IM-DISCO: Invariant Mining for Detecting IntrusionS in Critical Operations

Guilherme Saraiva[1], Filipe Apolinário[1][0000-0002-2067-3232], and Miguel L. Pardal[2][0000-0003-2872-7300]

INOV[1], INESC-ID Instituto Superior Técnico, Universidade de Lisboa[2]
{guilherme.a.saraiva,filipe.apolinario,miguel.pardal}@tecnico.ulisboa.pt

Abstract. In today's interconnected world, robust cybersecurity measures are crucial, especially for Cyber-Physical Systems. While anomaly-based Intrusion Detection Systems can identify abnormal behaviors, interpreting the resulting alarms is challenging. An alternative approach utilizes invariant rules to describe system operations, providing clearer explanations for abnormal behaviors. In this context, invariant rules are conditions that must hold true for a system's different operational modes. However, defining these rules is time-consuming and costly. This paper presents IM-DISCO, a tool that analyzes operational data to propose inference rules characterizing different modes of system operation. Deviations from these rules indicate anomalies, enabling continuous monitoring with incident detection and response. In our evaluation, focusing on rail transportation, we achieved 99.29% accuracy in detecting and characterizing operational modes using real-world train data. Additionally, we achieved 99.86% accuracy in identifying anomalies during simulated attacks. Notably, our results demonstrate an average detection time of 0.026 ms, enabling swift incident response to prevent catastrophic events.

Keywords: Anomaly detection · Cyber-physical systems · Invariants · Intrusion detection systems

1 Introduction

Cyber-Physical Systems (CPS) monitor and control physical processes using embedded computers with sensor networks [20]. They are commonly found in Critical Infrastructures (CI) such as transportation networks and power grids. However, the growing complexity and interconnectivity of CPS make them especially vulnerable to cyber-physical attacks [16]. For instance, on October 29, 2022, a ransomware attack on a railway in Denmark caused a disruption on the train services. The attack was not directly targeted at the train systems but at a third-party IT service provider. The attack led to the shutdown of servers, causing the mobile application used by train drivers to stop working. As a result, the trains came to a standstill for several hours.

Intrusion Detection Systems (IDS) for CPS have been developed throughout the years to prevent or mitigate cyberattacks [17]. An IDS passively collects and analyzes different data sources, such as network traffic and security logs. In particular, anomaly-based IDS establish a model of the system's normal behavior, and deviations from this model raise alarms [15]. Most analysis techniques are based on data mining and machine learning [25]. This type of IDS can detect novel attacks, but it is difficult to interpret the cause of its alarms, and occurrences of false positives are frequent.

An alternative approach is based on invariant rules [23]. An *invariant* is a physical condition that must be sustained in a certain operational mode, characterized by a unique combination of sensor readings and actuator states. In this approach, an "anomaly" corresponds to any physical process value that violates the rules. Invariant rules can be manually defined by CPS experts. However, manually defining invariants can be costly, time-consuming, and incomplete for complex CPS.

In this article, we present IM-DISCO (Invariant Mining for Detecting IntrusionS in Critical Operations), a tool that generates invariant rules for inferring operational modes and anomalies on CPS. This tool utilizes CPS log data as input to perform its analyses. Contrary to prior invariant rule mining approaches [27, 11, 10], IM-DISCO uses expert domain knowledge to define CPS operational modes and infers the invariants that describe the sensors and actuator conditions that best describe those modes. As a result, the invariant rules help to explain, based on observation, the current operational mode of a CPS. They can be used to monitor, in real-time, sensors and actuators, detect changes in CPS states, and identify anomalies. With IM-DISCO, experts are able to understand alerts and easily attest, based on the mode reported by IM-DISCO, the veracity of the alarm. Thus, IM-DISCO contributes to:

1. A novel approach for providing invariant rules for inferring operational modes within CPS;
2. An IDS method that allows the detection of anomalies in a timely way and that can be verified by human experts, as needed.

We performed an experimental evaluation with two real-world train datasets. The results of this evaluation showed that IM-DISCO can accurately detect operational modes. In addition, we used a simulated train ride containing attacks, during which IM-DISCO was able to identify them. We also showed that our solution maintained high accuracy even with low training data, demonstrating its agility in generating rules with limited input. Finally, we evaluated the performance of the different phases of our solution, and the results show fast rule generation and verification. This allows expert assessment for validation and implementation of suitable preventive actions.

2 Related Work

Detecting anomalies on CPSs is a topic of research that has been studied for many years in different industrial ecosystems [25]. We classify three different types of techniques used in Anomaly-based IDS: *Fingerprinting* [3, 4, 6], *Artificial*

Neural Networks (NN) [12, 13], and *Invariants* [27, 11, 10, 23, 5, 1, 28, 26]. On the one hand, the initial two methodologies represent the system's behavior using black-boxes, which allows to accurately detect anomalies in high-dimensional datasets but are difficult to be interpreted by the system expert. On the other hand, invariant-rule-based anomaly detectors employ explicit rules that define the expected behavior of a system. These rules can be understood and interpreted by humans, as they represent specific conditions or thresholds that should be met. When an anomaly is detected, the violated rules can be used to explain the cause of the alarm. For this reason, we focus on invariant-based works.

