Federated Learning-Based Explainable Anomaly Detection for Industrial Control Systems

Truong Thu Huong, Ta Phuong Bac, Kieu Ngan Ha, Nguyen Viet Hoang, Nguyen Xuan Hoang, Nguyen Tai Hung, Kim Phuc Tran

Federated learning-based Explainable Anomaly Detection for Industrial Control Systems

Truong Thu Huong[a,], Ta Phuong Bac[b], Kieu Ngan Ha[a], Nguyen Viet Hoang[a], Nguyen Xuan Hoang[a], Nguyen Tai Hung[a,], Kim Phuc Tran[c]

[a] *Hanoi University of Science and Technology, Vietnam*
[b] *School of Electronic Engineering, Soongsil University, Seoul, Korea*
[c] *Universit de Lille, ENSAIT, GEMTEX, F-59000 Lille, France*

Abstract

We are now witnessing the rapid growth of advanced technologies and their application, leading to Smart Manufacturing. The Internet of Things (IoT) is one of the main technologies used to enable smart factories, which is connecting all the industrial assets, including machines and control systems, with the information systems and the business processes. Industrial Control Systems of smart IoT-based factories are one of the top industries attacked by numerous threats. With the distributed structure of plenty of IoT front-end sensing devices in Smart Manufacturing, an effectively distributed architecture of an anomaly detection system should be created that can produce high detection performance while being able to handle the cybersecurity task in fast time scale. In this paper, we propose the so-called FedeX architecture that integrates Federated Learning into the detection learning model to aggregate various distributed learning models into one global model update for high detection performance in comparison with a variety of recent research proposed solutions Bo sung them vao day la ta dat duoc detection rate cu the the nao, hoac so sanh thi hon giai phap khac bao nhieu. FedeX is also robust in terms of running time and hardware consumption which allows us to deploy the detecting task on top of edge computing infrastructure in real-time. FedeX also uses eXplainable Artificial Intelligence to deal with the problem of "black-box" anomaly detection for Industrial Control Systems.

Keywords: Anomaly Detection, ICS, Federated Learning, XAI, VAE,

Email addresses: huong.truongthu@hust.edu.vn (Truong Thu Huong), hung.nguyentai@hust.edu.vn (Nguyen Tai Hung)

1. Introduction

AI and Bigdata present excellent potential in migrating the manufacturing paradigm to smart manufacturing, as it enables AI-driven IIoT systems to operate in real-time and be more precise and efficient [1].

Within the context of smart manufacturing, Industrial Control Systems (ICS) are an essential component of industrial systems, and their safety and security are becoming increasingly important in the Industrial Internet of Things (IIoT) landscape. However, the exponential rise of IIoT brings not only enormous benefits but also significant obstacles in terms of developing and deploying secured ICSs [2], [3]. In reality, a contemporary ICS is no longer a stand-alone system but rather linked to the Internet. As a result, if hackers were to acquire control of a network and steal security-critical data, or viruses and infections would infiltrate and damage a production line's operating system, the effects would be severe and costly. Industrial Control Systems based on the IIoT is currently one of the top industries attacked by various threats. As threats are becoming more complex, an anomaly detection method that can identify attacks quickly and correctly while being lightweight enough to be used in Internet of Things (IoT) devices with limited processing capacity in industrial environments is required.

From another perspective, Federated Learning (FL) - a distributed machine learning mechanism [4] is a promising candidate for communication costs in a distributed environment. Therefore, in this paper, we develop a FL-based anomaly detection for ICSs right at edge sites. This way helps aggregating distributed local learning models for a global model update, thereby giving each single local model knowledge of other data patterns at other edge zones. Therefore, Federated Learning can achieve efficiency similar to the centralized learning manner while distributing the detection task to different local edge zones, thereby not offloading the central cloud. While in another side, the benefit of deploying anomaly detection tasks at the edge with FL is that it definitely improves the system response time upon attack arrivals since the detection model is executed right at the edge which is near to attack/anomaly sources. The integration of the FL technique with an ML-based detection scheme helps to achieve the advantages produced by both techniques in an efficient and lightweight way.

