand_2 are random numbers between 0 and 1; pbest_i^g is the historical optimum solution of the ith particle in the gth generation of the particle swarm; and gbest^g is the global optimum solution in the gth generation of the particle swarm.

The parameter ω in the algorithm mentioned above is the inertia weight, and the size of this parameter affects the local and global search capability of the algorithm. When the inertia weight is set relatively large, the algorithm has good global search capabilities and does not easily fall into the local optimum solution, but it is likely to miss the optimal point, resulting in oscillation around the optimal value; when the inertia weight is set relatively small, the algorithm has good local search capabilities, which can improve the accuracy of the solution, but it easily falls into local optimum. Shi, Y. et al. [42] proposed the linear decreasing inertia weight (LDIW) method, and applied it in experiments to achieve better results. The formula of the LDIW method is as follows:

$$\omega^g = \omega_{\max} - \frac{g - (\omega_{\max} - \omega_{\min})}{g_{\max}} \tag{3}$$

where, ω^g is the value of the inertia weight when the algorithm is updated to the gth generation; $\omega_{\max}$, $\omega_{\min}$ are the maximum and minimum values of the inertia weight, respectively; g is the current iteration step number; and $g_{\max}$ is the maximum iteration step number.

In the optimization process of the algorithm, the particle swarm is first initialized; then the fitness values of all particles are calculated according to the fitness function and compared, and the individual optimal solution and the global optimal solution are updated if they are better; secondly, the current inertia weight ω is updated according to the LDIW method, and the velocity and position of the particles are updated using the current inertia weight value. Finally, the iteration ends when the end condition is satisfied and the optimal solution is solved.

3.2. Hierarchical Support Vector Machine Based on PSO Algorithm

Support vector machines (SVM) started out by solving binary classification problems, and later, as the problems became more complex, SVM needed to be extended to multi-classification problems [43]. There are generally two classification methods for multi-classification problems: one is to combine multiple binary classifiers to complete a multi-classification task, the other is to modify the objective function directly by combining the parameter solutions of multiple classification surfaces into one optimization problem. A one-time solution is performed, but this latter method has too many parameters in the optimization process and the training speed is slow, so it is generally not used. Combinatorial binary classifier methods include one-versus-rest SVM (1-v-r SVM), one-versus-one SVM (1-v-1 SVM) and hierarchical SVM (h-SVM).

In this paper, the h-SVM approach is used, and assuming that there are k categories to be classified, then the h-SVM will construct k − 1 classifiers to complete the classification task. First, all k categories are divided into two categories, and then subcategories are divided step by step until each individual category is divided [44]. In this paper, the operating states of a robotic arm are classified into five categories using the joint angle, velocity, acceleration and the closing angle of the end gripper as the feature vectors. Those five categories are: operating to the grasping point, performing grasping action, operating to the placing point, performing placing action and resetting the arm, and the corresponding state labels are state 1, state 2, state 3, state 4 and state 5. Then, 4 classifiers need to be constructed to classify the 5 types of operation states. Since all categories are firstly divided into two categories and then divided in turn, the problem faced is that there are many ways to build h-SVM. For example, it is possible to divide state 1 and state 2 into a category separate from state 3, state 4 and state 5, and also state 2 and state 4 into a category separate from state 1, state 3 and state 5, and the same principle applies to the next level of classification. Different construction methods will have different classification effects, so it is crucial to construct a reasonable h-SVM model.

Think of the configuration of the classifier of the h-SVM as a tree. Starting at the top level of the tree, if the number of categories in the left and right subtrees of each tree node is not balanced, then it is called a “skewed tree”; if the number of categories in the left and right subtrees of a tree node is equal, then it is called a “normal tree”; the rest of the structure is somewhere in between. The more the tree shape is oriented like a “normal tree”, the better the classification effect will be. Secondly, starting from the top level of the tree, it is necessary to make the accuracy of the classifier at the upper level as high as possible, that is, the two subclasses are as distinguishable as possible [45].

Following the above two principles, the structure of the h-SVM shown in Figure 2 was constructed, and the tree type is more inclined towards being a “normal tree”. Classifier 1 classifies state 2 and state 4 and state 1, state 3 and state 5 into two classes, with state 2 and state 4 indicating that the grabbing and placing actions are more separable from the other three states. Similarly, classifier 2 classifies state 2 with state 4, classifier 3 classifies state 3 with state 1 and state 5 and classifier 4 classifies state 1 with state 5.

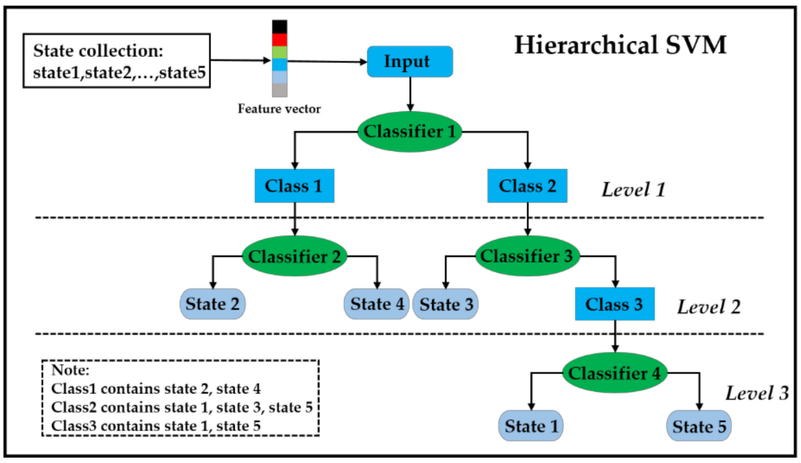


Figure 2. Schematic diagram of the structure of the h-SVM classifier.

The core of SVM lies in the kernel function, which maps the low-dimensional data into a high-dimensional space. The performance of a classification mainly depends on the choice of kernel function and the setting of hyperparameters. The universal Gaussian radial basis function (RBF) is chosen as the kernel function in this paper, and the hyperparameters have the penalty coefficient C and the parameter gamma, which comes with the RBF function. The size of the penalty factor C affects the generalization ability of the model, and the size of the gamma affects the number of support vectors, which in turn affects the speed of training and prediction. To further improve the classification accuracy of the h-SVM model, the particle swarm optimization algorithm mentioned in Section 3.1 is used for optimization, and the hyperparameters are optimized using PSO while training for each classifier. The chosen fitness function is the sum of the number of classification errors in each category, and the formula is as follows:

$$\mathrm{fv} = \sum_{\mathrm{i}=0,\mathrm{j}=0}^{\mathrm{i}=\mathrm{k}-1,\mathrm{j}=\mathrm{k}-1} \mathrm{confusion\_matrix}[\mathrm{i}][\mathrm{j}], \mathrm{i} \neq \mathrm{j} \tag{4}$$

where, fv is the fitness value; confusion_matrix[i][j] is the value of the ith row and jth column of the confusion matrix; and k is the number of categories.

