Task Aware Intrusion Detection for Industrial Robots

Davide Avanzi[1], Stefano Longari[1], Mario Polino[1],
Michele Carminati[1], Andrea Maria Zanchettin[1], Mara Tanelli[1] and Stefano Zanero[1]

[1]Politecnico di Milano, Dipartimento di Elettronica, Informazione e Bioingegneria, Milan, Italy

Abstract

Industrial Cyber-Physical Systems (CPSs), like any computer system, are vulnerable to cyberattacks that may endanger the safety of personnel on automated production lines. This paper studies IDSs for industrial robots that model their legitimate behavior through statistical and ML techniques and detect anomalies as deviations from the learned norm. We validate our approach against attacks in a simulated and real environment, demonstrating its effectiveness in detecting anomalous behaviors during the robot operation.

Keywords

intrusion detection, anomaly detection, industrial robots

1. Introduction

The fast adoption and large-scale usage of industrial collaborative robots [1] inside manufacturing plants and other critical sectors, combined with their increasing interconnection to external networks, including Internet-based cloud services, has increased their attack surface, raising the risks of economical and safety damage [2, 3, 4, 5]. Most of the current state-of-the-art research on cyberattacks detection in industrial environments are related to Supervisory Control And Data Acquisition (SCADA) systems [6, 7, 8, 9, 10]. Only a few works focus on industrial robots, mainly detecting critical hardware faults [11, 12] or intentional attacks [13] that degrade the production [14] or compromise the workers' safety. Instead of detecting the potential attack vectors, which are multiple, diverse, and already explored in the literature [14], in this work, we focus on their *physical effects* on the manufactured products. Focusing on the physical goals of the attacker, as opposed to the digital vector, as advocated in [15, 16], makes our approach general and independent from the specific attack vector. First, we define novel attacks against industrial robots, which are not detected by current anomaly detection techniques. Then, we suggest a novel anomaly detection approach that effectively detects them along with current state-of-the-art attacks based on the legitimate task performed by the industrial robot, which is known a priory since usually deployed in an assembly line. We define two different *operational scenarios* that define the conditions under which the robot operates and must be considered in the choice of the anomaly detection model. In the *first* scenario, the robot repeatedly operates a single task over time. Hence, we use a statistical Seasonal AutoRegressive Integrated Moving Average (SARIMA) model to predict its behavior and

ITASEC 2023: The Italian Conference on CyberSecurity, May 03–05, 2023, Bari, Italy

davide.avanzi@mail.polimi.it (D. Avanzi); stefano.longari@polimi.it (S. Longari); mario.polino@polimi.it (M. Polino); michele.carminati@polimi.it (M. Carminati); andreamaria.zanchettin@polimi.it (A. M. Zanchettin); mara.tanelli@polimi.it (M. Tanelli); stefano.zanero@polimi.it (S. Zanero)

perform anomaly detection. In the **second** scenario, the robot interacts with a more complex environment and changes its operation based on external conditions: in this case, we use a model based on Long Short-Term Memory Network (LSTM) Recurrent Neural Network (RNN) that predicts the legitimate behavior during production and detect anomalies as deviations from the learned norm. We evaluate our approach against novel and known attacks that introduce both spatial and temporal anomalies in simulated and real scenarios: we study the detection performance in terms of the speed and spatial thresholds under which attacks are identified. Results indicate that our approach is effective in learning the correct robot behavior and timely detecting the attacks within seconds of their introduction and low error rates. This research provides the following contributions:

- We design three novel attacks, not detected by state-of-the-art solutions, that modify the manufacturing process leading to the introduction of defective products.
- We propose an IDS comprising two different learning models capable of detecting the attacks developed in this research and current state-of-the-art attacks, demonstrating its effectiveness.

2. Related Works

Classical IDS for Industrial Control System (ICS) perform anomaly detection at process [17, 18] or network [19, 20, 21] level, trying to detect anomalous activity in the "cyber" part of the system as an indicator of a potential attack. A different approach consists of monitoring the behavior of the physical system to detect the effects of an attack. Recent research makes use of various signal processing [9] and machine learning techniques, such as Recurrent Neural Network (RNN) [22, 6], Convolutional Neural Networks (CNNs) [23, 7], Long Short-Term Memory Networks (LSTMs) [24, 25] and autoencoders [26, 8, 27, 28] to perform anomaly detection on large ICS for various types of cyberphysical systems, amongst which SCADA systems.

Cheng et al. [11] use Convolutional Variational Autoencoders (CVA) to learn long-term patterns and dependencies between collected variables (time series) in the robotic arm operation. They collect positional and electrical information and use it as input for the prediction model. The model is trained on normal operation data and used to perform predictions in an online anomaly detector to detect mechanical faults in the robotic arm.

Munawar et al. [12] propose an approach based on imaging to reduce the human labor and safety risk involved in directly monitoring industrial robots during their operations. They use a monocular camera to obtain a video feed of the operating area, then use a deep CNN for initial feature extraction, followed by an LSTM RNN to predict the following frame. If the prediction does not match the observed data, an anomaly is detected and flagged to the human supervisor. The underlying assumption is that the robot performs a repetitive task, and its normal behavior can be learned in advance. This approach focuses on detecting strong anomalies, such as sudden changes in environmental conditions or large changes in the robot movement, and is only tested for such cases. Therefore, it may not be suitable for detecting subtle attacks introduced by humans.

One of the most concerning types of attack on a production line is the introduction of micro-defects (product faults of small enough scale not to be noticed by supervising human operators but sufficient to cause critical failures when the product is used). Early, automated defect detection is the aim of several works that apply machine learning techniques to images of the manufactured product but the effectiveness against micro-defects inserted on purpose has not been proven yet.

Narayanan, et al. [13] propose an anomaly detection approach for industrial arm applications for detecting micro-defects introduced by compromised Additive or Subtractive Manufacturing robots. They assume that the robot is performing a single automated repetitive task and introduce the concept of a "tolerance envelope" (a fixed threshold relative to the programmed path that the robot cannot exceed during its movement). Their approach uses a Support Vector Machine (SVM) classifier to separate anomalies from benign data. This approach effectively detects small spatial distortions in the path performed by industrial robots but does not include a timing element regarding the robot's operation. This lack of time component in the classification model prevents this anomaly detection model from detecting two types of novel attacks proposed in this work.

