An Anomaly Detection Framework for Digital Twin Driven Cyber-Physical Systems

Chuanchao Gao
gaoc0007@e.ntu.edu.sg
Nanyang Technological University
Singapore

Heejong Park
hj.park@ntu.edu.sg
Nanyang Technological University
Singapore

Arvind Easwaran
arvinde@ntu.edu.sg
Nanyang Technological University
Singapore

ABSTRACT

In recent years, the digital twin has been one of the active research areas in modern Cyber-Physical Systems (CPS). Both the digital twin and its physical counterpart, called a plant, are highly intertwined such that they continuously exchange data to reveal useful information about the overall system. Such class of CPSs need to be robust to various types of disturbances, such as faulty sensors and model discrepancies, since the interplay between the physical plant's operation and digital twin's simulation may lead to undesirable or even destructive effect. To address this problem, this paper introduces a flexible anomaly detection framework for monitoring anomalous behaviours in digital twin based CPSs. In particular, our approach integrates both the digital twin and data-driven techniques that detect and classify anomalous behaviours due to modelling errors (e.g. incomplete models) and sensor and physical system's faults. The framework can be deployed to any general CPSs without the full knowledge of the digital twin's internal model. Therefore, our method is amenable to various types of digital twin implementations that enhance the traditional data-driven anomaly detection mechanism. We demonstrate the performance of our approach using the Tennessee Eastman Process model. The experimental result shows our approach is able to effectively detect and classify anomaly sources from the physical plant, sensor and digital twin, even in the situation when a certain combination of multiple anomalies occur simultaneously.

CCS CONCEPTS

• **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability.

KEYWORDS

Cyber-Physical System, discrepancy monitoring, digital twin, anomaly detection

ACM Reference Format:
Chuanchao Gao, Heejong Park, and Arvind Easwaran. 2021. An Anomaly Detection Framework for Digital Twin Driven Cyber-Physical Systems. In *ACM/IEEE 12th International Conference on Cyber-Physical Systems (with CPS-IoT Week 2021) (ICCPS '21), May 19–21, 2021, Nashville, TN, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3450267.3450533

ICCPS '21, May 19–21, 2021, Nashville, TN, USA

ACM ISBN 978-1-4503-8353-0/21/05.
https://doi.org/10.1145/3450267.3450533

1 INTRODUCTION

With the advancement of technologies used in today's Cyber-Physical Systems (CPSs), the concept of digital twin has emerged and its demand in CPS has been increasing significantly for the past several years [19]. Digital twin is a virtual replica of a real world object that can accurately monitor, predict and optimise processes of the twin's physical counterpart. Due to its ability to accurately imitate the physical process, many recent CPS applications, including aircraft [24], manufacturing [2, 8] and healthcare [3] etc., have been adopting digital twin based solutions. One of the main challenges for such CPSs, where the results of digital twin simulations and physical processes are highly interdependent, is their ability to detect potential anomalies that can result in undesired effects such as system failures.

In CPSs, anomalies are the observed sensory data affected by attacks, faults, etc and are not conforming to the well-defined system's behaviour. Anomaly detection for CPSs has been widely discussed in the literature due to the high demand for early detection and prevention of failures that can cause significant economic losses. The existing anomaly detection approaches can be classified into two categories: model-based and model-free [22]. In model-based approaches, a model is constructed from the prior knowledge of the underlying system using the first principle of physics. These models are then used to construct a residual generator that measures the degree of abnormality by comparing the online data collected from the physical system with the model's predictive value [6]. On the other hand, model-free or data-driven approaches employ classifiers that identify the types of abnormalities based on the online process measurements [7]. The performance of the model-free approaches, in general, depends on the quality of the offline training data, which is used for estimating the classifier.

In general, model-free approaches do not require expert knowledge of the target system, however, the availability of anomalous data is often limited that makes generating accurate classifiers hard. This makes the model-free approaches less practical to determine the anomalous sources. On the other hand, the model-based approaches have the benefit of being able to accurately detect anomalies given that such models are available or can be efficiently created by designers. Nevertheless, the model-based approaches generally suffer scalability issues and validation difficulties such that it is hard to be applied in large-scale systems. Besides, in the digital twin embedded CPS, the model synchronisation with real-time data and changes in the physical system's operating conditions (*modes*) might not be captured by the digital twin, which can cause synchronisation error. Therefore, the classical model-based approaches alone, which mostly assume the models are correct, are

not well-suited for anomaly detection for the digital twin based CPSs.

By combining digital twin simulation with streaming data observed from the physical system, our objective is to introduce an anomaly detection framework that can identify the sources of anomalous events effectively. Our approach is different from the traditional anomaly detection techniques, which mostly only consider anomalies in the physical plant whereas we consider more realistic scenario where the digital twin is an imperfect artefact. More precisely, the digital twin in our framework can model only a subset of the physical system's behaviour. This is mainly because a complete *fault-free* data from the physical plant is not always available. Moreover, exploring the complete state of the digital twin model is not always possible due to the problem of the model complexity.

As presented in our experiments on the Tennessee Eastman Process model [4], our framework has the capability to detect and differentiate anomalies caused by sensor faults, missing modes in the digital twin, or physical system failure. Besides, the anomalies caused by the simultaneous occurrence of the sensor faults and missing modes in the digital twin model can also be detected. In particular, it took by average 212 sampling delays for our approach to detect anomalies from their first occurrences when the anomalies are caused by both the plant and twin whereas 144.75 sampling delays for the sensor faults only.

The specific contributions of this paper are: (1) A novel anomaly detection and classification framework for large-scale digital twin embedded CPSs where the sources of anomalies can be either or both the digital twin and physical plant; (2) A detection mechanism that employs digital twin model as well as streaming data from the physical system to classify the anomalous behaviour; (3) Application of the approach to the realistic scenario that is based on the chemical process plant.

Our framework does not rely on specific modelling techniques for digital twin implementation. In fact, we assume these models are *black-box* that can only read and output time series data similar to the model containers in the Functional Mock-up Interface (FMI) standard [5]. We believe this is a reasonable assumption since it is often the case that the model vendors might hide the internal implementation of their models such as algorithms to protect their IPs. Moreover, this allows our framework to be applied to various scenarios without the need for modifying the existing CPS settings.

Our view on the importance of the introduction of the anomaly detection framework to the digital twin embedded CPSs is twofold. First, as previously mentioned, while the primary objective of the traditional fault detection and identification (FDI) approaches is to detect faults or anomalous behaviours in the physical system, the anomaly detection problem for the digital twin based CPSs also considers modelling issues. Second, there is the need for a quick and an efficient anomaly detection technique for the digital twin based CPSs which can select appropriate correction mechanism(s) to keep the system in the safe state. For example, there exist resilient control algorithms for uncertain models [10] adversarial attacks [14] or sensor/actuator faults [12]. Although the primary focus of our work is *not* on the correction mechanism but on the general framework for the anomaly detection, we foresee our approach can be merged with such control algorithms to avoid potential system failures.

The rest of the paper is organised as follows. Section 2 enumerates related works on the digital twin based anomaly detection techniques. An overview of our anomaly detection framework for digital twin based CPSs is presented in Section 3. The detailed description of individual components in the framework is given in Section 4. Experimental results that applying our technique on the real-world example based on the chemical plant are shown in Section 5. Finally, the paper concludes in Section 6.

2 LITERATURE REVIEW

In recent years, there have been several works that employ digital twin in anomaly detection scenarios. In [18], authors introduced a digital twin based anomaly detection system in operation and management (O&M) of buildings and civil infrastructures. In their work, the industry foundation classes (IFC) for the O&M data model are extended to integrate the digital twin for anomaly detection. The system uses the Bayesian online change point detection to identify suspicious anomalies. The introduced technique targets specifically for the asset management in construction and facility management whereas we focus more on the broad categories of the digital twin based CPSs.