Invariants are properties that must be held to maintain the normal behavior of the system. For instance, an invariant rule of an actuator A, a sensor S, and an operational mode M for a railway CPS can be:

$$A.doors{=}ON \land S.velocity{=}0 \implies M{=}\text{``}on\_station\text{''}$$

In this rule, each smallest subequation is a *predicate*. This rule states that if the doors of a train are opened and the velocity is equal to zero, then the train must be in the operational mode *on_station*.

Invariant rules can be derived from data logs and the system's design. Data-driven approaches rely on data mining and machine learning techniques to discover invariants. Typically, these approaches are divided into two steps: predicate generation and invariant rule mining. Some methods to generate predicates use the distribution of sensor value updates and estimate the parameters of this distribution [11]. Another set of predicates can be derived from sensor values that trigger changes in actuator states [11]. The manual definition of events with their associated data variables [5] and simple threshold calculations [27] represent other possible predicates. After obtaining sets of predicates, the goal is to find associations between them to generate invariants.

Association Rule Mining (ARM) is the technique most commonly used for this task [11, 23, 5, 26]. ARM is a rule-based method to uncover relationships between multiple state variables [19]. These relationships represent the final invariant rules. General dynamic analysis-based tools can also be used to mine associations [10]. From the system's design, it is possible to derive invariants from Process and Instrumentation Diagrams (P&ID), State Condition Graphs (SCGs) [26], and Automata [1] or use axiomatic design theory to decompose functional requirements into invariants [28]. Table 1 depicts the various techniques found in the literature to generate invariant rules, and our proposal, IM-DISCO.

System design approaches have the advantage of deriving invariants directly from the architecture of the system. However, it can have extensive documentation, which makes the process of deriving invariants impractical, and sometimes the documentation does not follow the evolution of the system. Due to these limitations, we opt to follow a data-driven approach as an alternative.

To the best of our knowledge, IM-DISCO is the first data-driven tool that generates predicates using out-of-bounds approaches, allowing the mining of invariants using Association Rule Mining to create understandable invariant rules that characterize operational modes that can be attested by CPS experts, optimizing the detection of anomalies, and improving CI protection.

Table 1. Key attributes of invariant-based anomaly detectors.

Related Work	Source	Predicate Generation	Invariant Mining	Knowledge-based
[27]	Data Logs	Thresholds	-	No
[11]	Data Logs	Distribution, Events	Association Rules	No
[10]	Data Logs	-	Dynamic Analysis	No
[23]	Data Logs	Manual Craft	Association Rules	Yes
[5]	Data Logs	Manual Craft	Association Rules	Yes
[1]	System's Design	Manual Craft	Hybrid Automata	Yes
[28]	System's Design	Manual Craft	Axiomatic Design	Yes
[26]	Data Logs, System's Design	Manual Craft	P&ID/SCGs, Association Rules	Yes
IM-DISCO	**Data Logs**	**Thresholds**	**Association Rules**	**Yes**

3 IM-DISCO

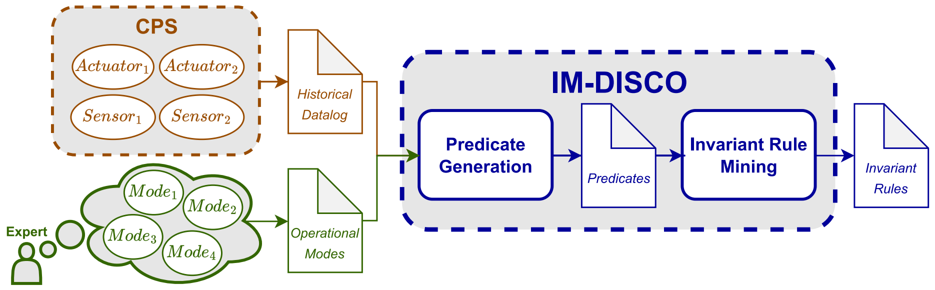

Fig. 1. Example of IM-DISCO generating invariant rules of a CPS with two actuators, two sensors, and four operational modes. IM-DISCO employs historical data logs encompassing CPS sensor and actuator values in distinct operational modes to create invariant rules.

As illustrated in Fig. 1, the goal of IM-DISCO is to provide *Invariant Rules* composed of *Predicates* that represent the physical conditions of system operation. These predicates will be extracted through the use of multiple techniques.

IM-DISCO uses a historical data log to generate invariant rules. This data log registers CPS sensors and actuator values while the system was operating in different modes. To generate invariant rules, IM-DISCO first calculates predicates that express the normal functioning of the CPS. These predicates are represented by statistical bounds of actuator and sensor values for each operational mode. Once the predicates are obtained, the tool finds associations between them to mine invariant rules. The final invariant rules can later be used by a detector to infer, based on sensor and actuator data, the current operational mode of the system and detect abnormal variances in the physical values.

3.1 Formalization of Concepts

A CPS comprises a group of $\mathbb{S}$ sensors that measure, at each timestep, some physical quantities of the environment (velocity, acceleration, position, etc.).