3. Threat model

Threat modeling is a necessary step of any security evaluation, especially when focusing on cyber-physical systems [29]. In this work, we focus on identifying and preventing attacks that alter the movement of industrial robots, potentially causing physical damage or defective products in an Industry 4.0 automated production line. Two main types of attackers [30, 31] are considered: internal attackers who have direct access to the robot controller and HMI and external attackers who exploit vulnerabilities in the ICS network to gain access. The worst-case scenario is assumed, where the controller is fully compromised, making it impossible for IDS to rely on data from the controller. In particular, we mainly consider *production sabotage* attacks, which result in defective products, causing economic loss and potential harm to users.

One specific concern is micro-defects, small faults that can lead to critical failures but are unnoticeable by human operators.

Belikovetsky et al. [32] provide an example of how micro-defects can cause major failures in the specific case of drone propeller blades.

Conventionally, defects are found after production by carefully examining samples. This is both time-consuming and costly for both the company and its customers. Early detection of defects during the production process would benefit the company.

We introduce three attacks:

Attack 1: Alteration of the robot movement. It involves altering the robot's firmware or parameters to affect its accuracy or movement. This can cause the robot to follow a different path or perform tasks at a different velocity, leading to defects. For example, a welding robot could introduce a product defect by performing the welding operation at a higher velocity than the one designed to set up the task while still following the programmed path. The Narayanan et al. [13] approach can detect spatial deviations from programmed paths, but not velocity alterations.

Attack 2: Alteration of the cycles of a sub-task. An attack on a robot operating repeated actions (such as applying multiple coats of paint), which alters the number of times the path is followed, leading to production defects. This is not detectable by a spatial-only anomaly detector as the robot follows the correct path but an incorrect number of times.

Attack 3: Alteration of the robot logic. It involves altering the robot's logic that uses external signals for decision-making. For example, a drilling robot could perform a different path based

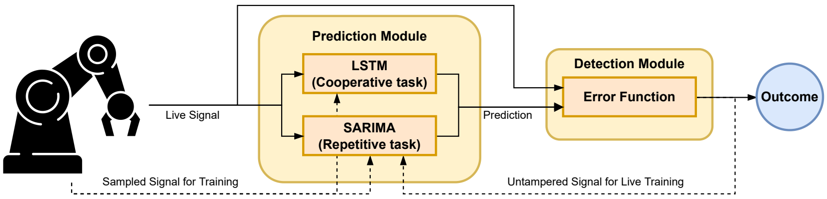


Figure 1: High level diagram of the overall approach of our IDS. Data collected are all joint angle positions: *rax_1* up to *rax_6*

on the type of product coming from the conveyor belt, and the type of product is determined by an external sensor, sending the signal to the robot through the network or a physical interface. An attacker can change the controller logic to cause the robot to perform the wrong task, leading to defects.

4. Approach

Our approach consists of a prediction module and a detection module, as shown in Figure 1. The prediction module builds a prediction from the nominal data in the form of time series of the position of the end effector of the robot during the robot's task execution, alongside the angular position of each of the joints of the robot calculated through an inverse kinematic function. The detection module produces a prediction error and identifies anomalous behavior.

Our approach is mostly unsupervised and uses nominal movement data to create a behavioral model for the untampered system. Anomalous behavior is identified by comparing the prediction with live operation data and flagging a movement that differs from the expected value by means of an error function.

The prediction module has two task-dependent sub-modules: SARIMA for repetitive tasks and LSTM for cooperative networked environment tasks.

Scenario 1: Repetitive Task. The robot performs a single repetitive task [13] (e.g., on a production line with a conveyor belt through a series of robots). An attacker – performing the first two attacks described in the previous section – aims to sabotage the accuracy of one or more robots, leading to defective products. An anomaly is detected when a robot's behavior deviates from its programmed behavior.

Scenario 2: Cooperative Networked Environment. It involves a networked environment with multiple robots and industrial devices performing coordinated tasks by exchanging information and signals. The robot's behavior can change based on external signals, and the task performed alternates between pre-defined tasks. An attacker – performing the third attack described in the previous section – could introduce defects in production by altering the network's coordinating logic.

The main concept behind our unsupervised approach is to use data obtained by logging the correct movement of the robot to learn a correct behavioral model. This model is then used to perform predictions during the live operation of the robot. For each of the two operational scenarios identified, we propose a specific model that learns the robot's behavior and performs predictions of its movement. For the first scenario (repetitive task), we leverage the repetitive nature of the robot task to learn its movement over time: in this case, we use a Seasonal AutoRegressive statistical model [33], which can be seen as a sub-model of the Seasonal AutoRegressive Integrated Moving Average (SARIMA) model. In the second scenario (cooperative networked environment), the overall task is not identically repeated over time, hence the robot behavior is not seasonal anymore: the SAR model cannot correctly learn the robot expected behavior in this context. For this scenario, we use a Long Short-Term Memory Network (LSTM) RNN, well known for its capability to learn long-term dependencies and behaviors in time series datasets [34].

Prediction Module for Repetitive Task: SAR(IMA). The core concept behind a SARIMA model is to predict future values of a time series as a function of its past values (lags) and prediction errors, while also including a seasonal component capable of addressing the seasonality of a time series. The model is composed of a non-seasonal component (a standard ARIMA model) and a seasonal part that consists of the same component delayed in time by the seasonal parameter. A SARIMA model is defined as follows: $SARIMA\ \underbrace{(p,d,q)}_{\text{non-seasonal}}\ \times \underbrace{(P,D,Q)_s}_{\text{seasonal}}$, where the p parameter is the order of the AR component, d is the parameter of the Integrator component, and q is the order of the MA component. P, D, and Q are the orders of the relative component lagged by the seasonality parameter s.

However, in order to accurately predict the signal of each robot axis, we do not require the full SARIMA model: given the stationary [35] and constant seasonal nature of the analyzed signal, we only require the Seasonal AutoRegressive (SAR) component of the SARIMA model. The complete model is then simplified to a $SARIMA(p,0,0)\times(P,0,0)_s$

The main benefit of a Seasonal AR model increases when the analyzed signal has a long seasonality and its seasonal component is predominant with respect to the other components. This is the case of the signals we collect from the industrial robot: the signal is periodic with close to zero variations from one cycle to the next.