A digital twin model construction and anomaly detection methods by observing indirect side-channels from the physical system is proposed in [8]. Their technique localises anomalies by comparing digital twin fingerprints generated from the side-channel emissions, such as acoustic, vibration and magnetic data etc., with the set of observable runtime features from the physical system. Nevertheless, their approach requires the existence of side-channel emissions from the physical system, which is not always the case, and do not consider the possibilities of the anomalies in the twin model itself.

Several recent works show the use of digital twins for anomaly detection in a wide range of engineering fields such as additive manufacturing (AM) [2, 23], aircraft [24] and industrial plants [17]. A real-time performance monitoring and anomaly detection in fused deposition modelling is proposed in [2]. The authors model a digital twin as a discrete and continuous dynamics using the process measurement data specified in signal temporal logic (STL). Based on the fixed window size for observed data, their approach validates if the monitored data violate the STL logic, which is the detection mechanism for anomalies. In [23], authors introduced a multi-modal sabotage attack detection system for AM machines. Similar to [8], they observe analogue side-channel emissions for detection of adversarial attacks and uses a simple threshold check between the observed data and the ideal physical dynamics described in G-code. A monitoring framework for a fleet of aircrafts is introduced in [24]. Similar to our work, their framework consists of sensor fault detection, and threshold monitoring for anomaly detection and fault identification components. However, their approach requires a detailed digital twin model of the turbofan engine and domain specific knowledge to generate fault signatures. Digital twin based virtual sensor generation for a gas turbine engine is proposed in [17]. Based on a pool of redundant sensors, the authors employ Bayesian hierarchical models to simulate a batch of healthy sensors for anomaly detections for spikes and bias faults.

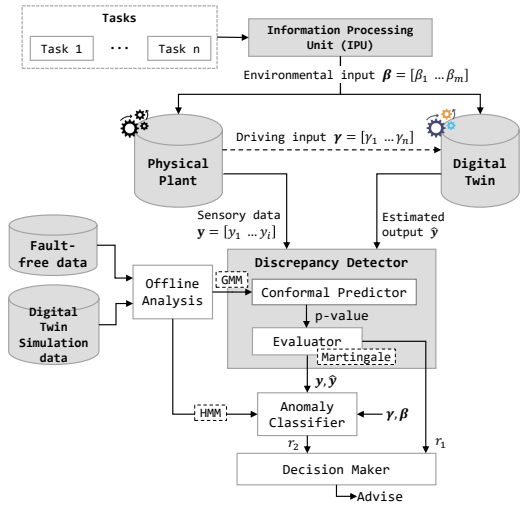

Figure 1: An overview of the anomaly detection framework for the digital twin based CPS

To summarise, most of the existing works assume the digital twin as a 'ground truth' and classify deviation from this as a fault or anomalous behaviour. Our work consider the twin as imperfect component in the framework and it can have missing behaviours due to human errors or impractical to cover all possible reachable states during the analysis phase, which is more realistic for most of the real and large complex systems. Furthermore, our framework aims to more general types of digital twin based CPSs, allowing designers to replace components used in the detection mechanism for their needs.

3 ARCHITECTURE

3.1 An Overview of the System

The architecture of the anomaly detection framework for digital twin based CPSs is shown in Figure 1. Our system consists of a physical plant, digital twin and a set of tasks, which are the specific operations to be carried out by the physical plant. Examples of tasks are the events that trigger a vehicle to move from a point A to B or a robot picking up an item at a loading station and so on. Such tasks are considered *environmental inputs*, generated from the outside of the system, to both the plant and digital twin. The conversion of the tasks to the environmental inputs is done by the Information Processing Unit (IPU). In this work, we consider any types of physical plant whose behaviour can be monitored via a set of sensors, which generate time series data with a fixed period T_i for each sensor $i \in \mathbb{Z}_{\geq 0}$. Here, we formalise each component in our framework shown in Figure 1.

Definition 1 (Digital Twin). *Our digital twin is considered as a function f_{q_k} that accepts $\mathrm{u} \in \mathbb{R}^{n+m}$, which consists of driving γ and environmental β inputs:*

$$\begin{aligned} \hat{\mathrm{x}}_{k+1} &= f_{q_k}(\hat{\mathrm{x}}_k, \hat{\mathrm{u}}_k) \\ \hat{\mathrm{y}}_k &= g_{q_k}(\hat{\mathrm{x}}_k, \hat{\mathrm{u}}_k) \end{aligned} \tag{1}$$

where

$$\hat{\mathrm{u}} = \begin{bmatrix} \gamma_1 & \dots & \gamma_n & \beta_1 & \dots & \beta_m \end{bmatrix}^T$$

$\hat{\mathrm{x}} \in \mathbb{R}^{n+m}$ and g_{q_k} are, respectively, the internal state of the twin and the output function. An output vector of the digital twin $\hat{\mathrm{y}}_k$ is used for checking discrepancy between the twin model and it's physical plant. To build a digital twin based CPS that is scalable, we allow only a partial of the system to be modelled in f_{q_k}. In such case, digital twin may require some inputs from the physical plant to simulate the partial system. For example, if digital twin only models a robot manipulator that transfers a workpiece between two locations, an event that captures the arrival of the workpiece to the robot arm is a driving input which should be provided by the physical plant. In case if the twin models the complete system, the driving input is empty.

The digital twin we consider in this work is a *hybrid system* whose operating mode at time k is captured by one of its sub-models $q_k \in \{1, \dots, s\}$. More precisely, we consider the class of switched (non-)linear systems whose switching logic is built in the digital twin model.

Definition 2 (Physical plant). *is defined similarly as in Eq. (1):*

$$\begin{aligned} \mathrm{x}_{k+1} &= f^p_{q_k}(\mathrm{x}_k, \mathrm{u}_k) \\ \mathrm{y}_k &= g^p_{q_k}(\mathrm{x}_k, \mathrm{u}_k) + \mathrm{w}_k + \mathrm{e}_k \end{aligned} \tag{2}$$

where

$$\mathrm{u} = \begin{bmatrix} \beta_1 & \dots & \beta_m \end{bmatrix}^T$$

is an input vector that consists of environmental inputs β_m only. w_k and e_k are the sensor noise and error vectors, respectively. The output vector

$$\mathrm{y}_k = \begin{bmatrix} y_1 & \dots & y_i & \gamma_1 & \dots & \gamma_n \end{bmatrix}^T$$

consists of observable sensory data $y_1, \dots, y_i$ which are directly comparable with $\hat{\mathrm{y}}_k$ from the digital twin for discrepancy check. The part $\gamma_1, \dots, \gamma_n$ are the driving inputs to the digital twin as explained previously. In this framework, we do not interfere control logic within the plant and digital twin since our framework is focused on anomaly detection and, therefore, the control signals are not considered in u and $\hat{\mathrm{u}}$. Instead, we assume they are observed in y and $\hat{\mathrm{y}}$.

3.2 Sources of Anomalies

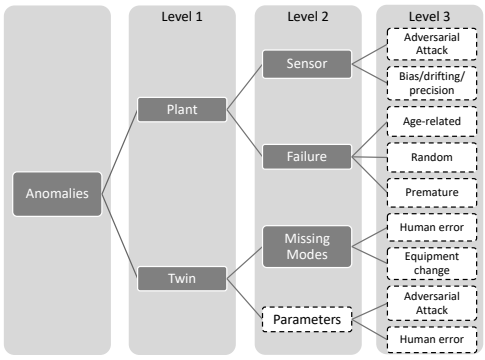

Figure 2: Examples of the sources of anomalous behaviour

One possible method to test an anomalous behaviour at time k is:

$$\| [y_1, \dots, y_i]^T - \hat{\mathrm{y}}_k \| \geq \varepsilon \tag{3}$$

for some ε. However, this test alone cannot classify the source of anomalies, whether it is from the plant or digital twin, nor if the anomaly is temporary (transient) or permanent. Therefore, we use two specialised components in our framework: (1) a *discrepancy detector* that first checks discrepancies between the plant's sensory data and estimated outputs from the digital twin and (2) an *anomaly classifier* that further identifies the sources of anomalies.