The pseudo-code for the overall workflow of the hierarchical support vector machines based on PSO (PSO-H-SVM) model is shown in Algorithm 1. Each particle is a two-dimensional data point, respectively C and gamma, and the position of each particle represents a set of optimization parameters. The global optimal solution is used in the iterative process to assign values to the two hyperparameters for model training, and the confusion matrix is calculated after each training session, followed by the calculation of the fitness value. The individual optimal solution and the global optimal solution are updated if the optimization conditions are met. After updating the velocity and position of the particles, the next iteration is performed until the optimization is completed by finding a set of optimal solutions, after which the model is used to predict the test set and calculate the accuracy of the model.

Algorithm 1 Hierarchical Support Vector Machine Based on PSO Algorithm

```
procedure optimizing each classifier in H-SVM using PSO algorithm
  Input: X, Y // X is the eigenvector, Y is the result label
  for classifier_i in $(k - 1)$ do
    particle_init = [(random(), random())*m] // m is the number of particles
    g = 0
    while $g < g_{max}$ do
      for i in m do
        position = particle_init[i]
        SC = SVC(kernel = 'RBF', C = position[0], gamma = position[1]).fit(X, Y) // Creat SVM
        CM = confusion_matrix(Y_true, SC.predict(X)) // Calculate the confusion matrix
        fv = sum(CM[i][j]) $(i \neq j)$ // Calculate the fitness value
        if pbest_fv > fv and gbest_fv > fv then
          Update individual and global optimal solutions
        $w = w_{max} - itrea*(w_{max} - w_{min})/g_{max}$ // Update inertia weight
        Update the position and velocity of each particle
    C, gamma = gbest_position[0], gbest_position[1]
  Combine classifiers and calculate accuracy
  Output: The values of C, gamma and accuracy
```

3.3. Intrusion Detection and Response Mechanism

In Section 3.2, a classification method for the operating state of the robotic arm is proposed, and the physical process logic detection and response mechanism is proposed based on the PSO-H-SVM model classification. As shown in Figure 3, the intrusion detection module and the response mechanism module are embedded in the robotic arm control system for securing the robotic arm system.

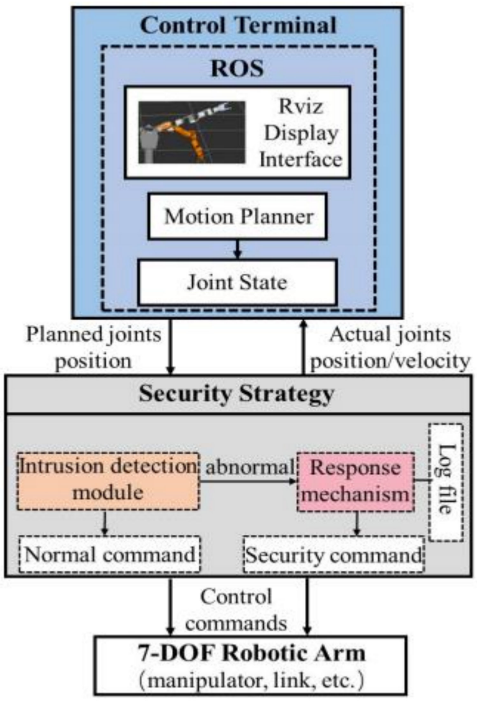


Figure 3. Robotic arm system based on detection and response.

In the robotic arm control system, the robot operating system (ROS) is run on the control terminal for motion planning of the robotic arm trajectory. The target pose of the robotic arm is input into the motion planning interface of the ROS when the robotic arm is performing the operation, and after the planner performs the trajectory planning [46], the inverse solver performs the inverse kinematic solution of the trajectory points to solve the joint position of each joint of the robotic arm. The joint position is sent as a control command to the actuator of the robotic arm to complete the operation task, and the actual joint position, speed and acceleration are fed back to the ROS while the robotic arm is in motion. The robotic arm in this paper is a 7-DOF robotic arm, with a redundant degree of freedom and more flexible movements that help the operation process avoid obstacles to complete the task. The robotic arm is mainly composed of mechanical links, control unit hardware, drivers, and sensors to complete a series of functions from controlling to driving to actuating and sensing data feedback. The composition of the robotic arm system is described in detail in Section 4.1 below. In this paper, an intrusion detection module and a response mechanism module are embedded in the robotic arm control system. The intrusion detection process will be carried out before the data are sent down to the robotic arm to prevent the attack from causing serious damage to the robotic arm system. The data are sent down to the robotic arm when no abnormality is detected, and the response mechanism is triggered to protect the robotic arm system when an abnormality is detected. The intrusion detection module detects the physical process logic of the robotic arm to determine whether the physical process of the robotic arm conforms to the normal logic during the operation, and if an abnormality is detected, the response mechanism module is triggered.

In the process of completing the entire closure operation of the robotic arm, the control instructions can be executed in the following order:

- Step 1: The initial position of the robotic arm
- Step 2: Move towards the target object position and reach the target object
- Step 3: Execute the grasping action
- Step 4: Move towards the placement position and reach the placement position
- Step 5: Execute the placing action
- Step 6: Reset the robotic arm.

The first and last of the above six execution sequences can be classified as the same state, and the robotic arm is reset by running it back to the initial position, so its motion state can be divided into five categories, namely: running to the gripping point, executing the gripping action, running to the placement point, executing the placement action, and resetting the robotic arm. cs is used to represent each state, S is used to represent the state collection, and SDS is used to represent the state scheduling collection, where S contains all the running states and the SDS state scheduling collection contains all the normal state scheduling. For example, $\mathrm{cs}[1] \to \mathrm{cs}[2]$ means transferring from state 1 to state 2, which conforms to the physical process logic. If $\mathrm{cs}[3] \to \mathrm{cs}[1]$ means jumping from state 3 to state 1, which does not conform to the physical process logic, this state scheduling does not exist in SDS. The expressions of cs and S are as follows:

$$\mathrm{cs}[\mathrm{i}] = \{\mathrm{cv}[\mathrm{i}][\mathrm{j}] | \mathrm{i} = 1, 2, \cdots, \mathrm{k},\ \mathrm{j} = 1, 2, \cdots, \mathrm{m}\},\ \mathrm{i} = 1, 2, \cdots, \mathrm{k} \tag{5}$$

$$\mathrm{S} = \{\mathrm{cs}[\mathrm{i}] | \mathrm{i} = 1, 2, \cdots, \mathrm{k}\} \tag{6}$$

$$\mathrm{SDS} = \{\mathrm{cs}[\mathrm{i}] \to \mathrm{cs}[\mathrm{j}] | \mathrm{j} - \mathrm{i} = \mathrm{e},\ \mathrm{e} = 0, 1\} \tag{7}$$

where, cv[i][j] is the ith eigenvalue in the ith state; cs[i] is the ith state, that is, cv[i][j] synthesizes state i; k is the number of all states; m is the number of eigenvalues indicating each state; and e is the difference of state values, taking only the two values of 0 and 1. The state eigenvalues are the joint-related parameters expressing each state, containing 23 sets of eigenvalue information on position, velocity, acceleration of the 7 joints, and the closing angle of the gripper joints.