By including the lags at the period of the signal, the model is able to learn its seasonality and perform a long prediction of a whole cycle. Signals with long periods would require very large AR models, which are not efficient to train. With a SAR model, we are able to include only the useful parameters at time lag $t-s$ and train the model with few parameters.

Therefore, for our first operational scenario, we choose to use a $SARIMA(1,0,0)\times(1,0,0)_s$ model to learn the robot's movement and perform predictions:

$\hat{y}_t = c + \theta_1 y_{t-1} + \Theta_1 y_{t-s} + \varepsilon_t$

The most straightforward approach to build our model would be to perform a single initial prediction of a complete signal cycle. The prediction obtained through this cycle would then be repeated in a continuous cycle to build the full prediction signal. However, this approach is not feasible in our scenario since the exact period of the signal is not necessarily an exact multiple of the sampling time. For this reason, an error - smaller than the sampling time - is introduced each cycle, leading to a drift between the real signal and the repeated predicted cycle, invalidating the comparison. To solve this issue, we perform a new prediction for each cycle of the robot, re-training the SARIMA model on a **sliding window** on the observed dataset, ending at the end of the last observed cycle. A sliding window approach is to be preferred to an expanding window because in that case, the training dataset could become very large after long productions. A very long dataset of a repeated signal would make the training less efficient while not providing much more information to the trained model. After training the model on the current sliding window, we perform the prediction for the next cycle and then move the window ahead of one cycle, for the prediction of the next.

The introduction of a sliding window as a source of the training set for the SARIMA model opens the possibility of the model learning anomalies as they happen and thus forgetting the correct behavior of the robot. While we assume that anomaly detection in the robot operation would bring human intervention to restore the correct status of the system, a model learning the anomaly could also be exploited by an aggressor to perform an attack where a tiny drift is slowly introduced over time. To evaluate this, we test the model in Experiment 2 against an attack of that type.

Prediction Module for Cooperative Networked Environment: LSTM. Recurrent Neural Networks are a type of neural network introduced to overcome the lack of information persistence through the time of traditional neural networks. While Recurrent Neural Networks (RNNs) are effective in considering recent information for each prediction, their performances worsen when the relevant information to be considered is further away in the past with respect to the current prediction time. To solve this issue, Hochreiter and Schmidhuber[24] introduced LSTM networks, a kind of RNN network specifically designed to learn long-term information dependencies. In our specific instance, to predict the correct behavior of the robot in the second scenario of the threat model 3, our model requires learning complex dependencies between the different variables we collected. To achieve this, we use LSTM in a multivariate and stacked configuration. A multivariate model takes in multiple input variables to perform predictions of one variable. This multivariate model learns the behavior of each robot axis, taking into account not only its past values but also the values of the other positions of the axes and the value of the input signal: this allows the model to detect when the robot movement does not reflect the provided input based on the initially learned behavior. The network topology of our model consists of a single main layer of LSTM neurons with a one-neuron output linear activation layer. The network weights are trained once for each robot axis, and each model is used to predict the relative axis behavior on the test set. To train the model, the complete dataset is split into training and test sets, where the training set does not contain anomalies. To perform a multivariate batch training of the model, we follow common data preparation techniques [36] that normalize and reshape the training time series set into a three-dimensional array that is needed for the LSTM model input [37].

Detection Module. For each axis of the robot, the values from the predictions dataset are compared to the observed ones using the sample-wise Root Mean Squared Error (RMSE). RMSE reduces noise influence while emphasizing larger variations. During training, the anomaly threshold is determined by comparing the non-anomalous live data to the predictions. The anomaly threshold is used to classify samples as anomalous or not based on whether they exceed it. The overall anomaly score of a sample is determined by summing the anomalous axes. Finally, the sample is classified as benign or anomalous by comparing its anomaly score to a user-defined threshold. In the first scenario, the process is repeated by considering whole cycles and averaging the prediction error function along the cycle for easier inspection and intervention on anomalous flagged cycles.

5. Experimental Evaluation

Our validation aims to prove the efficacy of our proposed attack detection models and compare them to the state of the art. We also aim to determine, where possible, the threshold levels under which our models do not reliably detect possible micro-scale attacks. We conduct 5 experiments, including spatial attack detection (experiment 1), progressive velocity alteration attack detection (experiment 2), anomaly detector against altered sub-task cycles (experiment 3), detection system against logic tampering attacks (experiment 4), and a comparison with a state-of-the art approach (experiment 5). Tests are performed in a simulated environment and on a real industrial robot, with a comparison of results and a discussion of differences.

Experimental Setup.

We conducted experiments on an ABB IRB 140 industrial robot [1] connected to an IRC5 controller. This robot is widely used in industrial operations due to its millimetric precision. The same robot has been the subject of a seminal experimental security analysis [14] which revealed multiple vulnerabilities at the time. All simulations were run using RobotStudio 2020.1.1 and the virtual IRC5 controller used RobotWare 6.10.02.00.

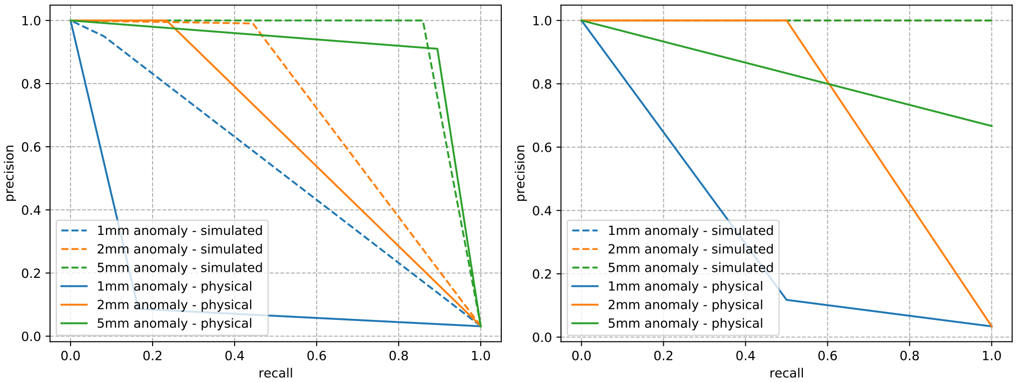
Dataset. The data fed to the detection system is logged directly into the robot controller memory. Each dataset consists of six columns, one for each axis value, a column for the anomaly status, and one for the simulated input signal (only for the 4th experiment). The sampling frequency, limited by the robot controller's computational constraints, is of 10Hz. The dataset consists largely of a normal, non-anomalous robot operation. After a predefined period of time we introduce an anomaly, depending on the experiment, and then terminate the robot operation, as would happen in real-life production environment operating under our assumptions. In a real world scenario, data collection would happen via a dedicated device directly wired to the robot axis sensors, as the robot controller is to be considered compromised and not reliable. [14]