A categorical overview of anomalous behaviours that can arise in the digital twin based CPS is shown in Figure 2. We consider the sources of anomalies that are divided into three levels. Depending on how concrete the anomaly sources can be identified to specific levels, our framework can generate a suggestion which correction algorithm should be used to eliminate the problem, if possible, or alert an administrator if the anomaly is permanent (i.e. unable to fix).

Depending on how specific the anomalies can be classified, we say our classifier is level 1, 2 or 3. For example, the level-1 classifiers are simply able to tell if the anomaly is from the plant or the twin. On the other hand, the level-2 classifier can further categorise, for example, if the anomalies are due to physical sensor faults, or inaccurate models in the digital twin. Components used for the classifier can be replaced based on the designer's need. In this paper, we present one concrete example of anomaly classifier that determines if the source of anomaly is from the plant, a missing mode in the twin or sensor faults. In Figure 1, this is shown as grey coloured nodes. Therefore, our anomaly classifier falls in the level-2 category. In particular, we employ Gaussian Mixture Model (GMM) and the exchangeability test using martingale [15] for quick discrepancy detection and further classification using Hidden Markov Model (HMM). In the next section, we present detailed description of individual components shown in Figure 1.

4 DETAILS OF THE COMPONENTS USED IN THE FRAMEWORK

The framework consists of several components that perform specific tasks as illustrated in Figure 1. Each component, depicted by rectangular boxes, can be replaced with different approaches depending on the application's requirements on the accuracy and detection speed trade-off etc. In this section, we present concrete examples that are suitable for our need, which is the anomaly detection for digital twin based chemical process plant.

The first step of our anomaly detection workflow is to preprocess data obtained from both the plant and digital twin. We assume the availability of *fault-free* data, which is collected from the physical plant from its previous operation. These data are *incomplete* meaning that they do not cover the complete operation of the target plant in each operation mode such that there could be missing data, which is often the case in most real-world scenarios. On the other hand, *simulated* data is obtained from the digital twin. Again, we do not assume the completeness of this data for each operation mode.

4.1 Clustering Fault-Free Data using K-medoids

Streaming data generated by the large-scale CPS are often characterised by high dimensional data. To make such data amenable to our analysis, the k-medoids algorithm [20] is applied to reduce the stream data dimension. Also, the k-medoids algorithm can group the more correlated variables within the same cluster and separate less correlated variables into different clusters. K-medoids algorithm is similar to k-means clustering but uses the data points as cluster centroids instead of their mean values. Therefore, the algorithm is less susceptible to outliers [21].

The input to the clustering algorithm is the fault-free data obtained from the physical plant:

$$D = \begin{bmatrix} y_1^1 & \cdots & y_\varphi^1 \\ \vdots & \ddots & \vdots \\ y_1^N & \cdots & y_\varphi^N \end{bmatrix} \tag{4}$$

where N is the number of variables v in the dataset and there are φ sampled data for each variable. We use notations y^i and D_i exchangeably to indicate an i'th row of D, which contains φ samples of the variable v_i.

Given $k \leq N$ clusters to be created from the dataset, the k-medoids algorithm associates each variable v_i to the nearest medoid v_j based on the pairwise dissimilarities $d_{i,j}$ between them. In this work, we use Pearson correlation coefficient for computing dissimilarities:

$$d_{i,j} = 1 - \frac{K_i (K_j)^T}{\sqrt{K_i (K_i)^T}\sqrt{K_j (K_j)^T}} \tag{5}$$

where $K_i = (D_i - \bar{D}_i)$ and D_i is the i'th row of D. We find medoids using the algorithm presented in [20]. The initial k medoids are selected based on the first k smallest values in the set:

$$\left\{ x_j \;\middle|\; x_j = \sum_{i=1}^{n} \frac{d_{i,j}}{\sum_{l=1}^{n} d_{i,l}}, 1 \leq j \leq N \right\} \tag{6}$$

where x_j is associated to the variable v_j. Given a set of initial medoids $V_m \subseteq \{v_1, \ldots, v_N\}$, initial clusters C_j are also created based on the distances of variables v_i to the nearest medoid:

$$C_j = \left\{ v_i \;\middle|\; v_i \in \{v_1, \ldots, v_N\},\ \arg\min_{v_j \in V_m} d_{i,j} = v_j \right\}, \quad j = 1, \ldots, N \tag{7}$$

Next, a new medoid v_i^{new} is to be found for each cluster i where $v_i^{new} = \arg\min_{v_k \in C_i} \sum_{j=1}^{|C_i|} d_{k,j}$ and a new cluster is formed using (7). The stopping condition for finding the next refined set of clusters is when two consecutive sums of distances of all variables to their respective medoids are equal.

To determine the optimal number of clusters for the k-medoids algorithm, we use the reconstruction error[1], whose detail is presented in Appendix A.

4.2 Building Models for Discrepancy Detector

After clusters are identified, we build models used by the components in the discrepancy detector. As shown in Figure 1, the detector consists of *conformal predictor*, which is used for generating a sequence of statistical p-values, and the *evaluator* that decides if the anomaly can be identified by directly inspecting the generated p-values. A detailed illustration of the workflow of the discrepancy detector is shown in Figure 3.

To generate p-values, we first build Gaussian Mixture Models (GMM) for each variable in the clusters. GMM training is done

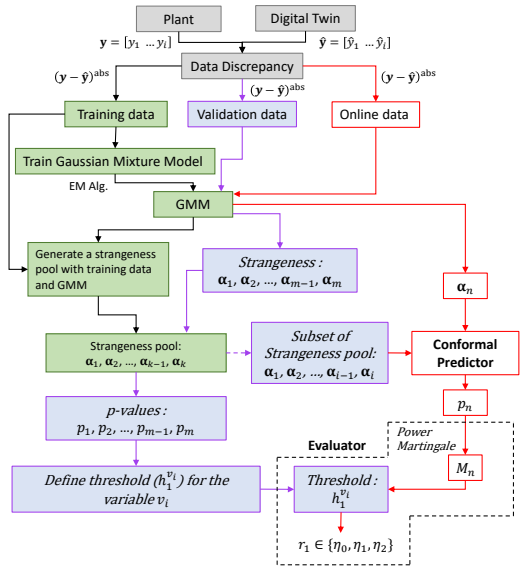

Figure 3: The workflow of the discrepancy detector employing GMMs and martingale tests

during the offline process, which is indicated by the green boxes in Figure 3. GMM is chosen in this work because we target the digital twin and physical system that each exhibits multimode behaviours as in Eq. (1) and (2). In GMM, the weighted sum of components $P(\mathrm{x}|\theta_m)$, where m is a component in the mixture distribution for a variable, is defined as:

$$P(\mathrm{x}|\Theta) = \sum_{m=1}^{M} \phi_m P(\mathrm{x}|\theta_m) \tag{8}$$

where $\mathrm{x} \in \mathbb{R}^{1\times n}$ is the n-dimensional vector, ϕ_m is the component weight and the parameter $\theta_m = \{\mu_m, \Sigma_m\}$ is a set of mean and covariance matrix. A set $\Theta \subseteq \phi^M \times \theta^M$ is a parameter for the Gaussian mixture. Since our fault-free and digital twin simulation data is incomplete, we use the Expectation Maximisation (EM) algorithm [9] to train GMM. The training dataset to the algorithm is obtained from the sequence of the absolute differences between two data samples from the plant and digital twin $(\mathrm{y}_k - \hat{\mathrm{y}}_k)^{\mathrm{abs}}$.

Once GMM for a variable is trained, we compute a validation set of p-values, which is considered as 'normal' behaviours for the plant and digital twin. This is another batch of offline process denoted as blue boxes in Figure 3. We first populate a strangeness pool, which is a sequence of nonconformity measures α_k generated by the GMM from the training dataset $(\mathrm{y}_k - \hat{\mathrm{y}}_k)^{\mathrm{abs}}$. Next, we obtain another sequence of nonconformity measures α_m from a validation dataset. Then a validation set of p-values $\{p_1, \ldots, p_m\}$ is generated from α_k and α_m via

$$p_m = \frac{|\{k : \alpha_k > \alpha_m\}| + c_m|\{k : \alpha_k = \alpha_m\}|}{m} \tag{9}$$

where $c_m \in [0, 1]$ is some random number.