The pseudo-code for the intrusion detection algorithm is shown in Algorithm 2. The physical process intrusion detection module is to detect whether the operation state of the robotic arm is abnormal during the entire closure operation. If the state transition does not exist in the SDS during the operation of the robotic arm, it means there is an abnormality. The intrusion detection module is the first to build a state classifier, in the normal operation of the robotic arm, that completes the closure operation process and collects the joint position, joint speed, joint angular velocity and the closing angle of the jaws of each path. The collected data will be pre-processed to produce data sets, with all the data divided into five categories of states according to the actual operating conditions and marked with the corresponding labels. The PSO-H-SVM model mentioned in Section 3.2 is trained based on the dataset, and the model is saved for later state classification when the model training achieves the expected effect of state classification. In the real-time intrusion detection of the running robotic arm, the real-time running joint data are collected, and the data are pre-processed and inputted into the classification model to classify the current running state, and to judge whether the state transfer is in line with the situation in SDS. If it is in line, then detection of the next set of real-time data should continue; if not, it means that an abnormality is detected, and the response mechanism should be triggered to carry out security protection measures for the robotic arm.

The response mechanism is an integral part of the intrusion detection system and is used to safeguard the system from malicious users [47]. The response mechanism module is triggered when a physical process logic anomaly is detected, and the response mechanism includes alarms, robotic arm protection measures, and logging functions. Among them, the alarm function is to display an alarm message at the detection terminal after detecting an abnormality: {Alert: An abnormality in the system, the transfer of state x-> state x is illegal, please check!}. This prompts the operator that an abnormality has occurred, and for them to manually intervene in the control of the robotic arm. The protection measure of the robotic arm is to continuously send the current control command of the joint position to keep the robotic arm in its current position and from no longer running. The measures used in engineering are generally to make the robotic arm power down and shut down; compared to the proposed method in this paper, the advantage is that there is no need to shut down the entire system, as the operator still has access to the robotic arm, which can be further secured. The logging function is to record abnormal information and state transition information when an abnormality is detected for subsequent analysis by the operator. The integrated application of the above intrusion detection module and response mechanism module can effectively detect physical process logic anomalies and mitigate damage caused by an intrusion, safeguarding the safety of personnel and robotic arms.

Algorithm 2 Physical Process Logic Intrusion Detection

```
procedure Detect exception and return alert
    Input: Joint_data // The eigenvector of joint position, velocity, etc.
    pre_state = state 1
    detection_value = True // Detection result
    SC = load_SVM() // Load the created classifier
    while detection_value == True do
        Joint_data = data_capture() // Obtaining joint data
        cur_state = SC.predict(Joint_data)
        if pre_sate->cur_state not in SDS then
            // Print error messages and trigger the response mechanism
            Debug.log(alert)
            Start (response_ mechanism)
            detection_value = False
         log(normal)
         pre_state = cur_state
    Output: detection_value is True or False
```

4. Experiments and Evaluation of Results

4.1. System Components

The industrial robotic arm operating system is an intelligent control system that exists for automated production lines. As shown in Figure 4a, the robotic arm operating system consists of a control terminal, a control software system, a 7-DOF robotic arm and the EtherCAT communication protocol. The physical model of the 7-DOF robotic arm is shown at the bottom of Figure 4a, which is composed of mechanical links, control unit hardware, drivers and sensors. The mechanical link is the actuating unit of the system.

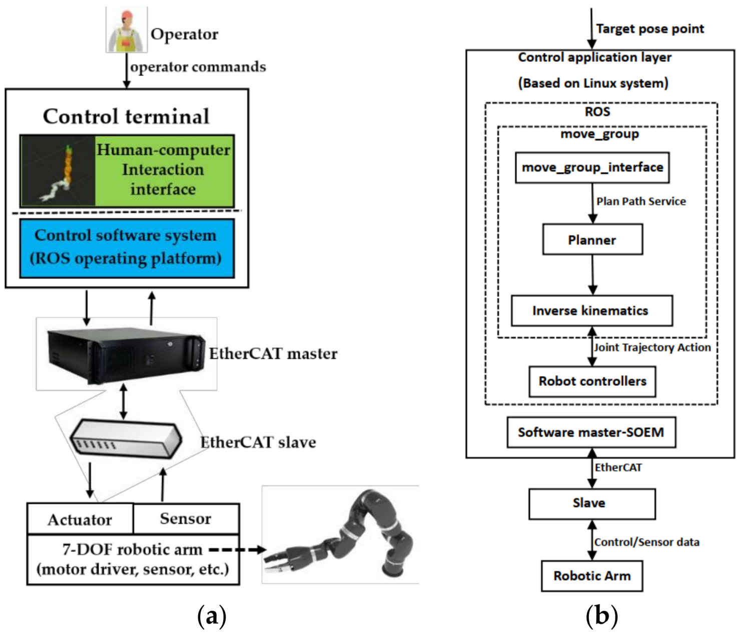


Figure 4. (**a**) The robotic arm operating system component. (**b**) Control software system.

The control unit hardware is an electronic governor that performs the function of controlling the motor; the driver provides power output to the robotic arm; and the sensor provides feedback data such as joint angle, speed and current, etc. The EtherCAT communication protocol is implemented as a master-slave structure, where the master is responsible for sending down the control data, receiving the sensor data from the slave, and loading all slave data in the downstream packet, while the slave unloads its own relevant data after receiving it, loads the feedback data and sends it to the next slave [48]. The control terminal mainly performs the human–computer interaction function, and is responsible for processing the operation instructions issued by the operator and receiving the data fed by the sensors.

The control software system is shown in Figure 4b. The control system is built based on the Linux operating system, mainly using the robot operating system (ROS) as the operating platform to complete the motion planning task of the robotic arm, which realizes the intelligent operation of the robotic arm. For the whole system, the operation point that the robotic arm needs to reach in the operation process is input into the ROS system, and the ROS plans the movement of the target point through the move_group function package. Firstly, all the paths that can reach the target point at the end are planned, then the planner will select the optimal path among these executable paths and extract its path points according to the control cycle. Finally, the inverse solver performs the inverse solution on the path points to find out the desired position of each joint, and connects with the communication interface through robot controllers to send the desired position data of the joints to the robotic arm through the EtherCAT communication protocol. In the whole control process, the sending of control data and the feedback of sensing data form a closed-loop control, and the control terminal waits for the arrival of the next command after the execution of the command sent by the operator is completed. According to the above elaborated composition of the robotic arm operating system, the physical experiment platform is completed based on this system architecture, as shown in Figure 5.