Evaluation Metrics. The anomaly detection in our IDS is performed as a binary classification problem. At any given point in time, based on the aforementioned techniques, our IDS classifies the robot position as anomalous or not. For each experiment, we present the results as a confusion matrix where we can compare prediction outcomes and actual values from the robot, obtaining true and false positives and negatives. Due to the imbalanced nature of the datasets, we expect to have a very high amount of negative class elements (robot operating in normal conditions) with respect to the positive class elements (anomalies). To further analyze the classificator performances, we include more binary classification performance indexes specifically effective in evaluating unbalanced datasets. Specifically, for each experiment we provide its balanced accuracy, F1 score, and Matthews Correlation Coefficient (MCC).

5.1. Experiment 1: detection of spatial attacks

The task executed by the robot in this experiment is a pick and place operation repeated over time. We introduce an anomaly in the task by altering the location where the place event takes place. This attack is one of the most common experiments in current research [14] [32] [13]. To compare our system with the current state of the art, we perform the attack multiple times, varying the entity of the deviation in order to understand what is the minimum deviation for the IDS to detect the attack. Starting from a 504mm long untampered movement, we test variations of 1mm, 2mm, and 5mm. The attack dataset consists of a log of the execution of the correct task for 120 cycles, followed by the anomalous task for 5 cycles. We execute the test both in the simulation environment and on the actual robot. As discussed in our approach, for repetitive

[1] https://global.abb/



(a) Sample-by-sample

(b) Cycle-by-cycle

Figure 2: Precision-Recall curves for the different anomalies in experiment 1, both in the simulated environment and with the physical robot.

Table 1

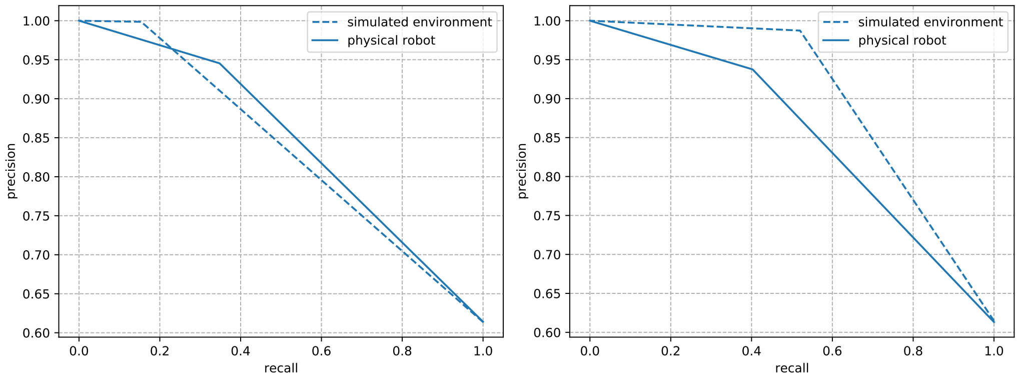
Experiment 1 - Confusion matrix values and metrics, divided in Sample-by-Sample and Cycle-by-Cycle tests. Time To Detection (TTD) is represented in seconds for the Sample-by-Sample IDS while in nuber of cycles for the Cycle-by-Cycle IDS.

Exp. 1 SARIMA:	**Sample-by-Sample Detection**						**Cycle-by-Cycle Detection**					
	Simulation			**Robot**			**Simulation**			**Robot**		
	1mm	2mm	5mm	1mm	2mm	5mm	1mm	2mm	5mm	1mm	2mm	5mm
TP	19	104	201	38	55	203	4	4	4	2	2	4
FP	1	1	0	402	0	20	0	0	0	15	0	2
FN	215	130	33	192	180	24	0	0	0	2	2	0
TN	7024	7024	7025	6684	7019	7069	115	115	115	99	113	112
F1 Score	.149	.613	.924	.113	.379	.902	1.0	1.0	1.0	.190	.667	0.8
MCC	.273	.657	.925	.079	.478	.899	1.0	1.0	1.0	.190	.701	.809
Bal. Acc.	.540	.722	.929	.554	.617	.945	1.0	1.0	1.0	.684	.75	.991
TTD	3.4 s	3.2 s	3.2 s	12.9 s	10 s	0 s	0 c	0 c	0 c	2 c	2 c	0 c

tasks we apply the SARIMA detection module. The rolling window training set size of the SARIMA model has been set to 5 cycles through empirical performance analysis.

Results. In table 1 we report the results of Experiment 1. For each task we present the values of the confusion matrix and the evaluation metrics mentioned before, on top of the Time To Detection (TTD) expressed in seconds or number of cycles for the sample-by-sample and cycle-by-cycle detection methods respectively.

The confusion matrices reported in Table 1 suggest that the model is capable of detecting anomalous samples with increasing accuracy on more significant anomalies. It is important to notice that the majority of false classifications happen as anomalous events that are not recognized (false negatives), which are mostly found in blocks of malicious data, as proven by the fact that the cycle-by-cycle detection process outperforms the sample-by-sample one, since small amounts of false negatives in a cycle mostly composed of true positives is still classified as anomalous. Overall, as also evident from the precision-recall curves for all the anomalies presented in Figures 2a,2b, performances for 5mm tests are optimal, for 2mm may still be useful depending on the implementation



(a) Sample-by-sample

(b) Cycle-by-cycle

Figure 3: Precision-Recall curves for the different anomalies in experiment 2, both in the simulated environment and with the physical robot.