We compute a martingale [15] from the validation set of p-values and define a threshold limit $h_1^{v_i}$ for the martingale that is considered abnormal for v_i. The general form of martingale

$$S_n = \prod_{i=1}^{k} f_i(p_i), \quad k = 1, 2, \ldots \tag{10}$$

computes the expected value of S_{k+1}. The martingale theory states $S_k \geq 0$ and $S_k = \mathrm{E}(S_{k+1} \mid S_0, S_2, \ldots, S_k)$, which means the expected S_{k+1} to be equal to the most recent observation of S_k. The betting function f_i defines the growth rate of the martingale. Here, we use a constant $\varepsilon \in [0, 1]$ in the betting function and make the *power martingale* which is defined as

$$M_k^{\varepsilon} = \prod_{i=1}^{k} \varepsilon p_i^{\varepsilon - 1} \tag{11}$$

The equation indicates that M_k^{ε} only grows for large number of small p-values. For large p-values, the martingale will not grow significantly and it becomes harder to reject the exchangeability between the data from the plant and digital twin. The martingale threshold $h_1^{v_i}$ for v_i is computed using:

$$h_1^{v_i} = \max\{M_1^{\varepsilon}, \ldots, M_k^{\varepsilon}\} \times \mu_{v_i} \tag{12}$$

where $\mu_{v_i} > 1$ is a constant parameter which can be selected by the performance of validation data.

During the online phase, the evaluator checks if the martingales computed from the online data indicate anomalies, i.e. there exist discrepancy between the data generated from the plant and digital twin using $h_1^{v_i}$ for each v_i. This is indicated by the red boxes in Figure 3.

As in [1], we consider anomalies detected in each cluster as (1) bounded or (2) multiple anomalies. Bounded anomalies are the ones that rarely occur in a subset of variables in each cluster, for example due to sensor faults. The intuition behind such types of anomalies is that they are highly unlikely to occur simultaneously. On the other hand, the source of multiple anomalies are most likely due to the faults in the plant and/or digital twin where the large proportion of anomalies would be observed from the clustered data stream.

To this end, we set the maximum number (typically small) of variables, denoted as ξ_i^1, for each cluster whose martingales can grow greater than $h_1^{v_i}$ before they are considered as multiple anomalies. The evaluator generates a signature r_1 which indicates the type of anomalies based on the following method:

$$r_1 = \begin{cases} \eta_0 & \text{if } x_i = 0 \\ \eta_1 & \text{if } 0 < x_i < \xi_i^1 \\ \eta_2 & \text{if } x_i \geq \xi_i^1 \end{cases} \tag{13}$$

where $x_i = |\{v_j \mid v_j \in C_i, M_{k,v_j}^{\varepsilon} \geq h_1^{v_j}\}|$ for all clusters C_i. The signatures η_0, η_1 and η_2 indicate, respectively, normal, bounded and multiple anomalies detected by the evaluator.

The signature generated from the evaluator merely tells that it has detected an anomaly without classifying its source. In particular, we would like to identify if the anomalies are due to the sensor/plant faults or missing modes (equivalently sub-models) in the digital twin. This task is done by the anomaly classifier as shown in Figure 1, which is described in the next section.

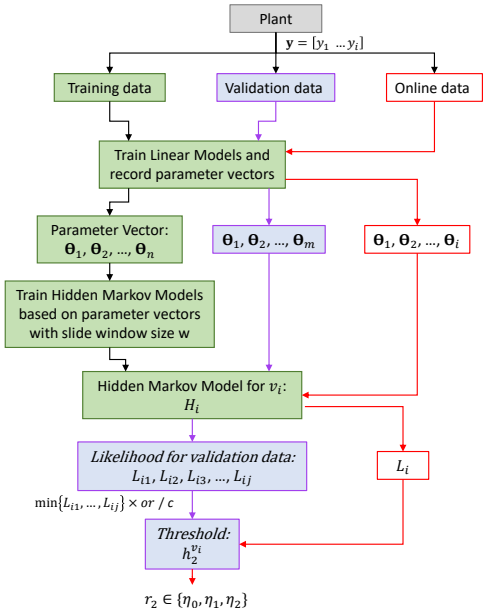

Figure 4: The workflow of the anomaly classifier employing HMM

4.3 Building the Anomaly Classifier

The role of the anomaly classifier is to identify the source of the anomalies, which is first observed from the discrepancy detector. In this work, we use HMM-based anomaly classifier, which is built from the fault free data during the offline analysis. A workflow of the classifier is shown in Figure 4. Similar to the discrepancy detector, this workflow also consists of training, validation and online phases, which are depicted as green, blue and red boxes respectively.

In our framework, HMM is a pair $H = \langle Q, A, O, B, \pi \rangle$ where Q is a finite set of n states, $A \in \mathbb{R}^{n \times n}$ is a transition matrix representing an unobservable Markov chain and A_{ij} indicates a transition probability from state i to state j and $\sum_{j=1}^{n} a_{ij} = 1$. O is a finite set of possible observations $\Theta \in \mathbb{R}^m$, which is a vector of coefficients for the linear models with m degrees. These models are trained from the output sequences of the plant (i.e. segmented rows of D with a fixed size). $B \in \mathbb{R}^n$ is an observation likelihood where $B_i = b_i(\Theta_i)$, $\Theta_i \in O$ is the probability of an observation Θ_i being generated from the state i. $\pi \in \mathbb{R}^n$ is the initial probability array where π_i indicates the probability that the Markov chain will start from the state i.

During the training phase, H_i is trained for each variable v_i (i'th row of D) for the plant. Then, using the validation dataset in D, which is non-overlapping with the training dataset, we compute the log-likelihoods L_{ij} for all sequences $j = (\Theta_1, \ldots, \Theta_n)$ observed from the variable v_i:

$$L_{ij} = \log\left(P(\Theta_n | q_n)\right) \tag{14}$$

where

$$P(\Theta_n | q_n) = \prod_{i=1}^{n} P(\Theta_i | q_i)$$

Then for each v_i, a log-likelihood threshold limit $h_2^{v_i}$ is computed using:

$$h_2^{v_i} = \begin{cases} \min\{L_{i1}, \ldots, L_{ij}\} \,/\, c & \text{if } \min\{L_{i1}, \ldots, L_{ij}\} \geq 0 \\ \min\{L_{i1}, \ldots, L_{ij}\} \times c & \text{if } \min\{L_{i1}, \ldots, L_{ij}\} < 0 \end{cases} \tag{15}$$

where c is a constant parameters which can be selected by the performance of validation data.

During online, a log-likelihood of a sequence of Θ for v_i, denoted as L'_{v_i}, is compared with $h_2^{v_i}$ in each cluster to generate a signature similar to (13):

$$r_2 = \begin{cases} \eta_0 & \text{if } x_i = 0 \\ \eta_1 & \text{if } 0 < x_i < \xi^2_{v_i} \\ \eta_2 & \text{if } x_i \geq \xi^2_{v_i} \end{cases} \tag{16}$$

where $x_i = |\{v_j \mid v_j \in C_i, L'_{v_j} < h_2^{v_j}\}|$ for all clusters C_i. The signature generated by the classifier indicates more concrete sources of anomalies when combined with the signature produced from the discrepancy detector. A method for building a decision maker based on these signatures is explained in the next section.

4.4 Decision Maker for Generating a Suggestion

Table 1: A table for interpreting signatures generated by the evaluator and the anomaly classifier

r_1	r_2	Description
η_0	–	No anomalies detected
η_1	η_1	Sensor faults
η_1	η_0 or η_2	Unclassified
η_2	η_0	Digital twin fault
η_2	η_1	Digital twin fault + sensor fault
η_2	η_2	Plant fault

Based on the signatures r_1 and r_2 generated by the evaluator and the anomaly classifier, a suggestion to choose a correction mechanism is made by the decision maker. Table 1 shows how the signatures are interpreted by the decision maker.