Specifically, each sensor S_i performs $measure(S_i, t) = x$ where $x \in \mathbb{Q}$ and t is the respective timestep. Additionally, there is a group of $\mathbb{A}$ actuators that control physical processes, such as moving the wheels of a train or adjusting the flow rate of a fluid. The sensors in a CPS are responsible for collecting data from the physical environment and transmitting it to the control system. The control system processes this data and sends commands to the actuators to adjust the physical processes in response to changes in the environment. Specifically, each actuator A_k can change its state $state(A_k, t) = y$ at a certain timestep t according to the physical values measured by the sensors and that can cause a change in the system mode. At each timestep, a CPS can be represented as vector $\overrightarrow{C_m}$ of $|S| \times |A|$ dimensions, where each dimension is the value of a physical sensor/actuator. The CPS can operate in different modes m that belong to a set $\mathbb{M}$. These modes can be observed by an expert, and correspond to a given set of CPS vector observations $\mathbb{C}_{\mathbb{M}}$. The main objective of IM-DISCO is thus, provide invariant rules that infers the operation mode based on a CPS vector, i.e., $imdisco(\overrightarrow{C}) = m$.

3.2 Railway Example

For instance, consider a train that uses a CPS to control its movement. The CPS comprises a group of sensors that measure the velocity $measure(S_{velocity}, t)$ and acceleration $measure(S_{acceleration}, t)$ of the train at each timestep t. The CPS also has a group of actuators, including the brakes and throttle, responsible for applying power to the train, making it go slower or faster. The system can operate in different modes: $m_1 = arriving\_station$, $m_2 = leaving\_station$, $m_3 = on\_station$, or $m_4 = riding$. To maintain the proper operation of the train, each mode has to respect a certain limit of velocity and acceleration, which are controlled by specific actuator state changes, such as $state(A_{brakes}, t) = ON$ to reduce the speed of the train when arriving at the station, $state(A_{throttle}, t) = ON$, to increase the speed of the train when leaving the station, or $state(A_{doors}, t) = OFF$ when the train is moving.

It is possible to establish relationships between the boundaries of the physical values and the corresponding states of the actuators for each operational mode of the CPS. This allows us to properly model the correct behavior of the CPS in the form of physical conditions. By incorporating these conditions into a detector, we can easily identify the current operational mode of the system and give context in the event of a detected anomaly.

3.3 Predicate Generation

The Predicates generated in IM-DISCO correspond to different types of boundaries for the sensor measurements and actuator states based on observation of previous runs of the CPS. The discovery of the CPS predicates will not only prevent malfunctions and attacks on the physical values, but will also constitute the sub-conditions of invariant rules. This approach is based on the fact that many values fall within safety limits, while others are binary in nature (i.e., ON/OFF switches).

Specifically, for each sensor S_i, actuator A_k, and mode m, we define a set of Predicates $\mathbb{P}_m$. To obtain these predicates, we applied techniques used by SIMPLE [27] in which different thresholds are defined for each sensor/actuator based on their characteristics. The difference between both systems is that SIMPLE models rules based on the individual actuator and sensor thresholds, while IM-DISCO uses them to uncover the CPS predicates and mine invariant rules. The approaches used by SIMPLE include: MinMax; Gradient; and SteadyTime.

MinMax extracts the minimum (Min) and the maximum (Max) values observed by each sensor. This approach is grounded on the premise that the physical values that constitute these systems are inherently constrained by well-defined limits. Consequently, measurements outside these limits have the potential to disrupt certain operations of the system. The resulting predicates are defined as $minmax : min(V_m) < measure(S_i, t) < max(V_m)$, in which V_m represents all the physical values captured in a certain mode m.

Gradient establishes the limits of each sensor's observed slope. Unlike the MinMax approach, the Gradient method is capable of detecting subtle attacks that aim to abruptly modify the physical values within operational limits. Such arbitrary changes to physical variables have the potential to cause critical disturbances to the system. The resulting predicates are defined as $gradient : min(G_m) < gradient(measure(S_i, t)) < max(G_m)$, in which G_m represents all the gradients of the physical values captured in a certain mode m, and $gradient(measure(S_i, t)) = measure(S_i, t) - measure(S_i, t-1)$.

Steadytime detects any instances wherein an actuator's value remains static for a duration shorter or longer than what has been previously observed. This approach is based on the observation that each actuator state should endure for a specific interval of time. The resulting predicates are defined as $steadytime : min(T_m) < (state(A_k, t) = y) < max(T_m)$, in which, T_m represents the totals of consecutive timesteps that the actuator A_k was in the state y in a certain mode m.

To complement these predicates, we can also define predicates of the form $state(A_k, t) = y$ for each timestep. This allows us to represent all the states that the actuator k can have during mode m.

With these techniques, we are able to extract predicates based on the physical values of the sensors and actuators. Associating these predicates allows the formation of invariant rules that characterize a certain mode. For instance, using the railway station example introduced earlier in Section 3.2, we can generate multiple predicates $\mathbb{P}_m$ for its sensors and actuators for when the train is leaving the station, i.e., $m = leaving\_station$. For example:

- $minmax : 0.0 < measure(S_{velocity}, t) < 29.9$. This predicate establishes the safety limits of the velocity of the train;
- $gradient : 0.0 < gradient(measure(S_{acceleration}, t)) < 2.42$. This predicate states that the acceleration slope of the train must be between 0.0 and 2.42;

- $steadytime : 90 < (state(A_{throttle}, t) = ON) < 142$. This predicate states that the throttle must be "ON" between 90 and 142 timesteps and only during this period;
- $state(A_{brake}, t) = OFF$. This predicate represents that the brake was "OFF" at a certain timestep.