Table 2
Experiments 2 and 3 - Confusion matrix values and metrics, divided in Sample-by-Sample and Cycle-by-Cycle tests. Time To Detection (TTD) is represented in seconds for the Sample-by-Sample IDS while in number of cycles for the Cycle-by-Cycle IDS. The Speed Change Before Detection, relevant for Experiment 2, is represented in mm/s.

	Exp. 2 - SARIMA				**Exp. 3 - SARIMA**				**Exp. 4 - LSTM**	
	Sample-by-Sample		**Cycle-by-Cycle**		**Sample-by-Sample**		**Cycle-by-Cycle**			
	Sim.	**Robot**	**Sim.**	**Robot**	**Sim.**	**Robot**	**Sim.**	**Robot**	**Sim.**	**Robot**
TP	1047	3219	78	60	1266	668	4	4	331	330
FP	2	186	1	4	0	100	0	1	6	7
FN	7727	6033	72	89	20	757	0	0	18	32
TN	5748	5628	93	90	38842	42489	110	109	1524	1622
F1 Score	0.267	0.509	0.681	0.563	0.992	0.612	1	0.889	0.96386	0.94344
MCC	0.256	0.368	0.530	0.3982	0.992	0.633	1	0.889	0.95632	0.93239
Bal. Acc.	0.577	0.658	0.755	0.680	0.992	0.735	1	0.890	0.97162	0.953535
TTD	812.1 s	580.1 s	91 c	90 c	0.1 s	0.3 s	1 c	0.995 c	0.51 s	0.75 s
SCBD (mm/s)	2.6	1.8	1.8	1.8	-	-	-	-	-	-

case, while 1mm ones do not perform well, especially when testing the real world robot scenario.

5.2. Experiment 2: detection of stealthy attacks

The task performed in experiment 2 is the same pick and place in experiment 1. The stealthy anomaly is represented by a slow increase in the robot operating speed, while mantaining the correct path.

A detection system that only considers the physical location of the end effector of the robot would therefore not be capable of detecting such anomaly.

For this experiment, the attack dataset is composed of 100 cycles of correct behavior followed by 100 cycles where the base speed of the robot (200mm/s) is increased by 0.1mm every 5 cycles. Again, for repetitive tasks we use the SARIMA detection module and the window training set is 5 cycles long.

Results. In Table 2 we report the results of Experiment 2. For each task we present the values of the confusion matrix and the evaluation metrics mentioned before, on top of the Time to Detection (TTD) and Speed Change Before Detection (SCBD).

We implemented this attack specifically to challenge our sliding window-trained SARIMA model, as evident from the results. We observe a large number of False Negatives, that correspond to the initial period in which the attack is introduced but the increase in movement is minor. However, once the anomaly increases over a threshold it is correctly and consistently detected. For this reason in Table 2 amongst the metrics for this experiment we show also the speed change before detection.

Despite being designed to be undetected and slip through the rolling window training, we argue that this anomaly is still detected by the model because its constant change in speed introduces a long-term variation in the overall duration of a single cycle. Being the cycle period estimated only once from a set of non-anomalous cycles, its value is fixed in time and used during the prediction of each future cycle. Training the same model with a signal with a different seasonal component would then increase the prediction error, thus triggering the anomaly detection. In Figure 3a we report the precision-recall curves of the two analyzed sample-by-sample classifications. Lower performance metrics are explained by the long time it takes to detect the anomaly during which the predictions result in a lot of False Negatives. Similarly to the previous experiment, averaging the prediction to entire cycles improves overall detection metrics (as visible also by comparing the precision-recall curves in Figures 3a,3b), although the speed change before detection does not significantly change in the real world robot scenario.

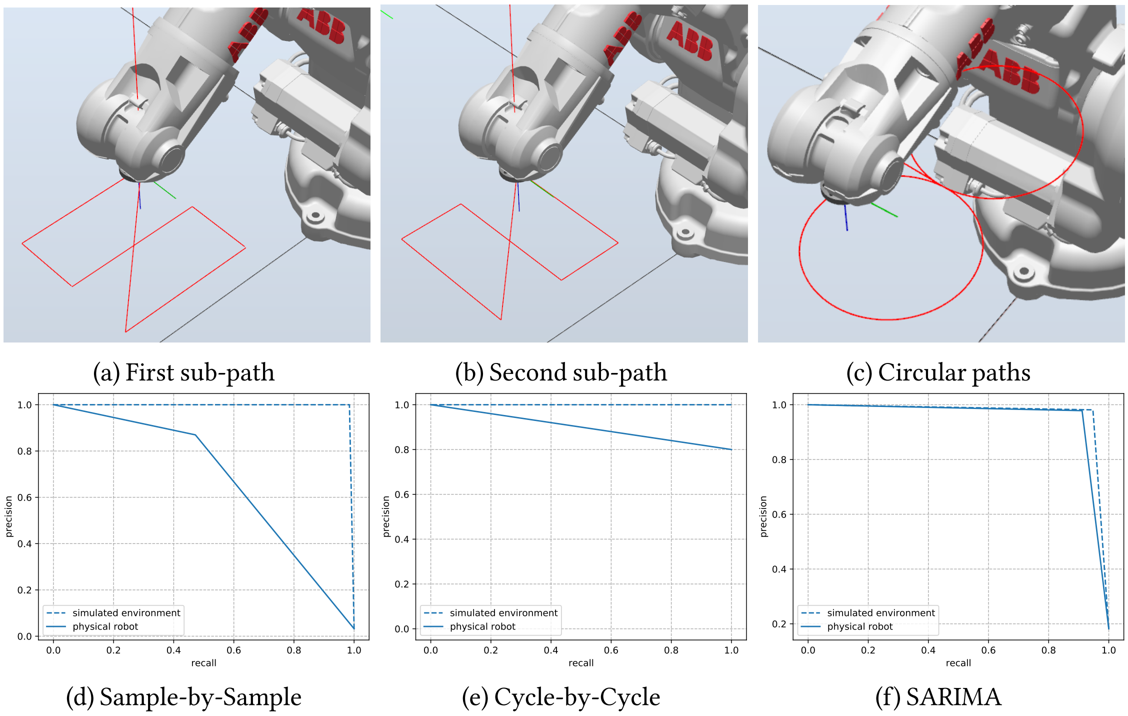
5.3. Experiment 3: detection of attacks against sub-task cycles

Experiment 3 is composed of the repetition of two paths, each one simulating the complete covering of an area. The two paths differ in the way they cover such area, as visible in Figure 4a and Figure 4b. This type of patterns can be used for tasks such as to paint, sand, or polish an area. Similarly to Experiment 2, a detection system that only considers the physical location of the end effector of the robot would therefore not be capable of detecting such anomaly. For the non-malicious part of the dataset, in each cycle both paths are repeated exactly three times. The anomaly we introduced alters the number of times each path is executed, bringing them to two and four times respectively. The attack dataset is composed of 120 untampered cycles and 5 tampered ones.