The decision maker can conclude that there is no anomalies in the system, regardless of the result of the anomaly classifier, if the evaluator generates $r_1 = \eta_0$. If the evaluator indicates bounded anomaly (η_1) but the classifier says otherwise, the decision maker concludes this as unclassified behaviour since the classifier contradicts the result of the evaluator. As explained in Section 4.2, the bounded anomalies are considered as sensor faults and this is validated by both the evaluator and the anomaly classifier ($r_1 = \eta_1$ and $r_2 = \eta_1$). The source of multiple anomalies indicated by evaluator ($r_1 = \eta_2$) is identified by the classifier as shown in Table 1. It is possible that both the sensor and digital twin can exhibit anomalous behaviours simultaneously, which is indicated by $r_2 = \eta_1$. This is when the evaluator detects multiple anomalies from the data observed from both the plant and digital twin (using GMM) but the anomaly classifier only detects bounded anomalies from the plant data (using HMM).

Our framework allows customising the decision maker to generate a specific suggestion. This suggestion can be used by external

components to trigger an appropriate correction algorithm. For future work, we plan to integrate some of the standard algorithms in the framework such as attack resilience control [14][hidden for blind review] and runtime parameter estimation for models [13]. In the next section we present the effectiveness of our anomaly detection framework using the model of the chemical process plant.

5 EXPERIMENTAL RESULTS

The experiment was carried out to evaluate the performance of our approach on four different types of anomalies: anomalies due to (1) the sensor faults, (2) the plant fault, (3) the absence of sub-models (modes) in digital twin and (4) the combination of (1) and (3). All the experiments run on Intel Core(TM) i7-8650U CPU (1.9Ghz) and 16GB of RAM.

5.1 Performance Measurement Criteria

To evaluate the performance of our anomaly detection framework, the following three criteria are used:

- Precision: TP / (TP + FP), the percentage of anomalies detected which are real anomalies among all anomalies detected.
- Recall: TP / (TP + FN), the percentage of anomalies detected which are real anomalies among all real anomalies.
- Detection Delay (DD): the time delay (**in number of data samples**) between the occurrence of the anomaly and the time it is detected by the classifier.

where TP (True Positive) is the number of correctly detected anomalies, FP (False Positive) is the number of incorrectly detected anomalies and FN (False Negative) is the number of anomalies that are failed to be detected. The Detection Delay is present in the form of minimum, maximum, and mean values of the experiment results.

5.2 Building the Framework Components

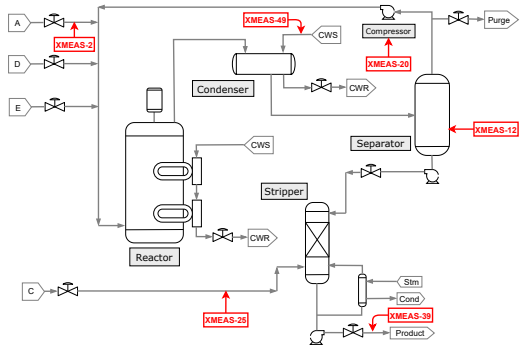

Figure 5: Tennessee Eastman Process Model

The whole experiment was conducted with the Tennessee Eastman Process (TEP) model., whose layout is shown in Figure 5. Due to space limitations, detailed description of the model is given in Appendix B. During the process model simulation, the measurement noise was injected to the plant model to emulate the physical plant sensors, On the other hand, the data collected from the simulation without the noise were considered as the digital twin's predictive values of the chemical process. In this experiment, the discrepancy data caused by the inaccurate simulation of digital model are not considered due to the limitation of the experiment model. The disturbance-free data gathered from the model simulated in the base mode were used as fault-free data for variable clustering. The same data were used for training and validation of the discrepancy detector as well as the anomaly classifier.

The simulation time of the base mode operation was 72 hours with a sampling rate of 0.01 hours per sample, thus, the simulation contains 7200 data samples in total. The data fluctuation in the transient period is much larger than the lateral processing period, thus, the transient period is considered as another operation mode compared with the processing period. In this experiment, we only consider process period during the framework components training, and the first 200 data samples were discarded from the physical plant's sensory data D_{pp} and digital twin's predictive values D_{dt} to exclude the transient period data. The remaining 7000 data samples were divided into two parts: the training data set (denoted as T_{pp} for D_{pp} and T_{dt} for D_{dt}) and the validation data set (denoted as V_{pp} for D_{pp} and V_{dt} for D_{dt}), where the training data set contains the first 4600 data samples and the validation data contains the last 2400 data samples.

During the variable clustering, T_{dt} was used to find the least cost clusters for a given cluster number k, and V_{dt} was used to determine the optimal k value. T_{pp} and T_{dt} were used to train the GMMs in the discrepancy detector and to populate the strangeness pool. The martingale threshold $h_1^{v_i}$ was calculated from V_{pp} and V_{dt}. Only D_{pp} was used by the anomaly classifier where T_{pp} was used for the HMM training and V_{pp} was used for parameter vectors log-likelihood threshold calculation.

The ε for the power martingale shown in Eq. (11) was set to 0.92. Both the constants ξ_1^i and ξ_2^i in Eq. (13) and (16) were set to 2 for the clusters with size less than 4 or 30% of the cluster size when this size was greater than 4. For the anomaly classifier, every linear model was trained with 200 continuous data samples, and the sliding window size w for HMM training was set to 100. The value of $\xi_2^{v_i}$ in Eq. (16) was set to be same as ξ_1^i. In all the experiments, different values of μ_{v_i} in Eq. (12) and c in Eq. (15) were selected to determine the the influence of their values on the performances of discrepancy detector and anomaly classifier.

5.3 Experiment 1: Anomalies from the Sensor Measurements

In this experiment, the following types of anomalies in the sensor measurements were considered [16]: (1) *Drift Faults*: sensor output value keeps increasing or decreasing compared with the normal state; (2) *Offset Faults*: sensor output value jumps to a higher level or drops to a lower level, but remains the similar trend; (3) *Erratic Faults*: variation of the sensor output significantly increases above the usual value; (4) *Spike Faults*: spikes are observed in the sensor output intermittently; (5) *Stuck Faults*: the sensor output gets stuck at a fixed value.

A disturbance-free data set gathered from TEP model operating in base mode was used as fault-free data in this experiment, and 6 variables out of the 73 recorded variables were chosen as abnormal sensors in the sensor measurement anomaly detection experiment.

The 6 chosen variables were D feed (XMEAS-2), product separator level (XMEAS-12), compressor work (XMEAS-20), C reactor feed (XMEAS-25), F product component (XMEAS-39), and condenser cooling water flow (XMEAS-49)[11], which are in 6 different clusters in the result of variables clustering, and the sensor physical location is shown in Figure 5. The 5 types of sensor measurement anomalies were injected into the fault-free data separately for every chosen variable.

Table 2: Sensor Measurement Anomaly Detection Summary

	Discrepancy Detector Threshold, μ_{v_i}				Anomaly Classifier Threshold, c		
	1.1	1.2	1.5	2.0	1.1	1.25	2.0
Precision	100%	100%	100%	100%	65.1%	79.4%	87.5%
Recall	90%	87%	90%	90%	93.3%	90%	93.3%
DD	(3,326,46)	(4,299,41)	(3,362,48)	(4,399,51)	(3,91,21)	(4,316,31)	(4,474,74)

The results of the experiment on the sensor anomaly detection are shown in Figure 6 and Table 2. As shown in Figure 6, all drift, offset and erratic faults can be identified by the both discrepancy detector (DD) and anomaly classifier (AC). Besides, all spike faults are detected by AC while some of the spike faults are missed by DD, which is caused by the design nature of the exchangeability martingale (EM) based discrepancy detector, as EM is more sensitive for continuous anomalies and less sensitive to random spike. Also, DD is able to detect all stuck faults while the AC misses some of the stuck faults. This is because the HMM based anomaly classifier tends to detect the change of linear relationship among all variables in a cluster. If the stuck value does not change the linear relationship, the HMM could not easily detect the anomaly as shown in the results. Our framework allows other statistical tools to replace EM and HMM methods used in DD and AC to improve performance of the applications where the spike faults or stuck faults are dominated.