Besides, although deploying FL enables distributed deep learning algorithms to work efficiently for anomaly detection in IIoT-based ICSs, anomaly detection techniques can only help detect abnormalities. The output of the Machine Learning-based detection model is difficult to explain or interpret, especially in ICSs where information is often abstract. Interpretability is the degree to which a human can understand the cause of a decision [5]. An explanation denotes the subset of elements in a sample that has the highest impact on predicting a label output of an ML-based detection model. Note that this is a very machine-learning-centric definition. In the domain of cybersecurity analysts, a satisfying explanation would also need a description of why those attributes are critical. Because of this limitation, persuading experts to accept and use anomaly detection technologies is difficult. Such ML-based model' outputs may contain abnormal cases that the systems analyst was previously unaware of, and an explanation of why an instance is abnormal might boost the analyst's confidence in the algorithm. The higher the interpretability of an ML model is, the more easily administrators can comprehend why certain predictions have been made. Furthermore, explanations might be contradictory, which is valuable and important for explaining anomalies. To overcome this drawback, the concept of eXplainable Artificial Intelligence (XAI) came into play for ICSs. XAI has been developed to explain predictions from anomaly detection algorithms.

Motivated by these potentials, in this paper, we propose a **Fed**erated learning-based **Ex**plainable Anomaly Detection for Industrial Control Systems - called FedeX as the whole architecture to detect and analyze anomalies in ICSs and to enable detection in a distributed environment with FL. In FedeX, we propose to:

- Distribute the anomaly detection task to the edge. Distributed edge computing approach brings a few benefits: detection response can be faster since the detection task is carried out right at the edge which is near attack sources. In case of attacks, closing an edge zone to seal attacks within does not affect the other zones' operation. Practitioners also can more easily allocate the source of attacks within each small monitored area with a limited number of connecting devices.

- Solve the detection problem by using the so-called FedVAE-SVDD model to guarantee high detection performance and real-time operation (i.e. minute-time scale). FedVAE-SVDD combines the advantage of Variational Autoencoder (VAE) [6] and Support Vector Data

Description (SVDD) [7] implemented at the Edge. While most current approaches determine outlier thresholds by either using heuristic methods or normally-distributed data assumptions that are unrealistic and increase false-positive rates. Therefore, FedVAE-SVDD, by using SVDD to seek for an optimal threshold, can deliver an automatic anomaly detection solution with high performance that is proved to outperform some other learning models such as Logistic Regression (LR), Linear Discriminant Analysis (LDA), K-nearest Neighbours (KNN), Nave Bayes (NB), Support Vector Machine (SVM), and Classification and Regression Tree (CART) in the same ICS context [8].

- Leverage FL to reduce huge work offloading in the cloud. Also, exchanging only model information from the edge devices to the cloud also solves the problem of data shortage in each device, especially with high-dimensional datasets. In addition, by using FL, each single detection model based on each local training dataset at each zone will be updated globally.

- Integrate XAI (i.e SHAP [9]) to deal with the current "black-box" detection approaches. SHAP can thoroughly explain anomalies detected by the FedVAE-SVDD model. This type of XAI aims to provide a comprehensive explanation and remove drawbacks in understanding the output of the FedVAE-SVDD anomaly detection. This XAI method can evaluate the model's faithfulness as well as provide the ability to detect a specific component causing the system problem. SHAP is the most exact agnostic XAI method available today, as a cause-of-interpretation tool. SHAP plots and outputs recorded values and accommodate practitioners to analyze and interpret the results, further evaluating the reliability of the proposed ML-based training model.

In summary of FedeX's benefits, FedeX is the framework that applies XAI to explain anomaly for ICSs in a liquid-storage infrastructure. FedeX is also proved to have high detection performance while being able to be deployed on top of such weak hardware of edge nodes in a distributed computing architecture. The FL-based architecture of Fedex enables faster system response capability upon attacks. FL also assures that the detection model in each single distributed zone can have global knowledge by updating the global model aggregated from all distributed learning models. Moreover, with real-time training capability (i.e in minute times-scale), FedeX is able

to retrain its learning model constantly in order to cope with any drift in the normal/abnormal behavior of data coming from devices (for example, drift caused by device aging inside a smart factory).

The rest of our paper is structured as follows. Section 2 discovers related and cutting-edge researches in the field of anomaly detection for ICSs and XAI for Anomaly detection in ICSs. The FedeX anomaly detection architecture will be elaborated in detail in Section 3. The evaluation of the FedeX performance in terms of detection capability, system response time, edge computing capability, and anomalies explanation is presented in Section 4. Finally, the conclusion of our findings is presented in Section 5.