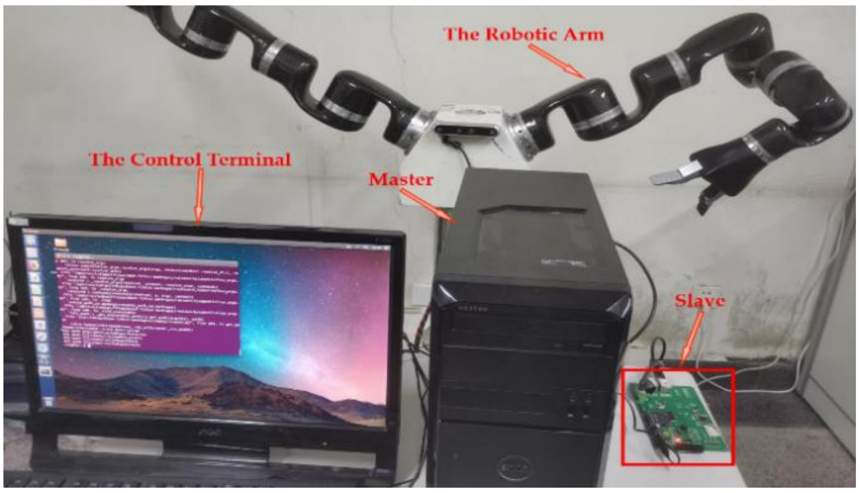


Figure 5. Physics experiment platform.

4.2. Data Collection and Processing

In the experiments, we collected the joint state data during the real-time closure operation of the robotic arm, which was used as the training data set for the machine learning model mentioned in Section 3.2. In the whole process of data collection, the robot arm completes the closure operation process in real time, that is, it starts to run from the initial position, grabs and places the target object in the designated position, and finally, the robot arm resets and returns to its initial position; the control terminal subscribes to the joint status data in real time to ensure the authenticity and availability of the collected data.

In the experiment, we chose a sampling frequency of 200 Hz, that is, a sampling period of 5ms, to acquire a set of joint state data on the robotic arm. The method of data collection was to subscribe to the/joint_state_publisher message topic by writing a topic subscription program through the ROS control terminal. This topic publishes the joint state data structure under sensor_msgs.msg, which contains information such as the joint position, joint speed, motor current value and time stamp of each joint of the robotic arm. In order to show the data of the subscription terminals more visually, we created a table as shown in Table 1, and due to space limitation, only two sets of data are shown. After subscribing to the message data of this topic, we chose to store the joint position, joint speed and current value of the motor in the data structure as JSON format files for later data processing. In the experiment, a total of 10 paths of the complete closure operation process were collected, and the 5 states were stored as 5 data files in the whole closure workflow. Then the tags of the corresponding state data files were marked out, and finally, all the joint state data and the corresponding tags were integrated to complete the production of the training data set. In order to verify the authenticity and usability of the data, the collected joint data were played back in the experiment, the robotic arm executed some of the collected joint data, and the experiment proved that it can complete the task of grasping and placing the target object.

Table 1. Example of data from a subscription terminal.

No.	Attribute	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6	Joint 7
1	Position (rad)	0.743	0.969	−0.396	0.510	−0.545	0.492	−1.152
	Velocity (rad/s)	−0.218	−0.291	0.288	−0.314	0.297	−0.316	0.330
	Current (A)	4.314	5.067	0.112	4.131	2.928	0.546	0.957
2	Position (rad)	0.396	1.122	−0.287	0.472	−0.318	0.576	−0.827
	Velocity (rad/s)	−0.237	−0.300	0.279	−0.316	0.274	−0.324	0.336
	Current (A)	4.565	4.850	0.140	4.183	2.776	0.502	0.907

All data retained 3 decimal places.

In order to have a better training effect, we processed the collected data. The units for joint position and joint speed as collected through the ROS terminal are radians (rad) and rad/s. The data are generally small, and in order to have a better model training effect, the unit was converted from the radian system to the angle system.

The joint acceleration value is used in the feature vector; however, at the control terminal, we only collected the joint velocity value, and the joint acceleration value needs to be calculated when performing data processing. However, the process of velocity data acquisition will be affected by noise, and if the data are not processed, that will affect the accuracy of the acceleration calculation. So, in this paper, the zero-phase low-pass Butterworth filter was used to filter the noise of the velocity data, and this filter uses a 10 Hz cut-off low-pass frequency. The acceleration value was calculated after noise filtering of the velocity data, and the formula is as follows:

$$\mathrm{Ac} = \frac{\mathrm{v}_{\mathrm{t}2} - \mathrm{v}_{\mathrm{t}1}}{\mathrm{t}2 - \mathrm{t}1}, \mathrm{t}2 - \mathrm{t}1 > 0 \tag{8}$$

where, Ac is the joint acceleration value; $\mathrm{v}_{\mathrm{t}2}$ and $\mathrm{v}_{\mathrm{t}1}$ are the velocity values at the moment of t2 and t1, respectively; and the sampling period is 5 ms in the experiment, so $\mathrm{t}2 - \mathrm{t}1 = 0.005$.

The above process completes the data acquisition and data processing work, which are fully prepared for the subsequent model training and intrusion detection experiments.

4.3. Physical Process Logic Attack

Nowadays, all attacks against robotic arms are mainly focused on vulnerabilities in the operating system and communication nodes. Since the operation of the robotic arm mainly relies on control data and feedback sensor data, attacks against control data and sensor data are, in some effect, equivalent to sensor failure or actuator failure in the physical domain, respectively [10]. After the control data are planned by the ROS, they are packaged as EtherCAT data packets for distribution. The data packaging junction is easily damaged by an attacker who can overwrite the normal control data with false data for data packaging and distribution. An attack on the control data can easily cause confusion in the robot operation and result in a physical domain attack, which can have serious consequences.

For the intelligent operation line, each robotic arm has a predetermined execution trajectory and status to ensure the completion of the closure operation process. Physical process logic (PPL) attacks occur during the operation of the robotic arm, that is, by overwriting normal control data with control data of the wrong state, which may lead to disorder in the robotic arm operation system and produce very serious damage in the physical domain [23]. A robotic arm was built in the laboratory for the grasping and placing of objects. By controlling the robotic arm, a round of closure grasping and placing operations can be completed; that is, the robotic arm runs from the initial position to the position where the item is placed, the gripper closes for grasping and then runs to the designated position for placing, and then the gripper opens to place the item and runs to the initial position.

The complete joint state data are collected during the whole operation process, and the joint angle change of the whole process is made into a two-dimensional image, as shown in Figure 6a. The horizontal coordinate "np" indicates the number of data points acquired, the vertical coordinate "joint angle" indicates the joint angle of each joint of the robot arm during operation, and the unit is the angle system. We can clearly observe each logical state of the robotic arm. Each operating state is marked in the figure, where s1 indicates running from the initial position to the position where the item is placed, s2 indicates that the gripper is closed for grasping, s3 indicates running to the specified placed position, s4 indicates that the gripper is open for placing the item, and s5 indicates running to the initial position.