Table 2 gives a summary of the framework performance with different values of threshold constants μ_{v_i} and c. The result shows that the small range adjustment of μ_{v_i} does not affect the discrepancy detector performance significantly, while the increasing in c values resulted in the increase of detection delay time. In addition, the synchronously increasing of precision and constant c shows that larger threshold constant of anomaly classifier can reduce the false alarm rate.

5.4 Experiment 2: Anomalies from the Faults in the Physical Plant

Examples of the physical plant faults considered in this paper are the abnormal changes in the chemical flow rate, environmental changes (e.g. temperature and pressure), and physical damages of the machine components. As stated in Appendix B, 28 different disturbances, which are considered as anomalies due to the faults in physical system, were introduced to the TEP model. The 28 different disturbances can be categorized into four types: step change, random variation, slow drift, and sticking valve. In this experiment, 4 disturbances from 4 different types were selected: (1) *Step change*: Feed loss of chemical A, IDV-6; (2) *Slow drift*: Reaction kinetics change, IDV-13; (3) *Random variation*: Chemicals A and C feed pressure, IDV-26; (4) *Sticking valve*: reactor cooling water valve sticking, IDV-14, and this disturbance is combined with D feed temperature anomaly with type of step change, IDV-3, as suggested by the model designer.

In this experiment, TEP model operating data in the base mode with measurement noise and disturbance was used as physical plant data. The operating data without noise and disturbance was used as digital twin simulation data. In addition, the samples from the 2000th data point and onwards were used to avoid the influence of the process model initial unstable states on experiment results.

Table 4: Physical Plant Anomaly Detection Summary

Disturbance	Decision Maker	DD for different thresholds, μ_{v_i} & c			
		1.2	1.5	2	3
IDV-6	Physical Plant	77	125	177	200
IDV-13	Physical Plant	24	28	32	128
IDV-3 & IDV-14	Physical Plant	40	53	73	682
IDV-26	Physical Plant	1253	5282	5287	5298

The experiment results are shown in Tables 3 and 4. According to Table 3, no false alarm has been detected by the discrepancy detector, and the anomaly classifier is shown to be more sensitive to step change and slow drift anomaly types, and more robust to random variation and sticking valve anomaly types. The result of Table 4 shows that the decision maker can still make correct predictions when anomaly classifier does not perform well, which guarantees the effectiveness of the whole framework. This is because the decision maker makes the estimation based on whether there exists any clusters to be found abnormal, but not how many clusters to be found abnormal. Furthermore, with the increasing of the threshold constant μ_{v_i} and c, the detection delay time of decision maker is also increasing, which is reasonable since the probability thresholds will decrease with the increasing of threshold constants, and it requires more data samples for discrepancy detector and anomaly classifier trend down to a lower probability value.

5.5 Experiment 3: Anomalies Due to the Lack of Modes in the Digital Twin Model

Physical processes may exhibit different operating behaviours based on different application requirements. In modelling terminology, this is akin to *mode switching* in the hybrid system. This experiment considers anomalies due to the changes in the operating mode of the physical plant while the digital twin model does not change to the corresponding mode in time due to the missing of new mode.

The same disturbances from Section 5.4 were used for emulating the digital twin anomalies due to the missing modes. Contrary to Section 5.4, the data with only measurement noise was considered as the physical plant data, whereas the noise-free data with disturbance was considered as the twin anomalies from the missing modes.

The results of digital twin anomaly detection are shown in Tables 5 and 6. Table 5 presents the details of digital twin mode missing anomaly detection result. The result of anomaly classifier is not present due to no anomaly exists in the physical plant data. As in Table 5, the discrepancy detector is able to detect the discrepancies caused by mode missing anomaly in almost 100% success rate, which is similar with the the discrepancy detector performance in Section 5.4. However, the table result also indicates the change of

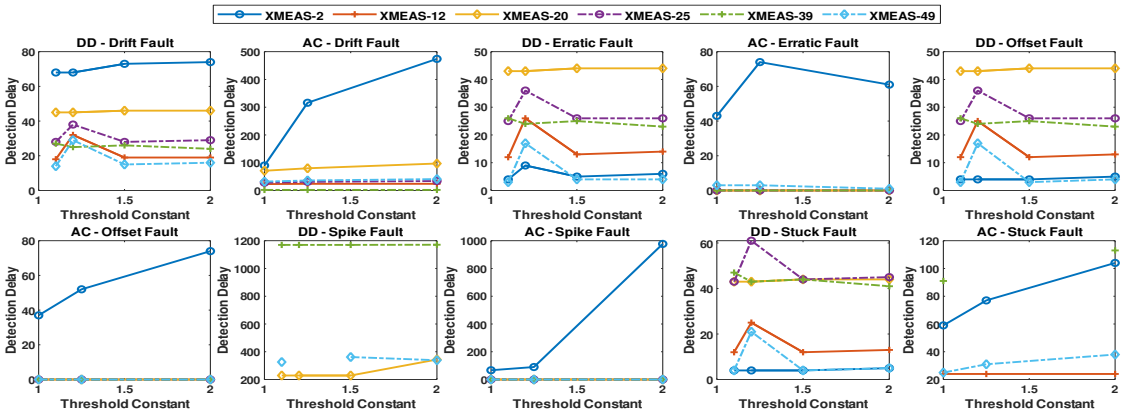

Figure 6: Sensor Measurement Anomaly Detection Details

Table 3: Physical Plant Anomaly Detection Details

Disturbance		Discrepancy Detector Threshold, μ_{v_i}				Anomaly Classifier Threshold, c			
		1.2	1.5	2.0	3.0	1.2	1.5	2.0	3.0
IDV-6	Precision	100%	100%	100%	100%	75%	75%	71.4%	60%
	Recall	100%	100%	100%	100%	100%	100%	83.3%	50%
	DD	(8, 216, 104)	(9, 216, 106)	(9, 217, 107)	(11, 217, 108)	(8, 574, 229)	(9, 598, 241)	(9, 598, 207)	(11, 305, 149)
IDV-13	Precision	100%	100%	100%	100%	62.5%	71.4%	71.4%	66.7%
	Recall	100%	100%	100%	100%	71.4%	71.4%	71.4%	57.1%
	DD	(12, 286, 158)	(12, 340, 167)	(13, 2072, 415)	(13, 2110, 425)	(12, 4894, 955)	(12, 1825, 399)	(13, 5274, 893)	(13, 588, 254)
IDV-3 & IDV-14	Precision	100%	100%	100%	100%	20%	25%	25%	33.3%
	Recall	100%	100%	100%	100%	33.3%	33.3%	33.3%	33.3%
	DD	(40, 2774, 954)	(50, 2782, 962)	(70, 2783, 975)	(87, 2918, 1229)	(40, 5294, 1309)	(50, 1162, 596)	(70, 1184, 612)	(87, 1123, 631)
IDV-26	Precision	100%	100%	100%	100%	16.7%	16.7%	20%	20%
	Recall	100%	100%	100%	100%	33.3%	33.3%	33.3%	33.3%
	DD	(400, 2243, 1083)	(442, 2254, 1134)	(442, 2263, 1275)	(443, 2813, 1462)	(400, 5272, 1681)	(442, 5282, 2566)	(442, 5287, 2040)	(443, 5298, 2049)