2. Related Work

For the non time series data type, we can find anomaly detection approaches for ICSs in [10] , [11], [12]. In [10], the author proposes a Logical Analysis of Data - LAD-ADS solution using a rule-based method to detect anomalous behaviors in ICS systems over the SWat dataset. LAD-ADS performs detection by extracting rules from a huge of data in the past. However, rule-based systems are often complex and challenging to manage and determine the cause of detected anomalies. In addition, it requires experience from experts and operational engineers in deploying and operating the system. In the same scenario of ICSs, [11] proposes a state-aware anomaly detection method that uses the CUSUM (Cumulative Sum) control chart to the state-dependent detection threshold. The training process is done centrally on a large amount of data. In fact, the data in ICSs are often distributed; using a centralized solution can cause disadvantages such as latency for sending all raw data to the central cloud and huge computer resource consumption for training. Moreover, in CUSUM [11], no detection performance metric such as Accuracy, Precision, Recall, or F1-score is revealed except the false alarm rate. In another aspect, training in a distributed environment and the privacy of data is addressed in [12]. In [12], the authors present a methodology called MADICS for Anomaly Detection in Industrial Control Systems using a semi-supervised anomaly detection paradigm with five main steps. The performance of MADICS in terms of Recall is slightly low over its testing dataset. In addition, this mechanism requires a large amount of data for semi-supervised learning, which faces data privacy issues when transmitting a large amount of raw data for training, resource capacity, and computational resources of the system. In terms of detection performance, Fedex is

also proved to outperform the previously proposed solutions MADICs [12], LAD-ADS [10] in the same factory contexts. Additionally, in [13], a statistical window-based anomaly detection method is adopted by using various deep-neural network architectures, showing effectiveness in detecting the attacks in a Secure Water Treatment (SWaT) infrastructure which is, in our opinion, not a purely time-series data scenario as well. However, the authors also indicated that their work needs to be improved with the interpretability of the outcomes and the behavior detection of fault ICS components.

For the time-series data type, we have observed various proposed AD solutions. Training in a distributed environment and the privacy of data is also an issue that needs to be addressed in [14]. From the aspect of using Federated Learning to implement an anomaly detection solution in a distributed ICS system, ensuring high accuracy while protecting data privacy, we can find some researches such as [15], [16]. In [15], the author proposes an FL framework that allows decentralized edge devices to cooperate in training an anomaly detector with an attention mechanism-based convolutional neural network long short-term memory (AMCNN-LSTM) model. Although designing an FL-based approach, but the experiments lack insight analysis in the performance of deploying such a learning model in an edge environment (i.e in weak hardware of an edge node). Due to the complexity of AMCNN-LSTM caused by using multi-layer CNN and LSTM, according to our experience, it is hard to feasibly deploy such a learning model on edge devices, much less for an expectation of achieving low computing complexity for running the learning model in minute-time scale and low power consumption. With a similar lack of performance testing on the edge hardware, work [16] proposes an FL-based anomaly detection approach for IoT networks based on the combination between Gated Recurrent Units (GRUs) and Long short term memory toward detecting anomalies with decentralized on-device data. However, the performance of the proposed method is not good enough in the distributed scenario; accuracy in each FL client is just around 90% on average.

In several other studies [17],[18], the authors develop and investigate attack detection solutions in ICS cyberspaces. In [18], the authors propose an attack detection model that uses a Deep Neural Network and a Decision Tree classifier to identify cyber-attacks in the ICS context with an F1-Score of 93.83% with the ICS gas pipeline dataset, which is higher than other algorithms such as Support Vector Machine (SVM), Long Short-term Memory (LSTM), Nave Bayes (NB), Decision Tree (DT), Deep Neural Network

(DNN), Random Forest (RF). Study [17] uses the semi-supervised techniques by leveraging K-means and Convolutional Autoencoder to protect the ICS system from cyberattack. Like in [18] , the experiments of the proposed methods were performed with the gas pipeline dataset and the water storage tank dataset. However, the anomaly detection performance of the proposed method needs to be improved. In contrast to our study, these studies only focused on evaluating the performance of detection algorithms and did not consider other important metrics when being implemented in the edge environments of an ICS such as detection time and power consumption. Thus, in this paper, we present a comprehensive evaluation of both detection performance and system performance.