3.4 Invariant Rule Mining

From the generated predicates, it is now possible to derive the invariant rules that characterize the operational modes of the CPS. The main goal is to identify the relationships that express the correct functioning of the CPS for each mode, which represent the rules that are critical for maintaining the infrastructure's normal operation. Specifically, from the sets of Predicates $\mathbb{P}_m$, we want to mine a set of Invariant Rules $\mathbb{I}_m$ in which each rule is of the form $A \implies C$, where A represents a subset of Predicates and C represents an operational mode m.

Invariant Rule Mining (IRM) is achieved through a data mining technique known as association rule mining, which discovers patterns and associations between variables in large datasets. ARM is typically divided into two phases: *Frequent Itemsets Extraction* (FIE) and *Association Rules Generation* (ARG).

Identifying *frequent itemsets* involves finding sets of items that commonly appear together – co-occur – in a dataset [2]. The most widely used algorithms to identify these itemsets are Apriori [22] and FP-Growth [18]. The items correspond to the predicates contained in $\mathbb{P}_m$.

The first step in identifying frequent itemsets is to define a *minimum support threshold*, which specifies the minimum proportion of entries that an itemset must appear in to be considered frequent [2]. After that, the algorithm scans the dataset to identify all the individual items and their support values. It then generates candidate itemsets, which are sets of items that have not yet been determined to be frequent. The algorithm then scans the dataset again to count the support of each candidate itemset. A candidate itemset is added to the list of frequent itemsets if its support is above the minimum support threshold. This process is repeated iteratively until no new frequent itemsets are found.

Given a dataset D, where each dataset entry d is a set of predicates $d \in \mathbb{P}_m$, and a minimum support threshold *minsup*, find all frequent itemsets F such that the support of I in D, denoted $supp(I)$, is greater than or equal to *minsup* ca be expressed as: $F = \{I \subseteq \mathbb{P}_m | supp(I) \geq minsup\}$ where $\mathbb{P}_m$ is the set of all predicates in the dataset D, and $supp(I)$ is the proportion of entries in D that contain all the predicates in I, or $supp(I) = \frac{|\{d \in D | I \subseteq d\}|}{|D|}$.

After identifying the frequent itemsets, the next step is to generate *association rules*. Association rules are statements that describe the relationship between different items in the dataset. The process of generating association rules involves setting a minimum confidence threshold $minconf$. The confidence of an association rule measures how often the rule is valid for the entries in the dataset [2]. The minimum confidence threshold specifies the minimum level of confidence that an association rule must have to be considered significant.

Once the $minconf$ is set, the algorithm generates all possible association rules from the frequent itemsets. Each association rule consists of an antecedent and a consequent, and the support and confidence of each rule are calculated. The algorithm then filters out the association rules that do not meet $minconf$.

The resulting set of association rules can be sorted by their support and confidence to identify the most significant rules. Other metrics can also be used to evaluate association rules, such as *lift*, which measures how much the presence of the antecedent increases the likelihood of the consequent, or *conviction*, which measures how much the absence of the antecedent decreases the likelihood of the consequent [8].

Hence, an association rule is an implication expression of the form $A \implies C$, where A is a subset of $\mathbb{P}_m$ and C is the respective operational mode, if and only if $conf(A \implies C) \geq minconf$, where $conf(A \implies C)$ is the respective confidence and can be calculated as:

$$conf(A \implies C) = \frac{supp(A \cup C)}{supp(A)}$$

In the end, we are able to derive invariant rules from the obtained predicates.

Considering again the railway scenario presented in Section 3.2. An example of an invariant rule for $m = leaving\_station$ would be:

$$minmax{:}0.0<v_t<29.9 \wedge steadytime{:}90<(throttle_t{=}ON)<142 \implies leaving\_station$$

By providing invariant rules, such as this one, our solution allows for straightforward interpretation and understanding of the system's behavior. The rule provides clear criteria for determining whether the system is operating correctly during the $leaving\_station$ mode in which the velocity of the train (v_t), given by $measure(S_{velocity}, t)$, can not be lower than 0.0 or higher than 29.9 units and the state of the throttle ($throttle_t$), given by $state(A_{throttle}, t)$, can not be "ON" less than 90 or more than 142 timesteps. Any deviation from these conditions would indicate the presence of a problem. By continuously monitoring the actuator states and comparing them to the defined rules, our solution can identify deviations or abnormal behaviors that may compromise the safety or efficiency of the system.

3.5 Summary

IM-DISCO is a tool that generates invariant rules composed of predicates to characterize the operational modes and detect anomalies in CPS. It utilizes historical sensor and actuator data to calculate predicates representing the normal functioning of the CPS. These predicates define the statistical bounds of physical values and actuator states for each operational mode.

IM-DISCO employs association rule mining to generate invariant rules. It extracts common co-occurring predicates and then finds relationships between them. The support and confidence of the generated rules are calculated, and the final rules are selected based on a minimum confidence threshold.

Overall, the use of easily understandable invariant rules facilitates effective preventive measures, enabling operators or experts to promptly respond to anomalies, mitigate risks, and ensure the smooth operation of the system.

4 Implementation

The IM-DISCO tool is composed of the phases presented in Fig. 1, namely: *Predicate Generation* and *Invariant Rule Mining*. We used Python programming language to develop our tool.