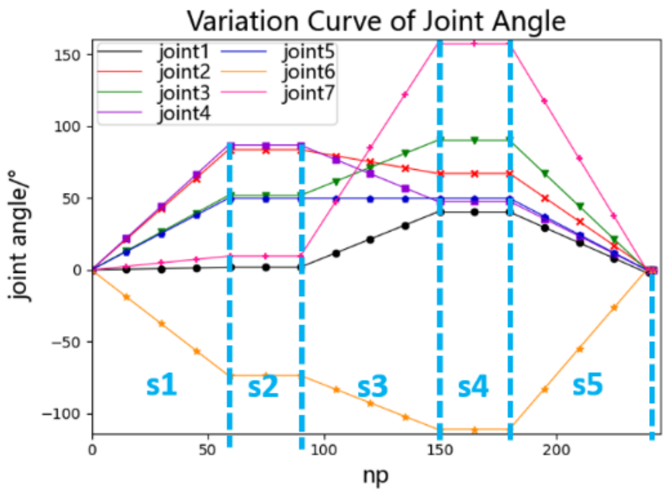
If the PPL attack is performed on the control data completed for planning, the attacker uses any one state of data to overwrite the normal control data before the control data are sent down during normal operation, which will lead to disorder in the logic state of the robotic arm operation. The attack model description formula is as follows:

$$\text{Normal state transfer}: \ \mathrm{s_{i+1}} = \mathrm{e_{i+1}},\ \mathrm{i} = 0, 1, \cdots, 4 \tag{9}$$

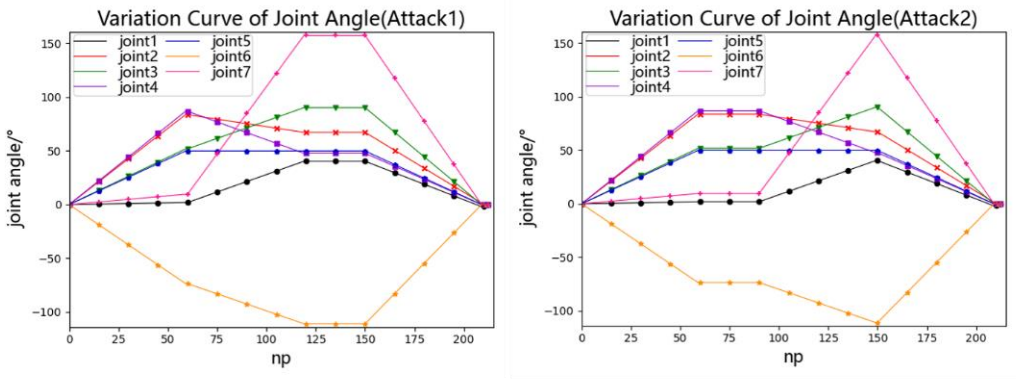
$$\text{PPL attack state transfer}: \ \bar{\mathrm{s}}_{\mathrm{i+1}} = \begin{cases} \mathrm{e_{i+random(-(i-1),\,0)}}, 1 \le \mathrm{i} \le 4 \\ \mathrm{e_{i+random(2,n-i)}}, 0 \le \mathrm{i} \le 3 \end{cases} \tag{10}$$

where, $\mathrm{s_{i+1}}$ is the next transfer state in the normal state, $\bar{\mathrm{s}}_{\mathrm{i+1}}$ is the next transfer state in the attack state, $\mathrm{e_i}$ is the current running state, $\mathrm{e_{i+x}}$ is x post-shift states of state i in the state set, $\mathrm{random}(-(\mathrm{i}-1),\ 0)$ and $\mathrm{random}(2, \mathrm{n}-\mathrm{i})$ are random integers in the range of $(-(\mathrm{i}-1),\ 0)$ and $(2, \mathrm{n}-\mathrm{i})$, and n is the total number of states, which has a value of five.

The attack generated as shown in Figure 6b is to run to the placement location without executing the grab action, or to run to the home location after reaching the specified placement location without executing the placement action, which can cause damage to the physical domain. Some attacks can even cause the state to jump, resulting in a sudden increase in joint speed and damage to the robotic arm. For the PPL attack described above, if there is no effective intrusion detection mechanism, then it may bring huge security problems and potential threats to the robotic arm system. Once a PPL attack occurs, it will not cause a change in the protocol length, nor will it cause a change in the transmitted packet traffic, and nor will it cause an alarm for abnormal control data. However, it will lead to a serious operational accident with great damage to the robotic arm.



(**a**)



(**b**)

Figure 6. (**a**) Joint angle data under normal condition. (**b**) Joint angle data under PPL attack.

4.4. Analysis of Results

In this subsection, we analyze the results of the hyperparameter optimization and the classification effect of the proposed machine learning model PSO-H-SVM, and simulate the intrusion behavior of the attacker on a physical experimental platform, testing the effectiveness of the physical process logic intrusion detection method based on state classification and the role of the response mechanism.

In this experiment, for the training dataset on robotic arm state classification, the experimentally completed dataset in Section 4.2 is used. To improve the accuracy of the classification model algorithm, the PSO-H-SVM classification model is proposed, multiple hierarchical classifiers are constructed, and the penalty parameter C and the kernel function parameter gamma are optimized by the PSO algorithm. The dataset is trained using the traditional SVM, PSO-SVM and PSO-H-SVM algorithms. Finally, we evaluate the merit of each machine learning model in terms of its classification accuracy [49], the accuracy being the ratio of the number of correctly predicted samples to the number of all samples; in order to calculate the accuracy, we need to know the true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN) for each category, calculated as follows:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \tag{11}$$

When training with the traditional SVM algorithm, default values are used for the hyperparameters such that $\text{C} = 1.0$ and gamma = 'auto' ; for training with the PSO-SVM algorithm, we use the SVC class from the scikit-learn library to perform multi-categorization directly, set the attribute value of decision_function_shape to 'ovo', build a classifier between each two categories, and finally, predict the categories by the "vote method". The initialization parameters for the PSO algorithm are set as follows: the number of individuals in the particle swarm is set to 120, the number of iterations is set to 20, the learning factors C_1 and C_2 are set to 2.0, the inertia weight is linearly decreasing, the search range of C is set to [0.001, 15], the search range of gamma is set to [0.01, 1]. For the PSO-H-SVM algorithm, four classifiers are constructed in order to classify five classes of operational states, and the four classifiers are optimized separately using the PSO algorithm. The classifiers are combined to construct a hierarchical support vector machine after parameter optimization is completed, and the initialization parameters in this algorithm are set as above for the PSO algorithm.