Results. In Table 2 we report the results of Experiment 3. For each task we present the values of the confusion matrix and the evaluation metrics mentioned before, on top of the Time to Detection (TTD). Due to the longer task, the number of samples is significantly higher. In this experiment the anomaly is not a proper micro-defect, but a variation in how well the overall task is performed. This type of change in the path is easily detected by the IDS as shown by the metrics in Table 2. Even in the case of the real robot the anomaly is detected only 0.3 seconds after the first occurrence.

5.4. Experiment 4: detection of an attack against robot's logic

Such as Experiment 3, the task performed in this experiment consists of two sub-paths that are alternatively performed by the robot. In experiment 4, however, to simulate an external input to choose which sub-path the robot should execute, we use a simulated random input signal.



(a) First sub-path

(b) Second sub-path

(c) Circular paths

(d) Sample-by-Sample

(e) Cycle-by-Cycle

(f) SARIMA

Figure 4: In the first row, the paths performed in each iteration of the task in Exp 3 and Exp 4. In the second row, the Precision-Recall curves, both in the simulated environment and with the physical robot.

Due to this, the LSTM module is used for detection. The two sub-paths compose two circular shapes in different spatial positions, similarly to the behavior of drilling or cutting operations.

Results. We are not able to rely on a defined repetitive operation in this task that can be broken down in sequential cycles, so we have to rely upon the sample-by-sample anomaly detection only. As the other experiments, we provide confusion matrices and performance metrics for both the simulated environment and the physical robot experiment. Being LSTM based on a stochastic process during training, different training runs will result in slightly different models: to ensure a relevant result, we reproduce the training 30 times and provide their averaged result in the following section. Observing the confusion matrices in Table 2 and the precision-recall curves in Figure 4f, we observe that this model performs similarly in the classification of both the simulated environment and the physical robot, with a little difference in the number of False Positives. We argue that this similarity in the results, in contrast to what has been observed in the performances of the SARIMA model, is due to a greater generalization capability of the model, which focuses less on small variations. The lower generalization capability of the SARIMA model, instead, makes it more suitable for anomaly detection where robot task is repetitive. The values of the performance metrics in Table 2 and precision-recall curves in Figure 4f present very positive classification results.

Table 3
Comparison between our SARIMA approach and current state-of-the-art IDS Experiment 1 with a deviation of 5mm and on Experiment 2

	Experiment 1		Experiment 2	
	SARIMA	Narayanan, et al. [13]	SARIMA	Narayanan, et al. [13]
F1 Score	0.9241	0.9289	0.2669	0.00641
MCC	0.92463	0.3841	0.25562	0.01127
Bal. Acc.	0.92948	0.9152	0.57684	0.50088

5.5. Experiment 5: Comparison with the state of the art

We compared our model to the state-of-the-art by testing Narayanan et al's SVM-based anomaly detection approach [13] using their model[2] and datasets from Experiment 1 and 2. The purpose was to demonstrate that the state-of-the-art is not effective at detecting our introduced speed-changing attacks and show that our model performs similarly in spatial tasks. Table 3 shows our anomaly detector effectively detects the first attack (the same as in Experiment 1 with 5mm spatial deviation). Narayanan et al's IDS performs better in detecting smaller spatial anomalies. It is important to notice, however, that Narayanan et al.'s IDS better detects spatial anomalies with smaller deviations.

As expected, the state-of-the-art IDS is only able to spot variation in the spatial domain of the robot performed task. Because of this, its anomaly detection is missing variations in the velocity of the robot.

6. Limitations and Future Works

6.1. Limitations

In this chapter we will briefly explain the limitations we encountered during our research and the implementation of some parts of our approach. First of all, an industrial robot physically consists of a robotic arm that supports an end effector or actuator. This end effector is the tool that performs the action and defines the robot work (for example a welding attachment, a drill, mechanical pliers to grab and move objects) and can be greatly responsible for the good quality of the pieces it produces.

The enormous variety and heterogeneity of end effectors make comprehensive research unfeasible, so we opted to limit our research to the signals we can extract from the robot. This decision is also been made based on the lack of end effectors available to us to attach to the physical robot and its controller in our laboratory, making it impossible for us to back any simulation test with a real-world experiment. A complete IDS would need to also consider the operational parameters of the end effector, which is part of the overall industrial robot and it certainly is one of the possible targets that an attacker could exploit to introduce defects in the production. For example, an attacker able to access the functioning of a welding attachment could alter its welding temperature in order to make dangerous defective welds on critical equipment. Moreover, end effectors are produced by different manufacturers often with their own proprietary interfaces, which makes it more difficult to have a single data collection system

[2] https://github.com/narayave/mh5_anomaly_detector

without some dedicated adjustments or interfaces. Despite these difficulties, data collected from the end effectors with dedicated sensors could be treated as time series and analyzed alongside the ones we gather in our research to expand the industrial robot models.

Secondly, we implemented our own test tasks for the experiments in this research mainly because of a lack of examples from real production lines we could use. This absence of already made functioning industrial robot tasks can be explained by the limited demand for them in the research environment combined with the development efforts punt into their realization. On top of that, real industry programs are covered by intellectual property from the manufacturing companies, greatly limiting the availability of real industrial tasks. We were not able to obtain any complete task script to test, so we developed movement tasks that we deem realistic and based our experimental evaluation on those. These tasks are a small subset of all the tasks that industrial robots can perform and surely do not cover all the possible robot use cases.