The *Predicate Generation* phase is responsible for generating predicates based on the input CPS data log, which includes telemetry from sensors and actuator states. To this data, the expert adds the observed operational modes. For each sensor and actuator, we calculate the appropriate thresholding approaches as described in Section 3.3. These thresholds determine the ranges of values that define different modes. Once the predicates are generated, they are combined to form itemsets, which are then used as input for the *Invariant Rule Mining* phase.

To associate the predicates and generate invariant rules, we used the *MLxtend* library [24], which offers a range of machine learning and data mining tools. We now describe the two subphases of the *Invariant Rule Mining* step: *Frequent Itemset Extraction* and *Association Rule Generation*.

The objective of FIE is to identify itemsets, which correspond to the generated predicates, that occur frequently in the dataset. First, we transform the dataset containing the predicates into a binary matrix representation using the MLxtend function $fit\_transform()$. This binary matrix indicates the position of each predicate in the dataset. We then apply the frequent itemset extraction algorithm. In our case, we utilized the FP-Growth implementation provided by MLxtend, which is reported to be approximately five times faster than the Apriori algorithm[1]. To capture the maximum number of frequent predicates, we set a *minsup* of 0.05, the minimum allowed value by the FP-Growth algorithm. Having a larger set of predicates enables us to generate a greater number of associations, resulting in more robust rules.

ARG aims to derive associations between the frequent predicates and generate invariant rules. We employed the $association\_rules()$ function from MLxtend, which takes the frequent predicates as input and returns the association rules that satisfy a certain confidence level. In our case, we set $minconf$ to 1, ensuring that only rules with perfect confidence are generated. This choice minimizes the risk of false positives and ensures that the derived rules are highly significant.

5 Evaluation

We performed several experiments using different train travel scenarios to evaluate the performance and efficacy of IM-DISCO in critical infrastructure. The evaluation aimed to answer the following research questions:

[1] https://rasbt.github.io/mlxtend/user_guide/frequent_patterns/fpgrowth/

RQ. 1) Can IM-DISCO infer the correct operational mode?
RQ. 2) Can IM-DISCO be applied for anomaly detection?
RQ. 3) How much time does IM-DISCO take to generate and verify rules?

By addressing these questions, we can obtain an understanding of the tool's accuracy, sensitivity to anomalies, and performance of rule generation and verification. The evaluation will provide insights into the practicality of the IM-DISCO tool and its ability to support anomaly detection in cyber-physical systems. The evaluation was performed on a computer with Intel Core i5-7300HQ CPU 2.50GHz processor, 16GB of RAM, and running Windows 10.

5.1 Data Collection and Experiment Setup

The datasets used to assess our solution correspond to two real-world suburban train rides on the same railway. The data, such as the geographic points and velocity, were collected at 1-second intervals using the *Strava* application running on a smartphone. This app is used to track physical exercise using Global Positioning System (GPS) data.

The first ride, or departure ride (R_d), was from Lisboa - Entrecampos to Portela de Sintra had a duration of 35 minutes and 22 seconds, and comprehended 12 stops until it arrived at the final one, which gives a total of 2122 datapoints. The second ride, or return ride (R_r), represents the return trip, had a duration of 34 minutes and 28 seconds, and comprehended the same number of stops totaling 2068 datapoints. The mean time between stops is approximately 3 minutes and the mean velocity of the train was 40.7 kilometers per hour (km/h). From this data, we have derived other metrics, such as acceleration, the state of the brakes, and throttle (on and off). We have also annotated the state of the doors (opened or closed), and more importantly the operational mode of the train (*on_station*, *leaving_station*, *riding*, and *arriving_station*). From this scenario, we have modeled the train as a cyber-physical system with a set of actuators $\mathbb{A}$ and sensors $\mathbb{S}$ (recall Section 3.2). Actuators are the brakes, throttle, and doors. The sensors are the speedometer and accelerometer.

However, the collected datasets have limitations in representing the diverse conditions of real-world train operations; the sample size may not capture the full range of train behaviors and potential anomalies; and abnormal values were captured due to low GPS signals in tunnels. These values were used to assess IM-DISCO's detection capabilities in this specific scenario.

5.2 Evaluation Metrics

In the evaluation of our solution, we employed a set of commonly used metrics to assess its efficacy, namely: false positives (FP), false negatives (FN), true positives (TP), true negatives (TN), recall, precision, F1-measure, and accuracy. An FP is the number of instances that do not belong to a specific class but are incorrectly classified as belonging to that class. An FN is the number of instances that belong to a specific class but are incorrectly classified as not belonging to that class. A TP is the number of instances that are correctly classified as a specific class. A TN is the number of instances that are correctly classified as not belonging to a specific class. Recall measures the proportion of correctly classified instances out of the total instances that actually belong to a specific class and is computed as $\frac{TP}{(TP+FN)}$. Precision is calculated as $\frac{TP}{(TP+FP)}$ and measures the proportion of correctly classified instances out of the total instances predicted as belonging to a specific class. The F1-measure, also known as the F1-score, is the harmonic mean of precision and recall that determines the overall performance. Its formula corresponds to $\frac{2*Precision*Recall}{(Precision+Recall)}$. Accuracy measures the overall correctness of the classification model, and it is calculated as $\frac{(TP+TN)}{(TP+TN+FP+FN)}$. We consider this a multiclass classification problem [14] where the goal is to classify instances into multiple categories or classes. In this context, each class represents a specific operational mode or anomaly.