Although the studies described above solve challenges surrounding cyberattack detection in ICSs, all of them have not concerned the interpretability of the models detected results up to now. As stated in [19], the interpretability of an anomaly detection model is almost as crucial as the prediction accuracy of the model. In the field of explaining the detection outcomes (XAI - Explainable AI), Kasun et al. in [19] used a method named Layer-wise Relevance Propagation (LRP) to calculate the input features relevance to explain the trained Deep Neural Network model with DoS attacks detection task. The evaluation is conducted with a subset of NSL-KDD Dataset an old network intrusion detection dataset released in 1999. Even though the combination of solutions to solve the black-box problem of DNN helps domain experts intuitively access the insight of the DNN algorithms, classification accuracy improvement is required when producing predictions in the test set. Very recently, the authors in [20] have proposed to use XAI to interprete anomaly detection outcomes of the multiple Bi-LSTM learning model in an ICS ecosystem. The scope of the ICS is the smart factory of steam-turbine power generation and pumped-storage hydropower generation. This paper can be considered as the forefront of interpreting anomaly detection in the ICS ecosystem.

As a result, in this paper, we design and investigate the FedeX solution for not just ensuring high detection performance and lightweight implementation at the edge devices, but also providing a detailed explanation for the detection model deployed in a liquid storage infrastructure.

3. Federated learning-based Explainable Anomaly Detection for ICSs - FedeX

3.1. FedeX Overview

In this paper, an architecture using FL for anomaly detection is proposed for ICSs. As Fig.1 shows, ICSs in smart factories can be organized in various zones (i.e. Zone 1, Zone 2, Zone 3...), and each of which is monitored by a local unit (i.e. Edge 1, Edge 2...) to detect anomalies. Those local monitoring units serve as edge computing stations that run an anomaly detection function based on their own incoming local data. Computing can be carried out at the distributed edges as long as detection algorithms running on top of it require a reasonable computing capacity. If this requirement is fulfilled, then this architecture becomes effective since the detection module is implemented near attack sources which makes the whole detection process respond faster. Moreover, this solution reduces the workload offloading up on the central cloud server as the traditional centralized computing architecture does.

As illustrated in Fig. 1, the FedeX workflow consists of 6 steps, as follows: Step ①: The edge device uses the sensing data collected from nodes within a zone as a local dataset. Step ②: The edge device performs the local model (i.e., VAE model) and the mechanism for determining thresholds at the last communication round (i.e., FedVAE-SVDD model) training on the local dataset. Step ③: The edge device uploads the weight matrix to the cloud aggregator. Step ④: The cloud aggregator obtains a new global model by aggregating the weights uploaded by the edge device. Step ⑤: The cloud aggregator sends the new global model to each edge device. The steps above are repeated until the global model achieves optimal convergence. This ideal global model can be used by decentralized devices to conduct anomaly detection tasks. Step ⑥: Periodically, the XAI-SHAP model will be run to interpret and verify the anomaly detection model; and identify the anomaly-causing elements in ICSs.