The values of the hyperparameters and the classification accuracy of the traditional SVM, PSO-SVM and PSO-H-SVM algorithms are shown in Table 2. The classification accuracy of the traditional SVM algorithm reached 93.87%, the classification accuracy of the PSO-SVM improved significantly to 95.04%, and finally, the accuracy of the PSO-H-SVM algorithm improved further compared to PSO-SVM, to 96.02%. To further analyze the performance of the PSO-H-SVM algorithm, we use the confusion matrix to evaluate the classification effect of the algorithm. The confusion matrix obtained from the result of classifying the operating state of the robotic arm using the PSO-H-SVM algorithm is shown in Figure 7. It can be observed from the confusion matrix that state 1 and state 5 have the highest classification accuracy at over 97%, and the rest of the states have an accuracy of over 95%. By analyzing the accuracy rate, we can verify that the proposed PSO-H-SVM model has a good classification effect.

Table 2. Hyperparameter values and accuracy of each algorithm.

Methods	Classifier	C	Gammma	Accuracy
SVM	classifier1	1.0	'auto'	93.87%
PSO-SVM	classifier1	7.5592	0.0256	95.04%
PSO-H-SVM	classifier1	8.8983	0.0796	96.02%
	classifier2	6.1910	0.0941	
	classifier3	9.2709	0.0707	
	classifier4	9.5290	0.0214	

The Classifier item in the table represents the number of classifiers constructed by each algorithm.

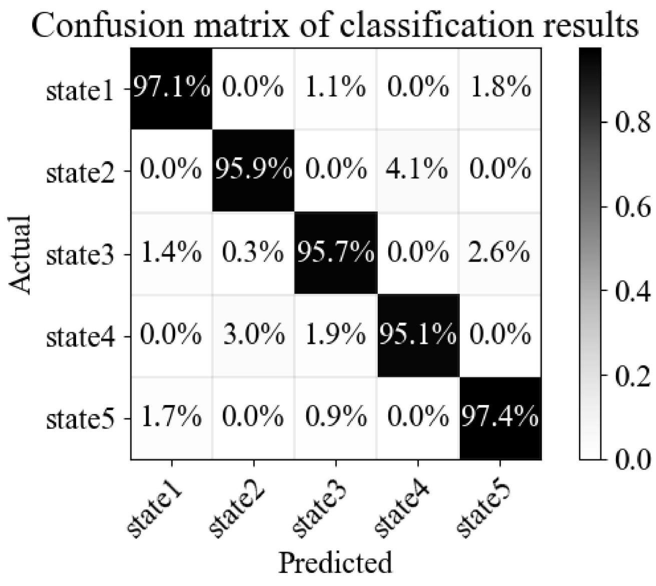


Figure 7. Confusion matrix of classification results.

Finally, we tested the physical process logic intrusion detection method on the physical experiment platform, and used two metrics to measure the effectiveness of the intrusion detection method throughout the testing process [50]. The first metric is the accuracy rate, which is the sum percentage of the number of correct judgments of the normal state and the attack state with the total number of tests; the second metric is the false detection rate, which refers to the percentage of the number of times the normal state is incorrectly judged as the attack state over the total number of times the normal state is judged. The expressions of the accuracy rate and the false detection rate are as follows:

$$\mathrm{ACC} = \frac{\mathrm{NN} + \mathrm{AA}}{\mathrm{Tnum}} \tag{12}$$

$$\mathrm{FAR} = \frac{\mathrm{NA}}{\mathrm{NN} + \mathrm{NA}} \tag{13}$$

where, ACC is the accuracy rate; FAR is the false detection rate; NN is the number of times the normal state is judged correctly; AA is the number of times the attack state is judged correctly; NA is the number of times the normal state is judged as the attack state; and Tnum is the total number of tests.

In the test experiment, the intrusion behavior of the attacker was simulated at the ROS control terminal, and joint commands with abnormal physical processes were issued to the robotic arm while performing the attack. For steps 1–6 of the closure operation presented in Section 3.3, the possible attacks on the system are given in Table 3. For the previously collected 10 paths, normal operations and attack experiments were conducted, resulting in a total of 100 sets of experiments, of which 90 are sets of the robotic arm performing normal instructions and 10 are sets of attack behavior. The experimental results were that 83 groups were judged as normal behavior, of which 81 groups were judged correctly and 2 groups were judged as normal behavior despite being attack behavior; 17 groups were judged as attack behavior, of which 8 groups were judged correctly and 9 groups were judged as attack behavior despite being normal behavior. According to Equations (12) and (13), the accuracy rate of this intrusion detection method can be calculated as 90% and the false detection rate as 10%. In the event of an attack, the terminal will prompt the operator to detect an abnormality and display the joint state value that maintains the current position, at which time the system is not powered down and the operator can intervene in the robotic arm system to take the next protective measures.

Table 3. Attack behavior of the system.

No.	Steps under Attack	Attack Commands
1	Step 2	Execute the placement command
2	Step 3	Execute the abnormal movement command
3	Step 4	Execute the grab command
	Step 4	Execute the reset command
4	Step 5	Execute the grab command
	Step 5	Execute the reset command

In summary, the accuracy of the PSO-H-SVM classification model has reached 96.02%, and the classification accuracy of each class of states based on the confusion matrix can be observed to be more than 95%. For physical process logic attacks, the detection accuracy of the intrusion detection method reaches 90%, which can effectively demonstrate that the intrusion detection method based on the PSO-H-SVM state classification model and response mechanism can effectively detect the physical process logic attack against the robotic arm and mitigate the damage caused by the attack.

5. Discussion and Conclusions

In this paper, the current security situation of industrial robotic arms is analyzed and the possible physical process logic attacks on robotic arms during closure operations are discussed. For physical process logic attacks, we have established an intrusion detection method based on the PSO-H-SVM state classification model and response mechanism. The PSO-H-SVM algorithm is an improvement on the traditional SVM algorithm, and the specific construction process of the algorithm model and its performance after improvement are described in detail. The workflow of the intrusion detection method and response mechanism are also specifically described in the paper. Finally, we built a physical experimental platform in the laboratory to verify the effectiveness of the intrusion detection method proposed in the paper. The experimental results show that the PSO-H-SVM algorithm achieves an accuracy of 96.02% for the classification of five classes of operating states of the robotic arm; for the physical process logic attack, the detection accuracy of the intrusion detection method reached 90%, which has a good intrusion-detection effect. In addition, our proposed response mechanism is also verified in the experiment: when there is an intrusion, the alarm, protection measures and log recording functions can operate normally, which can effectively protect the robotic arm system from serious harm. Therefore, the proposed response mechanism is effective.

Similarly, there are still some shortcomings in this paper: (1) The intrusion detection method in this study is for specific application scenarios, and will be optimized for different application scenarios at a later stage; (2) All the methods and tests in this paper are based on experimental environments, and it is necessary to test them in real industrial operating environments; (3) The operation of the robotic arm is a dynamic process, and the analysis of the dynamics will be added in a later study to improve the accuracy of intrusion detection.