5.3 Operational Mode Inference (RQ. 1)

To evaluate the accuracy of our rules in correctly identifying the operational modes of the system, we conducted a training and testing process. Following the same approach of previous works [27, 11], we have trained IM-DISCO with 80% of the dataset, and tested with 20%. Specifically, we used the data collected from the first 10 stops to generate invariant rules that characterize each mode. Then, we trained a detector with the resulting rules and tested it using the remaining data. Table 2 demonstrates the results of this experiment.

Table 2. Results of using invariant rules for detecting R_d and R_r operational modes.

	Precision		**Recall**		**F1-score**		**Accuracy**	
	R_d	R_r	R_d	R_r	R_d	R_r	R_d	R_r
arriving_station	100%	100%	98.53%	97.89%	99.26%	98.94%		
leaving_station	100%	43.75%	100%	100%	100%	60.87%		
on_station	100%	100%	100%	100%	100%	100%		
riding	99.60%	100%	99.60%	93.21%	99.60%	96.48%		
IM-DISCO	99.90%	85.94%	99.53%	98.22%	99.71%	89.07%	**99.29%**	**95.17%**

These results demonstrate that our solution can accurately detect the mode of the train using only two sensors and three actuators. The achieved high accuracy indicates the effectiveness of our approach. In addition, our solution successfully detected two out of the three abnormal datapoints caused by the low GPS signal. The third abnormal datapoint went undetected as the train maintained a constant movement during the interference.

However, we can observe by the precision metric in R_r that the *leaving_station* operational mode was wrongly inferred by the detector. This happened due to the overlapping of predicates in different invariant rules. Specifically, the *Min-Max* predicate for velocity and acceleration overlapped between the *riding* and *leaving_station* states. For example, the predicates for *riding* were $29.452 \leq$ velocity ≤ 96.98 and $-13.676 \leq$ acceleration ≤ 21.078, while the predicates for *leaving_station* were $0.007 \leq$ velocity ≤ 29.974 and $-6.469 \leq$ acceleration ≤ 13.018. As a result, when the predicates of the two sensor values overlap, and the predicates for all actuators are the same, the detector chooses the first mode that meets its invariant rules. Having a larger set of sensors and actuators would significantly reduce the likelihood of this occurrence.

Moreover, to evaluate how well our solution maintains its high accuracy in detecting OMs, we use different training/testing split datapoints. Fig. 2 displays a graph of the accuracy values for both rides for different training sizes. Our approach consistently achieves high accuracy, exceeding 95%, even with small training sizes. This shows that our solution effectively generates accurate rules despite limited data. The ability to achieve high accuracy with low training data is significant, as it allows our approach to be applied in challenging scenarios where obtaining large amounts of data is impractical. This is valuable in real-world situations with limited data collection due to constraints.

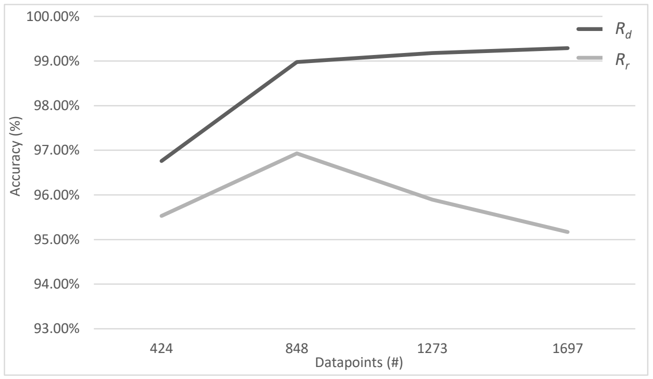

Fig. 2. Accuracy of IM-DISCO (y-axis) across different training sizes (x-axis).

5.4 Anomaly Detection (RQ. 2)

To evaluate the capability to detect anomalies, we developed a train ride simulator, since we could not implement attacks in the real-world datasets used before. The simulated train ride uses the same operational modes as previously mentioned, comprising a throttle, a brake, and the doors that open when the train arrives at a station. Sensors were utilized to capture the velocity, acceleration, throttle force, and brake force at each second.

The resulting training dataset consisted of a 48-minute ride with 13 stops, providing substantial and representative data for training IM-DISCO. The testing dataset, obtained under the same conditions, encompassed a 12-minute ride with 3 stops. The stops were unevenly spaced to accurately replicate a real train ride. Additionally, an attack was introduced in the testing dataset, involving tampering with the brakes. The locomotive driver was unable to use the regular brakes and instead had to rely on an emergency brake that exerted a force 2.5 times stronger than the regular brake. This attack was enacted twice in the testing dataset. From this experiment, IM-DISCO generated a total of 76 predicates and 4 invariant rules, one for each operational mode.

Table 3. Results of using invariant rules for anomaly detection in a simulated ride.