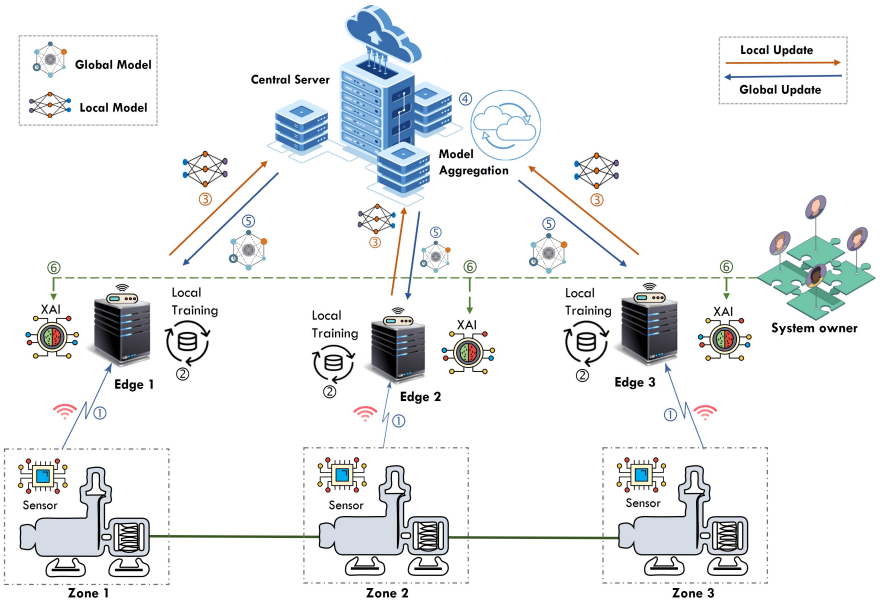


Figure 1: Our FedeX Model in ICS

For this type of specific smart factory, we propose our architecture FedeX, standing for **Fed**erated Learning-based **Ex**plainable Anomaly Detection. The main contributions are as follows:

1. *Anomaly Detection Model on Edges*:
VAE [6] is a tuned Autoencoder architecture to run on top of the edge device for effective anomaly detection. The benefit of VAE is the ability to minimize over-fitting by ensuring that features from its latent space are good enough for data generation. A basic idea here is, if a model was only trained on normal data, then when encountered with anomalous data, the inability to reconstruct data or, more precisely, the range of the reconstruction error that it entails, can signal the presence of anomalous data.

2. *Distributed Learning Mechanism for Efficient Resource Sharing*:
Moreover, this architecture expands to an FL-based VAE (i.e. FedVAE) anomaly detection solution that can solve the problem of missing training data at each edge device when deep learning models often need large amounts of data to train. With the FL technique, the central cloud can federate information with different characteristics from various zones to improve the detection performance of the overall network without the need of having knowledge of original raw data.

3. *Accurate, Fast and Automatic Determination Mechanism of Threshold for Anomaly Detection*:
In addition, the SVDD method [21] is proposed to automatically determine the threshold for efficient anomalies detection (FedVAE-SVDD). Every sample at the output of the VAEs block with the loss larger than the chosen threshold will be considered anomalous.

4. *Explainable Artificial Intelligent to Interpret the Outputs of Black-Box Learning Models*:
Finally, a XAI-based explanation method called SHAP is integrated into our architecture. SHAP enables us to explain the reason why an instance is predicted as an anomaly by showing the contribution of the features to the prediction, thereby enhancing the reliability of the black-box model. As a result, FedeX effectively assists domain engineers in finding the physical cause of anomalies quickly and making responses timely.

In the following sections, we will present a detailed design and deployment of the VAEs, FL, and SVDD as the hybrid-anomaly-detection model at the edge. And finally we will elaborate how to deploy Explainable-Artificial-Intelligent (XAI) for better understanding of anomalies occurred in the ICS of a liquid storage infrastructure.

3.2. Design and Development of Variational AutoEncoder (VAE) as Local Training Model on Edge

Since the detection module is implemented on Edge hardware, the overall design is supposed to be lightweight, whilst still ensuring the detection accuracy requirement. Therefore, in our proposed detection method, we try our best to reduce the computing complexity of the algorithms. As elaborated in Figure. 1, the AI-based detection learning model each is deployed locally at each single Edge. Then the model update of each local edge will be done through a global updating, facilitated by Federated Learning. cau nay la de giai thich cho edge computing tai comment 1.3, can bo sung them neu can..
In this research, we propose to utilize VAE for anomaly detection purposes. In fact, many different VAE architectures have been proposed, with different types of layers such as Dense, LSTM, and CNN. We design the VAE encoder and decoder with only two fully connected layers each because this approach aims to achieve the model's simplicity and lightweight, allowing it to be trained on top of edge devices with limited hardware resources, as part of an IoT-based ICS, while also lowering communication costs and providing

sufficient detection performance. We also emphasize the real-time training guarantee for this model which will be illustrated in Section 4.

For the background, an autoencoder (AE) is a symmetrical unsupervised neural network, slightly different from other network architectures in that: The network uses the input itself as the ground truth. It consists of 3 main parts: encoder, latent representation, and decoder. Usually, the centre hidden layer has fewer nodes than the input and output layer (a "bottleneck"). Thus, the network learns to compress the input to the bottleneck layer and then from which subsequently restore the input. This middle layer thus becomes the "latent representation" of the input, retaining most information about the input using fewer features. The part of the network before this layer becomes the encoder, and the part after becomes the decoder.