Nowadays, many scholars have extended their research on industrial control system security to many aspects, among which the identification of internal faults and network security attacks is one of the most important. In this paper, we mainly study the content of intrusion detection, and at a later stage, we will also include research content on the distinction between the faults and attacks of specific robotic arm systems in our work plan.

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

Funding: This research was funded by the National Key R&D Program of China, grant number 2018YFC2001700, and the Beijing Natural Science Foundation, grant number L192005.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Sobhan, N.; Shaikat, A.S. Implementation of Pick & Place Robotic Arm for Warehouse Products Management. In Proceedings of the 2021 IEEE 7th International Conference on Smart Instrumentation, Measurement and Applications (ICSIMA), Bandung, Indonesia, 23–25 August 2021.
2. Yenorkar, R.; Chaskar, U.M. GUI Based Pick and Place Robotic Arm for Multipurpose Industrial Applications. In Proceedings of the 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 14–15 June 2018.
3. Silva, M.Z.; Brito, T.; Lima, J.L.; Silva, M.F. Industrial Robotic Arm in Machining Process Aimed to 3D Objects Reconstruction. In Proceedings of the 22nd IEEE International Conference on Industrial Technology (ICIT), Valencia, Spain, 10–12 March 2021.
4. Vihonen, J.; Mattile, J.; Visa, A. Joint-Space Kinematic Model for Gravity-Referenced Joint Angle Estimation of Heavy-Duty Manipulators. *IEEE Trans. Instrum. Meas.* **2017**, *66*, 3280–3288. [CrossRef]

5. Ding, D.; Han, Q.L.; Xiang, Y.; Ge, X.; Zhang, X.M. A survey on security control and attack detection for industrial cyber-physical systems. *Neurocomputing* **2018**, *275*, 1674–1683. [CrossRef]
6. Haddadin, S.; De Luca, A.; Albu-Schäffer, A. Robot Collisions: A Survey on Detection, Isolation, and Identification. *IEEE Trans. Robot.* **2017**, *33*, 1292–1312. [CrossRef]
7. Li, L.; Xie, L.; Hao, B.; Yang, L.; Hu, T.; Wang, Z. Data Logic Attack on Heavy-Duty Industrial Manipulators. *IEEE Access.* **2020**, *8*, 17419–17433. [CrossRef]
8. Kim, T.; Kim, C.H.; Choi, H.; Kwon, Y.; Xu, D. Revarm: A Platform-Agnostic Arm Binary Rewriter for Security Applications. In Proceedings of the 33rd Annual Computer Security Applications Conference, Orlando, FL, USA, 4–8 December 2017.
9. Jeong, S.; Choi, I.; Kim, Y.; Shin, Y.; Kim, K. A Study on ROS Vulnerabilities and Countermeasure. In Proceedings of the Companion of the 2017 ACM/IEEE International Conference on Human-Robot Interaction, Vienna, Austria, 6–9 March 2017.
10. Li, W.; Xie, L.; Deng, Z.; Wang, Z. False sequential logic attack on SCADA system and its physical impact analysis. *Comput. Secur.* **2016**, *58*, 149–159. [CrossRef]
11. Zhu, G.; Yuan, H.; Zhuang, Y.; Guo, Y.; Zhang, X.; Qiu, S. Research on Network Intrusion Detection Method of Power System Based on Random Forest Algorithm. In Proceedings of the 13th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA), Beihai, China, 16–17 January 2021.
12. Liu, Z.; Yin, X.; Hu, Y. CPSS LR-DDoS Detection and Defense in Edge Computing Utilizing DCNN Q-Learning. *IEEE Access.* **2020**, *8*, 42120–42130. [CrossRef]
13. Li, H.; Chasaki, D. Ensemble Machine Learning for Intrusion Detection in Cyber-Physical Systems. In Proceedings of the IEEE INFOCOM 2021—IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Vancouver, BC, Canada, 10–13 May 2021.
14. Yimeng, D.; Gupta, N.; Chopra, N. On content modification attacks in bilateral teleoperation systems. In Proceedings of the 2016 American Control Conference (ACC), Boston, MA, USA, 6–8 July 2016.
15. Albu-Schäffer, A.; Ott, C.; Hirzinger, G. A Unified Passivity-based Control Framework for Position, Torque and Impedance Control of Flexible Joint Robots. *Int. J. Robot. Res.* **2007**, *26*, 23–39. [CrossRef]
16. Shang, W.; Xie, F.; Zhang, B.; Cong, S.; Li, Z. Adaptive Cross-Coupled Control of Cable-Driven Parallel Robots With Model Uncertainties. *IEEE Robot. Autom. Lett.* **2020**, *5*, 4110–4117. [CrossRef]
17. Park, J.H.; Uhm, T.Y.; Bae, G.D.; Choi, Y.H. Stability Evaluation of Outdoor Unmanned Security Robot in Terrain Information. In Proceedings of the 2018 18th International Conference on Control, Automation and Systems (ICCAS), PyeongChang, Korea, 17–20 October 2018.
18. Singh, P.K.; Singh, R.; Nandi, S.K.; Ghafoor, K.Z.; Nandi, S. An efficient blockchain-based approach for cooperative decision making in swarm robotics. *Internet Technol. Lett.* **2019**, *3*, 140–145. [CrossRef]
19. Azzalini, D. Modeling and Comparing Robot Behaviors for Anomaly Detection. In Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS), Auckland, New Zealand, 9–13 May 2020.
20. Li, X.J.; Shen, X.Y. A Data-Driven Attack Detection Approach for DC Servo Motor Systems Based on Mixed Optimization Strategy. *IEEE Trans. Industr. Inform.* **2020**, *16*, 5806–5813. [CrossRef]
21. Bezemskij, A.; Loukas, G.; Gan, D.; Anthony, R.J. Detecting Cyber-Physical Threats in an Autonomous Robotic Vehicle Using Bayesian Networks. In Proceedings of the 2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Exeter, UK, 21–23 June 2017.
22. Guerrero, H.; Ángel, M.; Noemí, D.G.; Vicente, M. Detection of Cyber-attacks to indoor real time localization systems for autonomous robots. *Robot. Auton. Syst.* **2018**, *99*, 75–83. [CrossRef]
23. Pang, Z.H.; Liu, G.P.; Zhou, D.; Hou, F.; Sun, D. Two-Channel False Data Injection Attacks Against Output Tracking Control of Networked Systems. *IEEE Trans. Ind. Electron.* **2016**, *63*, 3242–3251. [CrossRef]
24. Clark, G.; Doran, M.; Glisson, W. A Malicious Attack on the Machine Learning Policy of a Robotic System. In Proceedings of the 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE), New York, NY, USA, 1–3 August 2018.
25. Li, W.; Xie, L.; Wang, Z. Two-Loop Covert Attacks against Constant-Value Control of Industrial Control Systems. *IEEE Trans Ind. Inform.* **2018**, *15*, 663–676. [CrossRef]
26. Khojasteh, M.J.; Khina, A.; Franceschetti, M.; Javidi, T. Learning-based attacks in cyber-physical systems. *IEEE Trans. Control. Netw. Syst.* **2020**, *8*, 437–449. [CrossRef]
27. Zhao, Z.; Huang, Y.; Zhen, Z.; Li, Y. Data-Driven False Data-Injection Attack Design and Detection in Cyber-Physical Systems. *IEEE Trans Cybern.* **2020**, *51*, 6179–6187. [CrossRef] [PubMed]
28. Hector, J.B.; Katsiaris, P.; Carey, N.E.; Cote, N.; Rawat, D.B. On the Security of Cyber-Physical Robotic Systems Using Dynamic Modeling and Simulation. In Proceedings of the 2021 IEEE International Conference on Communications Workshops (ICC Workshops), Montreal, QC, Canada, 14–23 June 2021.
29. Tang, Y.; Zhang, D.; Ho, D.W.C.; Yang, W.; Wang, B. Event-Based Tracking Control of Mobile Robot With Denial-of-Service Attacks. *IEEE Trans. Syst. Man. Cybern.* **2020**, *50*, 3300–3310. [CrossRef]
30. Hong, J.; Liu, C.; Govindarasu, M. Integrated Anomaly Detection for Cyber Security of the Substations. *IEEE T Smart GRID.* **2014**, *5*, 1643–1653. [CrossRef]