	Precision	**Recall**	**F1-score**	**Accuracy**
anomaly	95.24%	100%	97.56%	
arriving_station	100%	99.56%	99.78%	
leaving_station	100%	100%	100%	
on_station	100%	100%	100%	
riding	100%	100%	100%	
IM-DISCO	99.05%	99.91%	99.47%	**99.86%**

Table 3 shows the obtained evaluation metrics. The experiment's results demonstrated the capability of our solution to accurately identify both anomalies and the other operational modes, achieving 99.86% accuracy in detection. This high accuracy was mainly due to the efficiency of the generated predicates and invariant rules, and also due to the good representativity of the training data. For instance, our tool detected an acceleration outside the values captured in training for each operational mode and considered it an anomaly. However, we got one FN inferring the mode *arriving_station* due to a velocity value that fell outside the range of values captured during training. Nonetheless, these results showcase the effectiveness of our approach in identifying abnormal situations.

5.5 Invariant Rules Verification and Validation (RQ. 3)

To evaluate the performance of rule generation and verification, we benchmarked the time taken to generate and validate invariant rules. We utilized different training/testing split datapoints and ran our tool 30 times to avoid skewed benchmarks. The duration of three key processes was recorded: predicate generation (Section 3.3), invariant rule generation (Section 3.4), and operational mode detection. By running our tool multiple times and calculating the means of the duration for each process, we assess the average speed our solution performs. Fig. 3 depicts the results of these experiments.

The results indicate that there is a direct relationship between the size of the training data and the time required for predicate and invariant rules generation. As expected, larger training datasets require more computational resources, leading to increased processing times in both processes. However, it is noteworthy that the rule generation process exhibits a lower rate of increase in processing time compared to the predicate generation process. This can be attributed to the fact that the number of sensors and actuators, and the number of predicates/itemsets remains constant throughout the experiments (2 sensors, 3 actuators, and 52 predicates). Despite this, we can observe that both processes

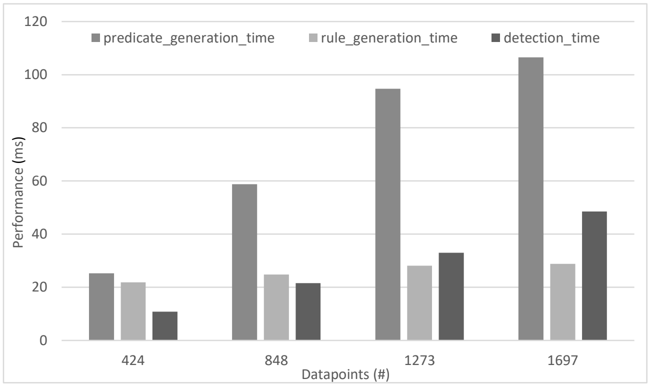

Fig. 3. Performance of our solution across different dataset sizes (x-axis, number of datapoints).The y-axis is the average performance in milliseconds (ms) of 30 executions.

are relatively fast, which will be advantageous in deploying the solution in a real-world scenario. Moreover, the low detection times, averaging around 0.026 ms per datapoint, further enhance the responsiveness of our solution, allowing prompt identification and preventive actions.

6 Conclusion

In this work, we described IM-DISCO, a solution for inferring the operational modes and detecting anomalies in cyber-physical systems using historical data. Our approach generates invariant rules based on different types of predicates that represent the physical conditions of the CPS. Through the evaluation of our solution, we have demonstrated its effectiveness, agility, and performance. The generated invariant rules exhibited an accuracy above 95% in detecting the correct operational modes with low training sizes. This indicates that our solution can reliably characterize and identify the different CPS operational modes even with limited training data available. In addition, we verified that the invariant rules are able to detect anomalies with 99.86% accuracy. We also assessed the performance of our solution in terms of rule generation and verification. The results showed that our solution is able to generate invariant rules in less than 150ms for 1697 datapoints, which can be verified in around 0.026 ms.

From the obtained results, we can conclude that IM-DISCO provides an effective approach for detecting and characterizing the operational modes or abnormal behavior of a cyber-physical system. It demonstrates high accuracy and agility in rule generation and fast detection of operational modes. Our solution facilitates timely anomaly detection and aids in the prevention of potential disruptions or catastrophes. IM-DISCO could be applied in infrastructures as an additive security tool: correlation with network detection tools [7], can reveal a network intrusion that originated the CPS anomaly; correlation with process monitoring tools [21], may identify anomalous processes; integration with impact assessment [9] may enable the investigation of CPS anomaly cascading effects.

Acknowledgement

Work supported by the European Commission through contract 101021797 [2].