A variational autoencoder (VAE) [6] is a combination of the AE with the Variational Bayesian method. But instead of generating a representation in the hidden space for a data point in the original space, the underlying principle behind VAE is to find a probability distribution for that data point.

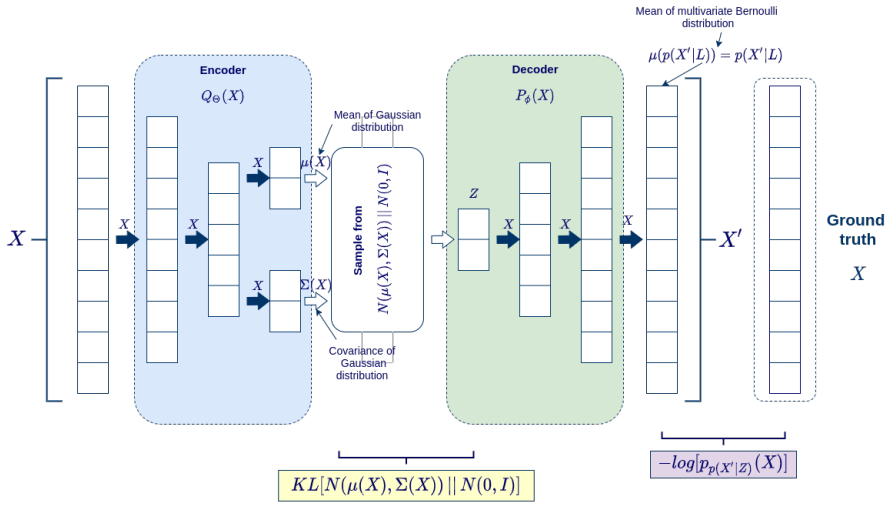


Figure 2: VAE Structure: the data are represented via a normal distribution, and the data dimensions are independent random variables.

Let X be data generated by inputting a latent variable Z through a random process with network parameters θ. The goal is to model the data as a distribution, $Q_\theta(X)$. Since the computation cost to calculate $Q_\theta(X)$ is expensive, we translate the problem into the autoencoder domain by defining the

"probabilistic encoder" $P_\phi(Z|X)$ to approximate the posterior distribution $Q_\theta(Z|X)$.

The VAE module is designed to combine with FL as follows:

- In each iteration of the training, one option is to let the participating nodes synchronize their VAE models over the parameter server. This, however, necessitates many rounds of communication between the federated nodes and the parameter server, possibly resulting in network communication congestion.

- Instead, we let the participating nodes run a number of local modifications and occasionally synchronize with the parameter server. Specifically, after the nodes receive an updated model from the server, they update it locally by running ρ iterations of the clients' optimizer algorithm, and then send the appropriate data to the server for updating the aggregated model. Finding the optimal choice of ρ for lowering the overall communication cost of the process is a critical trade-off problem that we have to solve.

To update the network weights by backpropagation, as with any deep learning problem, a differentiable loss function must be defined. For VAEs, the objective is to minimize the generative model parameters θ to reduce the reconstruction error between the network's input and output, and also ϕ, to have $P_\phi(Z|X)$ as close to $Q_\theta(Z|X)$ as possible.

For calculating the distance between two distributions, the commonly used function is the Kullback-Leibler divergence function. The loss function called evidence lower bound (ELBO), is obtained:

$$-\mathcal{L}_{(\theta,\phi)} = log(Q_\theta(X)) - D_{KL}(P_\phi(Z|X) \,||\, Q_\theta(Z|X)) \leq log(Q_\theta(X)) \qquad (1)$$

The parameters of the model can thus be expressed as:

$$(\theta^*, \phi^*) = argmax_{\theta,\phi}\mathcal{L}_{(\theta,\phi)}(x) \qquad (2)$$

As the idea mentioned above, we will sample in the hidden space and feed to the encoder to generate new data. There should be a certain rule for this sampling. A simple yet efficient way is suggested to use the normal distribution $N(0, 1)$. That is, we add a constraint that each data point

will be represented by a normal distribution that approximates the $N(0,1)$ distribution. This is the idea of reparameterization [6]. To approximate two distributions, again the KL-Divergence loss of two Gaussian distributions with univariate function is shown below:

$$D_{KL}[N(\mu(X),\Sigma(X))\|N(0,1)] = \frac{1}{2}\sum_{k}\big(\exp(\Sigma(X)) + \mu^2(X) - 1 - \Sigma(X)\big) \tag{3}$$

with k is the dimension of our Gaussian.