Table 5: Digital Twin Anomaly Detection Detail

Disturbance		Discrepancy Detector Threshold, μ_{v_i}			
		1.2	1.5	2.0	3.0
IDV-6	Precision	100%	100%	100%	100%
	Recall	100%	100%	100%	85.7%
	DD	(10, 216, 94)	(10, 216, 95)	(11, 217, 100)	(12, 217, 94)
IDV-13	Precision	100%	100%	100%	100%
	Recall	100%	100%	100%	100%
	DD	(11, 228, 117)	(12, 229, 122)	(13, 229, 122)	(14, 231, 125)
IDV-3 & IDV-14	Precision	100%	100%	100%	100%
	Recall	100%	100%	100%	100%
	DD	(114, 4130, 1460)	(168, 4104, 1607)	(169, 4105, 1637)	(170, 4107, 1639)
IDV-26	Precision	100%	100%	100%	100%
	Recall	100%	100%	100%	100%
	DD	(3919, 3919, 3919)	(3919, 3919, 3919)	(3920, 3920, 3920)	(3922, 3922, 3922)

Table 6: Digital Twin Anomaly Detection Summary

Disturbance	Decision Maker	DD for different thresholds, μ_{v_i} & c			
		1.2	1.5	2	3
IDV-6	Digital Twin	10	10	11	13
IDV-13	Digital Twin	11	12	13	14
IDV-3 & IDV-14	Digital Twin	114	168	169	170
IDV-26	Digital Twin	3919	3919	3920	3922

discrepancy detector threshold constant c does not affect the detector's performance. Table 6 shows that the decision maker is able to make correct estimation for all tested missing mode anomalies, while the change of detection delay time is not obvious with the increasing of threshold constant c.

Table 7: Anomaly classifier result for combinational anomalies

	Anomaly Classifier Threshold, c			
	1.1	1.2	1.3	1.5
Precision	100%	100%	100%	100%
Recall	66.7%	66.7%	66.7%	56%
DD	(0, 3442, 512)	(0, 3460, 515)	(0, 3483, 519)	(0, 1118, 104)

5.6 Experiment 4: Anomalies Due to Multiple Sources

This experiment was designed to evaluate the performance of our detection mechanism when both the faulty sensors and the missing modes in the digital twin induce anomalies. For sensor measurement anomaly, three types of sensor measurement fault were used: drift fault, erratic fault, and offset fault, and those sensor measurement faults were applied to 6 variables which were the same as Section 5.3 (Experiment 1). For digital twin anomaly, two disturbances, IDV-13 and IDV-3 & IDV-14, were introduced into the mode 1 noise-free operation data to emulate the digital twin mode missing anomalies. In each test, only one type of sensor measurement fault was used together with one digital twin missing mode.

The experiment results are shown in Table 7, Figure 7 and 8. Table 7 includes the summary of anomaly classifier performance on sensor measurement anomalies, Figure 7 shows the details of the anomaly classifier performance, and Table 8 shows the discrepancy detector performance on combinational anomalies data. According

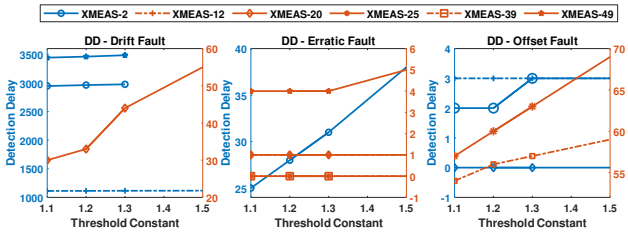

Figure 7: Anomaly classifier result details for combinational anomalies

Table 8: Discrepancy detector result for combinational anomalies

		Discrepancy Detector Threshold, μ_{0_i}			
		1.2	1.5	2.0	3.0
IDV-13	Precision	100%	100%	100%	100%
	Recall	100%	100%	100%	100%
	DD	(7, 377, 196)	(7, 379, 197))	(8, 414, 203)	(9, 416, 212)
IDV-3 & IDV-14	Precision	100%	100%	100%	100%
	Recall	100%	100%	100%	100%
	DD	(114, 4103, 1461)	(168, 4820, 1607)	(169, 4105, 1636)	(170, 4107, 1637)

to Table 7, no normal data have been classified as anomalies by the anomaly classifier, while 1/3 of the real abnormal data are missed by the anomaly classifier during the online monitoring, which demonstrates that the anomaly classifier is less sensitive to sensor measurement faults. Figure 7 shows that for some of the variables, small threshold constant range adjustment did not influence the detection delay time significantly. This means the probability of the corresponding variables change rapidly when the anomalies happen. The result of Table 8 shows the high sensitiveness and accuracy of discrepancy detector on digital twin mode missing anomalies. The discrepancy detector does not miss any true abnormal data and classify normal data as anomaly. Combining the results of Table 7 and Table 8, the correctness of the decision maker remains in 66.7%.

6 CONCLUSION

This paper introduced an anomaly detection framework for digital twin based CPS. The framework consists of the Gaussian Mixture Model based discrepancy detector, which initially checks if there is an anomaly from two data sources one from the plant and the other from the digital twin. Then the anomaly classifier, which employs Hidden Markov Model, further classifies the types of anomalies based on the signatures generated from the discrepancy detector. The experiments showed the application of our framework using the Tennessee Eastman process model that correctly detected and identified different types of anomalies. In future work, we would like to integrate correction mechanisms to our framework that can keep the system in the stable state depending on the results of the classification.

ACKNOWLEDGMENTS

This work was supported by Delta-NTU Corporate Lab for Cyber-Physical Systems with funding support from Delta Electronics Inc. and the National Research Foundation (NRF) Singapore under the Corp Lab@University Scheme.

REFERENCES

[1] Cesare Alippi, Stavros Ntalampiras, and Manuel Roveri. 2016. Model-Free Fault Detection and Isolation in Large-Scale Cyber-Physical Systems. *IEEE Transactions on Emerging Topics in Computational Intelligence* 1, 1 (2016), 61–71.

[2] Efe C Balta, Dawn M Tilbury, and Kira Barton. 2019. A Digital Twin Framework for Performance Monitoring and Anomaly Detection in Fused Deposition Modeling. In *2019 IEEE 15th International Conference on Automation Science and Engineering (CASE)*. IEEE, 823–829.

[3] Pietro Barbiero, Ramon Viñas Torné, and Pietro Lió. 2020. Graph representation forecasting of patient's medical conditions: towards a digital twin. *arXiv preprint arXiv:2009.08299* (2020).

[4] Andreas Bathelt, N Lawrence Ricker, and Mohieddine Jelali. 2015. Revision of the tennessee eastman process model. *IFAC-PapersOnLine* 48, 8 (2015), 309–314.

[5] Torsten Blochwitz, Martin Otter, Martin Arnold, Constanze Bausch, Christoph Clauss, Hilding Elmqvist, Andreas Junghanns, Jakob Mauss, Manuel Monteiro, Thomas Neidhold, et al. 2011. The Functional Mockup Interface for Tool Independent Exchange of Simulation Models. In *Proceedings of the 8th International Modelica Conference*. Linköping University Press, 105–114.

[6] Andreas Bunte, Benno Stein, and Oliver Niggemann. [n.d.]. Model-based diagnosis for cyber-physical production systems based on machine learning and residual-based diagnosis models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 2727–2735.

[7] Tao Chen and Jie Zhang. 2010. On-line multivariate statistical monitoring of batch processes using Gaussian mixture model. *Computers & chemical engineering* 34, 4 (2010), 500–507.

[8] Sujit Rokka Chhetri, Sina Faezi, Arquimedes Canedo, and Mohammad Abdullah Al Faruque. 2019. QUILT: Quality Inference from Living Digital Twins in IoT-Enabled Manufacturing Systems. In *Proceedings of the International Conference on Internet of Things Design and Implementation*. 237–248.

[9] Arthur P Dempster, Nan M Laird, and Donald B Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)* 39, 1 (1977), 1–22.

[10] Warren E Dixon, Ian D Walker, Darren M Dawson, and John P Hartranft. 2000. Fault detection for robot manipulators with parametric uncertainty: A prediction-error-based approach. *IEEE Transactions on Robotics and Automation* 16, 6 (2000), 689–699.

[11] James J Downs and Ernest F Vogel. 1993. A plant-wide industrial process control problem. *Computers & chemical engineering* 17, 3 (1993), 245–255.