31. Khaitan, S.K.; McCalley, J.D. Design Techniques and Applications of Cyberphysical Systems: A Survey. *IEEE Syst J.* **2015**, *9*, 350–365. [CrossRef]
32. Ali Alheeti, K.M.; Al-Zaidi, R.; Woods, J.; McDonald-Maier, K. An intrusion detection scheme for driverless vehicles based gyroscope sensor profiling. In Proceedings of the 2017 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 8–10 January 2017.
33. Zhou, C.; Hu, B.; Shi, Y.; Tian, Y.C.; Li, X.; Zhao, Y. A Unified Architectural Approach for Cyberattack-Resilient Industrial Control Systems. *P IEEE.* **2021**, *109*, 517–541. [CrossRef]
34. Zhou, C.; Huang, S.; Xiong, N. Design and Analysis of Multimodel-Based Anomaly Intrusion Detection Systems in Industrial Process Automation. *IEEE Trans. Syst. Man. Cybern.* **2015**, *45*, 1345–1360. [CrossRef]
35. Akpinar, K.O.; Ozcelik, I. Anomaly Detection on EtherCAT Based Water Level Control Automation. In Proceedings of the 2020 5th International Conference on Computer Science and Engineering (UBMK), Diyarbakir, Turkey, 9–11 September 2020.
36. Akpinar, K.O.; Ozcelik, I. Analysis of Machine Learning Methods in EtherCAT-Based Anomaly Detection. *IEEE Access.* **2019**, *7*, 184365–184374. [CrossRef]
37. Qiu, C.; Shan, J.; Shandong, B. Research on intrusion detection algorithm based on BP neural network. *Int. J. Secur. Its Appl.* **2015**, *9*, 247–258. [CrossRef]
38. Maushart, F.; Prorok, A.; Hsieh, M.A.; Kumar, V. Intrusion detection for stochastic task allocation in robot swarms. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017.
39. Narayanan, V.; Bobba, R.B. Learning Based Anomaly Detection for Industrial Arm Applications. In Proceedings of the 2018 Workshop on Cyber-Physical Systems Security and PrivaCy–CPS-SPC '18, Toronto, ON, Canada, 15–19 October 2018.
40. Zihao, W.; Lan, L.; Zongyi, X.; Guangtao, C. The forecasting model of wheelset size based on PSO-SVM. In Proceedings of the 2019 Chinese Control And Decision Conference (CCDC), Nanchang, China, 3–5 June 2019.
41. Nguyen, T.T.; Nguyen, V.H.; Nguyen, X.H. Comparing the Results of Applying DE, PSO and Proposed Pro DE, Pro PSO Algorithms for Inverse Kinematics Problem of a 5-DOF Scara Robot. In Proceedings of the 2020 International Conference on Advanced Mechatronic Systems (ICAMechS), Hanoi, Vietnam, 10–13 December 2020.
42. Shi, Y.; Eberhart, R.C. Empirical study of particle swarm optimization. In Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), Washington, DC, USA, 6–9 July 1999.
43. Stockman, M.; Awad, M. Predicting microarchitectural power using Interval Based Hierarchical Support Vector Machine. In Proceedings of the 2010 International Conference on Energy Aware Computing, Cairo, Egypt, 16–18 December 2010.
44. Gaohui, L.; Jiakun, C. Research on Modulation Recognition of OFDM Signal Based on Hierarchical Iterative Support Vector Machine. In Proceedings of the 2020 International Conference on Communications, Information System and Computer Engineering (CISCE), Kuala Lumpur, Malaysia, 3–5 July 2020.
45. Dang-Ngoc, H.; Cao, T.N.M.; Dang-Nguyen, C. Citrus Leaf Disease Detection and Classification Using Hierarchical Support Vector Machine. In Proceedings of the 2021 International Symposium on Electrical and Electronics Engineering (ISEE), Ho Chi Minh, Vietnam, 15–16 April 2021.
46. Attanasio, A.; Marahrens, N.; Scaglioni, B.; Valdastri, P. An Open Source Motion Planning Framework for Autonomous Minimally Invasive Surgical Robots. In Proceedings of the 2021 IEEE International Conference on Autonomous Systems (ICAS), Montreal, QC, Canada, 11–13 August 2021.
47. Lawal, M.A.; Shaikh, R.A.; Hassan, S.R. Security Analysis of Network Anomalies Mitigation Schemes in IoT Networks. *IEEE Access.* **2020**, *8*, 43355–43374. [CrossRef]
48. Brinkman, A.; Morris, J.; Chen, I.; Sheikh, N.; Warren, P. Fastcat: An Open-Source Library for Composable EtherCAT Control Systems. In Proceedings of the 2021 IEEE Aerospace Conference (50100), Big Sky, MT, USA, 6–13 March 2021.
49. Pan, H.; Xie, L.; Lv, Z.; Li, J.; Wang, Z. Hierarchical support vector machine for facial micro-expression recognition. *MTAP* **2020**, *79*, 1–15. [CrossRef]
50. Koizumi, Y.; Murata, S.; Harada, N.; Saito, S.; Uematsu, H. SNIPER: Few-shot Learning for Anomaly Detection to Minimize False-negative Rate with Ensured True-positive Rate. In Proceedings of the ICASSP 2019–2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019.