Algorithm 1: Phase-1: FedVAE

Input: Initial model w_0, Client optimizer *Opt*
Output: $VAEcomplete$ - Trained VAEs model in each client

N - the number of zones;
Rounds - number of communication rounds;
for $r = 1$ **to** $Rounds - 1$ **do**
    Server randomly picks C zones;
    Server sends ω_r to C zones ;
    for *node* $c \in C$ **do**
        $\omega_{r,0} \leftarrow \omega_r$;
        for $t = 0$ **to** $\rho - 1$ **do**
            Compute stochastic gradient
            $\tilde{\nabla} f(X,\omega) = \nabla \mathcal{L}_{\theta,\phi}(X)$;
            set $\omega_{r,t+1}^{(c)} \leftarrow$ Opt $(\tilde{\nabla} f_c(\omega_{r,t}^{(c)}), \omega_{r,t}^{(c)}, \alpha, t)$;
        end for
        send $\omega_{r,\rho}^{(c)}$ to the server;
    end for
    server finds $\omega_{r+1} \leftarrow \frac{1}{C}\sum_{c\in C}\omega_{r,\rho}^{(c)}$;
end for
$VAEcomplete = f(\omega_{Rounds})$;
return $VAEcomplete$

3.3. Design and Development of Federated-Learning based VAE - FedVAE

In our Federated-Learning based VAE model (or called FedVAE), each edge device performs the training and detection process with local data from

each manufacturing area, and the Edge device only sends information of the weight matrix of the trained model to the cloud server, rather than the entire raw data, as a traditional cloud-based training system would. Although the cloud has the storage and computing power to manage the volume of data generated in manufacturing, the computationally intensive operations and vast data storage that are hosted on cloud servers may cause a delay. Because this delay is caused by the time required to send, transfer, and process massive amounts of data from IoT devices at production sites. This is a significant issue in a smart factory that must undertake huge monitoring and detection in real-time. Within this context, the concept of Edge-Cloud Computing combined with FL shall arise to circumvent this constraint.

- Firstly, the initial model is created by the Cloud Server as a Weight "Federator".
- The VAE model was then applied to solve anomaly detection. It then subscribes to numerous MQTT topics to which the zones will send the weights of their models.
- After the first model's weights are published to the aggregated model topics, the Cloud Server awaits requests from the VAE model configuration from each zone.
- Local models are trained at each edge based on their own dataset.
- In each communication round, the weights of the trained models $\omega_{r,\rho}^{(c)}$ are sent to the Cloud Server for FL.
- The Cloud then uses the formula (4) to calculate the weight of the federated global model:

$$\omega_{r+1} = \frac{1}{C} \sum_{c=1}^{C} \omega_{r,\rho}^{(c)} \tag{4}$$

Where:

C: the number of zones
$\omega_{r,\rho}^{(c)}$: the weight of the local model of zone c at round r
ω_{r+1}: the federated global model's weight at round $r+1$.

- Finally, the weight from the federated global model is sent downward to update the local model of each zone.

For every communication round, the server, at first, randomly picks up C zones, then sends the model $\omega_{r,\rho}^{(c)}$ to each zone. The model, which is the VAE model, will be run ρ iterations locally with the aim to minimize loss function $\mathcal{L}_{\theta,\phi}(X)$. $f(\omega)$ stands for the neural network with parameter ω. Moreover, since Adam [22] is the best among the adaptive optimizers in most of the cases, we set it for client optimizer *Opt* by default. The FedVAE model is presented in Algorithm 1.

3.4. Design and Development of FedVAE-SVDD to Determine Thresholds

As aforementioned, in Phase 2, to monitor and automatically determine the threshold for the anomaly detection model - the FedVAE model, we propose to use the Support Vector Data Description (SVDD) method to go over this request accurately while keeping the real-time assurance.