[12] Christopher Edwards and Chee Pin Tan. 2006. Sensor fault tolerant control using sliding mode observers. *Control Engineering Practice* 14, 8 (2006), 897–908.

[13] Ilenia Epifani, Carlo Ghezzi, Raffaela Mirandola, and Giordano Tamburrelli. 2009. Model evolution by run-time parameter adaptation. In *2009 IEEE 31st International Conference on Software Engineering*. IEEE, 111–121.

[14] Hamza Fawzi, Paulo Tabuada, and Suhas Diggavi. 2014. Secure estimation and control for cyber-physical systems under adversarial attacks. *IEEE Transactions on Automatic control* 59, 6 (2014), 1454–1467.

[15] Valentina Fedorova, Alex Gammerman, Ilia Nouretdinov, and Vladimir Vovk. 2012. Plug-in martingales for testing exchangeability on-line. *arXiv preprint arXiv:1204.3251* (2012).

[16] Sana Ullah Jan, Young-Doo Lee, Jungpil Shin, and Insoo Koo. 2017. Sensor fault classification based on support vector machine and statistical time-domain features. *IEEE Access* 5 (2017), 8682–8690.

[17] Enzo Losi, Mauro Venturini, Lucrezia Manservigi, Giuseppe Fabio Ceschini, and Giovanni Bechini. 2019. Anomaly Detection in Gas Turbine Time Series by Means of Bayesian Hierarchical Models. *Journal of Engineering for Gas Turbines and Power* 141, 11 (2019).

[18] Qiuchen Lu, Xiang Xie, Ajith Kumar Parlikad, and Jennifer Mary Schooling. 2020. Digital twin-enabled anomaly detection for built asset monitoring in operation and maintenance. *Automation in Construction* 118 (2020), 103277.

[19] MarketsandMarkets. [n.d.]. *Digital Twin Market by Technology, Type, Application, Industry, and Geography - Global Forecast to 2026.* https://www.marketsandmarkets.com/Market-Reports/digital-twin-market-225269522.html

[20] Hae-Sang Park and Chi-Hyuck Jun. 2009. A simple and fast algorithm for K-medoids clustering. *Expert systems with applications* 36, 2 (2009), 3336–3341.

[21] Leonard KAUFMAN Peter J RDUSSEEUN. 1987. Clustering by means of medoids. (1987).

[22] Khaoula Tidriri, Nizar Chatti, Sylvain Verron, and Teodor Tiplica. 2016. Bridging data-driven and model-based approaches for process fault diagnosis and health monitoring: A review of researches and future challenges. *Annual Reviews in Control* 42 (2016), 63–81.

[23] Shih-Yuan Yu, Arnav Vaibhav Malawade, Sujit Rokka Chhetri, and Mohammad Abdullah Al Faruque. 2020. Sabotage attack detection for additive manufacturing systems. *IEEE Access* 8 (2020), 27218–27231.

[24] Valentina Zaccaria, Mikael Stenfelt, Ioanna Aslanidou, and Konstantinos G Kyprianidis. 2018. Fleet monitoring and diagnostics framework based on digital twin of aero-engines. In *Turbo Expo: Power for Land, Sea, and Air*, Vol. 51128. American Society of Mechanical Engineers, V006T05A021.

A OPTIMAL CLUSTER NUMBER SELECTION STRATEGY

The overall strategy for determining the optimal number of clusters is to find the number of clusters with smallest reconstruction error [1] by iterating all possible numbers of clusters. The whole collected fault-free data set is divided into two parts: the training data set D_t and the validation data set D_v, which are defined as follows:

$$D_t = \begin{bmatrix} y_1^1 & \cdots & y_\mu^1 \\ \vdots & \ddots & \vdots \\ y_1^N & \cdots & y_\mu^N \end{bmatrix}, D_v = \begin{bmatrix} y_{\mu+1}^1 & \cdots & y_T^1 \\ \vdots & \ddots & \vdots \\ y_{\mu+1}^N & \cdots & y_T^N \end{bmatrix}$$

where D_t contains N variables and μ data samples, and D_v contains N variables and $T - \mu$ data samples. The minimum number of clusters is 1, and for avoiding overspreading of the variables, the maximum number of clusters, k_{max}, should not be larger than $\frac{1}{2}N$. The iteration of number of clusters starts from 1 to k_{max}.

For every tested number of clusters, k, the k-medoids algorithm is applied on D_t, and the steps for obtaining the reconstruction errors e_{total}^k are specified as below:

(1) After applying the k-medoids algorithm for given cluster number k on D_t, the clusters $C_1, C_2, ..., C_k$ are generated.

(2) Since the variables in the same cluster are highly correlated, for each cluster C_j (with $j = 1, ..., k$), $|C_j|$ Multiple-Input Single-Output (MISO) predictive models ($|\cdot|$ is the cardinality operator) are created. Assuming the variable i is in cluster C_j (with $i = 1, ..., |C_j|$), the MISO predictive model for variable i sets variable i as single output and the remaining $|C_j| - 1$ variables as inputs, which can be represented as:

$$\begin{aligned} \widehat{y}_t^{i,j} = & g^{i,j}(y_{t-1}^{i,j}, y_{t-2}^{i,j}, ..., y_{t-\tau}^{i,j}, \\ & y_t^{1,j}, y_{t-1}^{1,j}, y_{t-2}^{1,j}, ..., y_{t-\tau}^{1,j}, ..., \\ & y_t^{|C_j|,j}, y_{t-1}^{|C_j|,j}, y_{t-2}^{|C_j|,j}, ..., y_{t-\tau}^{|C_j|,j}, \Theta) \end{aligned}$$

where the τ and Θ are suitably estimated based on the value of N [1]. Among all MISO predictive models, the AutoRegressive with eXogenous input (ARX) model is used. The ARX model for variable i in cluster C_j is denoted as $M_\Theta^{i,j}$, and N ARX models will be trained based on D_t in total.

(3) For every ARX model $M_\Theta^{i,j}$ (with $i = 1, ..., |C_j|$ and $j = 1, ..., k$), the predictive vector Y_i for variable i is calculated with input D_v on model $M_\Theta^{i,j}$, where $Y^i = [\hat{y}_{\mu+1}^i, \cdots, \hat{y}_T^i]$. Denote the actual value of variable i in D_v as D_v^i, where $D_v^i = [y_{\mu+1}^i, \cdots, y_T^i]$, the reconstruction error $e_{i,j}$ for variable i on model $M_\Theta^{i,j}$ is defined as follows:

$$e_{i,j} = \sum_{t=\tau+1}^{T} abs(y_t^i - \hat{y}_t^i)$$

(4) The overall reconstruction error $e_{overall}^k$ is define as

$$e_{overall}^k = \frac{1}{k} \sum_{j=1}^{k} \frac{1}{|C_j|} \sum_{i=1}^{|C_j|} e_{i,j}.$$

The value of the overall reconstruction error evaluates the ability of the ARX models trained on D_t upon capturing the relationships existing each cluster. After all iterations, choosing the k with least overall reconstruction error as the optimal number of clusters.

B TENNESSEE EASTMAN PROCESS MODEL

The Tennessee Eastman Process (TEP) model is originated from an actual industrial process of Eastman Chemical Company. Originally introduced in [11], the model was initially coded with FORTRAN that describes the non-linear relationships in the unit operations and the material and energy balances. The model was then revised in [4] using MATLAB/Simulink that uses a variable-step integrator. In this section, the revised model is used for evaluating the performance of our detection technique for various types of anomalies.

As shown in Figure 5, the TEP model consists of five major unit operations: reactor, product condenser, vapour-liquid separator, recycle compressor and product stripper. The chemical process involves four reactants that produces two products and two byproducts. The model consists of 73 measurable outputs and can be configured to introduce 28 different disturbances that emulate faults. There are two modes of process operation for the revised models: *base mode* and *alternative mode*, which are called mode 1 and mode 3 in [11]. Furthermore, the model provides a set of parameters for activating or deactivating the measurement noises.