Finally, with regards to our fourth experiment, we did not have an adequate network of robots at our disposal. Having to work with a single robot, we had to simulate the presence of an external signal within the robot controller itself, limiting the complexity of the system considered in the experiment. We were therefore not able to test the performance of our LSTM based anomaly detection model with more complex systems consisting of multiple robots and sensors working together. More tests are then needed to fully evaluate the proposed anomaly detection technique in this scenario.

6.2. Future Works

This work is part of a broader line of research on industrial robots security from the NECST Laboratory at Politecnico di Milano. Future works based on this research will start by dealing with the limitations outlined.

First of all, the proposed anomaly detection models need to be expanded to consider end effectors, by providing direct test examples and offering a more generalized approach, able to cover most types of end effectors. The models proposed in this research can be expanded with the inclusion of other time series datasets with little modifications. If the operating conditions of the two scenarios are maintained even in the end effector operations (e.g., cyclicity of the operation in the first scenario), the IDS should be able to be tested against anomalies of the same type as the ones analyzed in this work. To achieve this goal, more research needs to be carried out into the development of a common interface to gather data from different kinds of end effectors and merge them with the ones collected from the robot controller. Following the indications in Section 4, such a system should be implemented externally of the robot controller in order to prevent an attacker to alter them after compromising the controller. With the availability of a more complex production line testing environment, more research can be carried out on anomaly detection on robots interacting with each other or their environment. Furthermore, future works could also expand the experiments of this research with more advanced tasks obtained from real production lines.

Finally, more research can be done in evaluating more advanced and new machine learning or statistical models to improve the learning and prediction capabilities of the system behavior.

7. Conclusions

This paper presented an IDS for industrial robots that identifies abnormal behavior. The IDS has a prediction and detection module, including two predictors: a SARIMA model for repetitive tasks and an LSTM-based model for tasks in cooperative networked environments. We evaluated the system in two different scenarios (a simulated environment and a real robot) and found that it outperforms state-of-the-art detection systems. We demonstrated that, unlike our system, the current state of the art could not handle a set of attacks that modified the speed and not the spatial behavior of the robot. Our results were promising and showed the different capabilities of the system's predictors on their tasks. The tasks performed in our experiments mimic common industrial operations, and the introduced attacks simulate the introduction of potential micro-defects in the production outcomes. The results of our experiments suggest that this approach would be capable of detecting the effects real-world attacks on manufacturing robots.

References

[1] R. Bloss, Collaborative robots are rapidly providing major improvements in productivity, safety, programing ease, portability and cost while addressing many new applications, Industrial Robot: An International Journal 43 (2016) 463–468. URL: https://doi.org/10.1108/IR-05-2016-0148. doi:10.1108/IR-05-2016-0148.

[2] M. Pogliani, D. Quarta, M. Polino, M. Vittone, F. Maggi, S. Zanero, Security of controlled manufacturing systems in the connected factory: the case of industrial robots, Journal of Computer Virology and Hacking Techniques 15 (2019) 161–175. URL: https://doi.org/10.1007/s11416-019-00329-8. doi:10.1007/s11416-019-00329-8.

[3] A. Humayed, J. Lin, F. Li, B. Luo, Cyber-physical systems security – a survey, 2017. arXiv:1701.04525.

[4] N. Falliere, L. O. Murchu, E. Chien, W32. stuxnet dossier, White paper, Symantec Corp., Security Response 5 (2011) 29.

[5] D. U. Case, Analysis of the cyber attack on the ukrainian power grid, Electricity Information Sharing and Analysis Center (E-ISAC) 388 (2016).

[6] J. Goh, S. Adepu, M. Tan, Z. S. Lee, Anomaly detection in cyber physical systems using recurrent neural networks, in: 2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE), 2017, pp. 140–145. doi:10.1109/HASE.2017.36.

[7] M. Kravchik, A. Shabtai, Detecting cyber attacks in industrial control systems using convolutional neural networks, in: Proceedings of the 2018 Workshop on Cyber-Physical Systems Security and PrivaCy, CPS-SPC '18, Association for Computing Machinery, New York, NY, USA, 2018, pp. 72–83. URL: https://doi.org/10.1145/3264888.3264896. doi:10.1145/3264888.3264896.

[8] R. Taormina, S. Galelli, Deep-learning approach to the detection and localization of cyber-physical attacks on water distribution systems, Journal of Water Resources Planning and Management 144 (2018) 04018065. URL: https://ascelibrary.org/doi/abs/10.1061/(ASCE)WR.1943-5452.0000983. doi:10.1061/(ASCE)WR.1943-5452.0000983.

[9] W. Aoudi, M. Iturbe, M. Almgren, Truth will out: Departure-based process-level detection of

stealthy attacks on control systems, in: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS '18, Association for Computing Machinery, New York, NY, USA, 2018, pp. 817–831. URL: https://doi.org/10.1145/3243734.3243781. doi:10.1145/3243734.3243781.

[10] A. Erba, R. Taormina, S. Galelli, M. Pogliani, M. Carminati, S. Zanero, N. O. Tippenhauer, Constrained concealment attacks against reconstruction-based anomaly detectors in industrial control systems, in: Annual Computer Security Applications Conference, ACSAC '20, Association for Computing Machinery, New York, NY, USA, 2020, pp. 480–495. URL: https://doi.org/10.1145/3427228.3427660. doi:10.1145/3427228.3427660.

[11] T. Chen, X. Liu, B. Xia, W. Wang, Y. Lai, Unsupervised anomaly detection of industrial robots using sliding-window convolutional variational autoencoder, IEEE Access 8 (2020) 47072–47081. doi:10.1109/ACCESS.2020.2977892.

[12] A. Munawar, P. Vinayavekhin, G. De Magistris, Spatio-temporal anomaly detection for industrial robots through prediction in unsupervised feature space, in: 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), 2017, pp. 1017–1025. doi:10.1109/WACV.2017.118.

[13] V. Narayanan, R. B. Bobba, Learning based anomaly detection for industrial arm applications, in: Proceedings of the 2018 Workshop on Cyber-Physical Systems Security and PrivaCy, CPS-SPC '18, Association for Computing Machinery, New York, NY, USA, 2018, pp. 13–23. URL: https://doi.org/10.1145/3264888.3264894. doi:10.1145/3264888.3264894.