References

1. Adepu, S., Mathur, A.: From design to invariants: Detecting attacks on cyber physical systems. In: 2017 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C). pp. 533–540. IEEE (2017)
2. Agrawal, R., Imieliński, T., Swami, A.: Mining association rules between sets of items in large databases. In: Proceedings of the 1993 ACM SIGMOD international conference on Management of data. pp. 207–216 (1993)
3. Ahmed, C.M., Ochoa, M., Zhou, J., Mathur, A.P., Qadeer, R., Murguia, C., Ruths, J.: *NoisePrint*: Attack Detection Using Sensor and Process Noise Fingerprint in Cyber Physical Systems. In: Proceedings of the 2018 on Asia Conference on Computer and Communications Security. pp. 483–497. ACM, Incheon ROK (2018)
4. Ahmed, C.M., Zhou, J., Mathur, A.P.: Noise Matters: Using Sensor and Process Noise Fingerprint to Detect Stealthy Cyber Attacks and Authenticate sensors in CPS. In: Proceedings of the 34th Annual Computer Security Applications Conference. pp. 566–581. ACM, San Juan PR USA (Dec 2018)
5. Aliabadi, M.R., Kamath, A.A., Gascon-Samson, J., Pattabiraman, K.: Artinali: dynamic invariant detection for cyber-physical system security. In: Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering (2017)
6. Apolinário, F., Escravana, N., Hervé, É., Pardal, M.L., Correia, M.: Fingerci: generating specifications for critical infrastructures. In: Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing. pp. 183–186 (2022)
7. Apolinário, F., Guiomar, J., Hervé, É., Hrastnik, S., Escravana, N., Pardal, M.L., Correia, M.: Comsec: Secure communications for baggage handling systems. In: European Symposium on Research in Computer Security. pp. 329–345. Springer (2022)
8. Brin, S., Motwani, R., Ullman, J.D., Tsur, S.: Dynamic itemset counting and implication rules for market basket data. In: Proceedings of the 1997 ACM SIGMOD international conference on Management of data. pp. 255–264 (1997)
9. Carvalho, O., Apolinário, F., Escravana, N., Ribeiro, C.: Ciia: critical infrastructure impact assessment. In: Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing. pp. 124–132 (2022)
10. Ceccato, M., Driouich, Y., Lucchese, M., Lanotte, R., Merro, M.: Towards reverse engineering of industrial physical processes (09 2022)
11. Feng, C., Palleti, V., Mathur, A., Chana, D.: A systematic framework to generate invariants for anomaly detection in industrial control systems (01 2019)
12. Feng, C., Tian, P.: Time Series Anomaly Detection for Cyber-physical Systems via Neural System Identification and Bayesian Filtering. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. KDD '21, Association for Computing Machinery, New York, NY, USA (2021)

[2] H2020 STARLIGHT https://www.starlight-h2020.eu/

13. Fung, C., Srinarasi, S., Lucas, K., Phee, H.B., Bauer, L.: Perspectives from a comprehensive evaluation of reconstruction-based anomaly detection in industrial control systems. In: Atluri, V., Di Pietro, R., Jensen, C.D., Meng, W. (eds.) Computer Security – ESORICS 2022. pp. 493–513. Springer Nature Switzerland, Cham (2022)
14. Grandini, M., Bagli, E., Visani, G.: Metrics for multi-class classification: an overview. arXiv preprint arXiv:2008.05756 (2020)
15. Hajj, S., El Sibai, R., Bou Abdo, J., Demerjian, J., Makhoul, A., Guyeux, C.: Anomaly-based intrusion detection systems: The requirements, methods, measurements, and datasets. Transactions on Emerging Telecommunications Technologies **32**(4), e4240 (2021)
16. Huang, Y.L., Cárdenas, A.A., Amin, S., Lin, Z.S., Tsai, H.Y., Sastry, S.: Understanding the physical and economic consequences of attacks on control systems. International Journal of Critical Infrastructure Protection **2**(3), 73–83 (2009)
17. Kaouk, M., Flaus, J.M., Potet, M.L., Groz, R.: A review of intrusion detection systems for industrial control systems. In: 2019 6th International Conference on Control, Decision and Information Technologies (CoDIT). pp. 1699–1704 (2019)
18. Kiran, R.U., Reddy, P.K.: Novel techniques to reduce search space in multiple minimum supports-based frequent pattern mining algorithms. In: Proceedings of the 14th international conference on extending database technology (2011)
19. Kumbhare, T.A., Chobe, S.V.: An overview of association rule mining algorithms. International Journal of Computer Science and Information Technologies (2014)
20. Lee, E.A.: Cyber Physical Systems: Design Challenges. In: 2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC). pp. 363–369 (May 2008)
21. Lima, J., Apolinário, F., Escravana, N., Ribeiro, C.: Bp-ids: Using business process specification to leverage intrusion detection in critical infrastructures. In: 2020 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW). pp. 7–12. IEEE (2020)
22. Liu, B., Hsu, W., Ma, Y.: Mining association rules with multiple minimum supports. In: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 337–341 (1999)
23. Pal, K., Adepu, S., Goh, J.: Effectiveness of association rules mining for invariants generation in cyber-physical systems. In: 2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE). pp. 124–127. IEEE (2017)
24. Raschka, S.: Mlxtend: Providing machine learning and data science utilities and extensions to python's scientific computing stack. Journal of open source software **3**(24), 638 (2018)
25. Rubio, J.E., Alcaraz, C., Roman, R., Lopez, J.: Analysis of intrusion detection systems in industrial ecosystems. In: SECRYPT. pp. 116–128 (2017)
26. Umer, M.A., Mathur, A., Junejo, K.N., Adepu, S.: Generating invariants using design and data-centric approaches for distributed attack detection. International Journal of Critical Infrastructure Protection **28**, 100341 (2020)
27. Wolsing, K., Thiemt, L., Sloun, C.v., Wagner, E., Wehrle, K., Henze, M.: Can industrial intrusion detection be simple? In: Atluri, V., Di Pietro, R., Jensen, C.D., Meng, W. (eds.) Computer Security – ESORICS 2022. pp. 574–594. Springer Nature Switzerland, Cham (2022)
28. Yoong, C.H., Palleti, V.R., Maiti, R.R., Silva, A., Poskitt, C.M.: Deriving invariant checkers for critical infrastructure using axiomatic design principles. Cybersecurity **4**(1), 1–24 (2021)