[14] D. Quarta, M. Pogliani, M. Polino, F. Maggi, A. M. Zanchettin, S. Zanero, An Experimental Security Analysis of an Industrial Robot Controller, in: Proceedings of the 38th IEEE Symposium on Security and Privacy, San Jose, CA, 2017.

[15] S. Longari, A. Cannizzo, M. Carminati, S. Zanero, A secure-by-design framework for automotive on-board network risk analysis, in: 2019 IEEE Vehicular Networking Conference, VNC 2019, Los Angeles, CA, USA, December 4-6, 2019, IEEE, 2019, pp. 1–8. URL: https://doi.org/10.1109/VNC48660.2019.9062783. doi:10.1109/VNC48660.2019.9062783.

[16] S. Zanero, When cyber got real: Challenges in securing cyber-physical systems, in: 2018 IEEE SENSORS, 2018, pp. 1–4. doi:10.1109/ICSENS.2018.8589798.

[17] M. Hussain, E. Foo, S. Suriadi, An improved industrial control system device logs processing method for process-based anomaly detection, in: 2019 International Conference on Frontiers of Information Technology (FIT), 2019, pp. 150–1505. doi:10.1109/FIT47737.2019.00037.

[18] M. Pogliani, F. Maggi, M. Balduzzi, D. Quarta, S. Zanero, Detecting Insecure Code Patterns in Industrial Robot Programs, in: Proceedings of the 15th ACM Asia Conference on Computer and Communications Security (ASIA CCS '20), Taipei, TW, 2020.

[19] J. Zhang, S. Gan, X. Liu, P. Zhu, Intrusion detection in scada systems by traffic periodicity and telemetry analysis, in: 2016 IEEE Symposium on Computers and Communication (ISCC), 2016, pp. 318–325. doi:10.1109/ISCC.2016.7543760.

[20] D. Hadžiosmanović, R. Sommer, E. Zambon, P. H. Hartel, Through the eye of the plc: Semantic security monitoring for industrial processes, in: Proceedings of the 30th Annual Computer Security Applications Conference, ACSAC '14, Association for Computing Machinery, New York, NY, USA, 2014, pp. 126–135. URL: https://doi.org/10.1145/2664243.2664277. doi:10.1145/2664243.2664277.

[21] H. Yoo, S. Kalle, J. Smith, I. Ahmed, Overshadow plc to detect remote control-logic injection attacks, in: R. Perdisci, C. Maurice, G. Giacinto, M. Almgren (Eds.), Detection of Intrusions and Malware, and Vulnerability Assessment, Springer International Publishing, Cham, 2019, pp. 109–132.

[22] L. Medsker, L. C. Jain, Recurrent neural networks: design and applications, CRC press, 1999.

[23] K. O'Shea, R. Nash, An introduction to convolutional neural networks, 2015. arXiv:1511.08458.

[24] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (1997) 1735–80. doi:10.1162/neco.1997.9.8.1735.

[25] C. Feng, T. Li, D. Chana, Multi-level anomaly detection in industrial control systems via package signatures and lstm networks, 2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN) (2017) 261–272.

[26] D. Bank, N. Koenigstein, R. Giryes, Autoencoders, 2021. arXiv:2003.05991.

[27] S. Longari, D. H. N. Valcarcel, M. Zago, M. Carminati, S. Zanero, Cannolo: An anomaly detection system based on LSTM autoencoders for controller area network, IEEE Trans. Netw. Serv. Manag. 18 (2021) 1913–1924. URL: https://doi.org/10.1109/TNSM.2020.3038991. doi:10.1109/TNSM.2020.3038991.

[28] S. Longari, A. Nichelini, C. A. Pozzoli, M. Carminati, S. Zanero, Candito: Improving payload-based detection of attacks on controller area networks, CoRR abs/2208.06628 (2022). URL: https://doi.org/10.48550/arXiv.2208.06628. doi:10.48550/arXiv.2208.06628. arXiv:2208.06628.

[29] S. Zanero, When cyber got real: Challenges in securing cyber-physical systems, in: 2018 IEEE SENSORS, New Delhi, India, October 28-31, 2018, IEEE, 2018, pp. 1–4. URL: https://doi.org/10.1109/ICSENS.2018.8589798. doi:10.1109/ICSENS.2018.8589798.

[30] A. Humayed, J. Lin, F. Li, B. Luo, Cyber-physical systems security-a survey, IEEE Internet of Things Journal 4 (2017) 1802–1831. doi:10.1109/JIOT.2017.2703172.

[31] F. Maggi, D. Quarta, M. Pogliani, M. Polino, A. M. Zanchettin, , S. Zanero, Mobile Systems IV, Technical Report, Trend Micro TrendLabs, 2017.

[32] S. Belikovetsky, M. Yampolskiy, J. Toh, Y. Elovici, dr0wned - cyber-physical attack with additive manufacturing, 2016. arXiv:1609.00133.

[33] R. Shumway, D. Stoffer, Time series: a data analysis approach using R, CRC Press, 2019.

[34] S. Muzaffar, A. Afshari, Short-term load forecasts using lstm networks, Energy Procedia 158 (2019) 2922–2927. URL: https://www.sciencedirect.com/science/article/pii/S1876610219310008. doi:https://doi.org/10.1016/j.egypro.2019.01.952, innovative Solutions for Energy Transitions.

[35] T. Iordanova, Introduction to stationary and non-stationary processes, Retrieved September 19 (2009) 2013.

[36] I. Bongiorni, Tensorflow 2.0 notebooks, 04.03 rnnmmultivariate regression, ???? URL: https://github.com/IvanBongiorni/TensorFlow2.0_Notebooks/blob/master/TensorFlow2__04.03_RNN_multivariate_regression.ipynb.

[37] S. Malakar, S. Goswami, B. Ganguli, A. Chakrabarti, S. S. Roy, K. Boopathi, A. G. Rangaraj, Designing a long short-term network for short-term forecasting of global horizontal irradiance, SN Applied Sciences 3 (2021) 477. URL: https://doi.org/10.1007/s42452-021-04421-x. doi:10.1007/s42452-021-04421-x.