Usually, experts in the industry establish the threshold after attempting a range of values, then select the one that best balances the requirements (performance, true positive, or false negative ,.etc). SVDD also works well as an outlier detection algorithm, especially with high-dimensional datasets, but just like all SVMs, it does not scale to large datasets. Therefore, we suggest a combination of FedVAE and SVDD, as a moderate addition: FedVAE serves as the main anomaly detection model for the distributed system, while SVDD, trained with a small set of error vectors from the output of the FedVAE model, can correspond to finding a small region that encompasses all instances.

Support Vector Data Description or SVDD ([7]) is a type of support vector method used for single-class classification and outlier detection. The primary idea behind SVDD is to wrap samples in a high-dimensional space with the smallest volume. For the anomaly detection task in which most of the collected data are normal, the hypersphere is usually taken as the boundary around normal samples, separating them from outliers.

Primal Form:

Ojective Function:

$$minR^2 + C\sum_{i=1}^{n}\varsigma_i \tag{5}$$

Subject to:

$$\|x_i - I\|^2 \leq R^2 + \varsigma_i, \forall i = 1, ..., n\varsigma_i \geq 0, \forall i = 1, ..., n \tag{6}$$

Where:
$x_i \in \mathbb{R}^m, i = 1, ..., n$ indicates the training data
R: radius represents the threshold
ς_i: the slack of each variable
I: the centre
$C = \frac{1}{ne}$: the penalty constant that controls the trade-off between the volume and the errors
e: the expected outlier fraction
Dual Form:
The dual formulation is obtained using the Lagrange multipliers.
Objective Function:

$$Max \sum_{i=1}^{n} \alpha_i (x_i \cdot x_j) - \sum_{i,j=1}^{n} \alpha_i \alpha_j (x_i \cdot x_j). \tag{7}$$

$$s.t \;\; 0 \leq \alpha_i \leq C \quad and \quad \sum_{i=1}^{n} \alpha_i = 1 \tag{8}$$

Where:
$\alpha_i \in \mathbb{R}, \; i = 1, \ldots, n$ are the Lagrange coefficients
Duality Information:
The following results are valid depending on the observation position:

- Position of Centre I:

$$\sum_{i=1}^{n} \alpha_i x_i \tag{9}$$

- Inside Position:

$$\|x_i - I\| < R \rightarrow \alpha_i = 0 \tag{10}$$

- Boundary Position:

$$\|x_i - I\| = R \rightarrow 0 < \alpha_i < C \tag{11}$$

- Outside Position:

$$\|x_i - I\| > R \rightarrow \alpha_i = C \tag{12}$$

The circular data boundary can include amount of very sparse distribution of training observations space that can increase the probability of false positives. Hence, the support-vector-based boundary is usually used rather, since it is more flexible to cover the data with the volume as small as possible.

The SVDD becomes more flexible by replacing the inner product $(x_i \cdot x_j)$ with an appropriate kernel function $K(x_i, x_j)$. Results 9 through 12 hold true when the kernel function is used in the mathematical formulation. Then the threshold R is calculated using the Kernel function as follows:

$$r = \sqrt{K(x_l, x_l) - 2\sum_i (x_i, x_l) - \sum_{i,j} \alpha_i \alpha_j (x_i, x_j)} \tag{13}$$

using any $x_k \in S$ where S is the set of support vectors that have $\alpha_k < C$.

Any function that meets the Mercer condition can be used as a kernel function. Some commonly used kernels are

Gaussian kernel

$$K(x, y) = \exp \frac{-\|x - y\|^2}{2\sigma^2} \tag{14}$$

σ is the Gaussian kernel width

Exponential kernel

$$K(x, y) = \exp \frac{-\|x - y\|}{2\sigma^2} \tag{15}$$

The Gaussian kernel and the exponential kernel are very similar, with only the square of the norm left out.

Laplacian kernel

$$K(x, y) = exp(-\frac{\|x - y\|}{\sigma}) \tag{16}$$

In fact, the Laplace Kernel is completely equivalent to the exponential kernel, except for being less sensitive for changes in the σ parameter. Although both radial basis function kernels above give excellent performance, in our case study, we decide to use the Laplace kernel as default since it allows the SVDD model to run faster.

Scoring:

For each new observation Z, the distance $d(Z, I)$ is calculated as

$$d(Z,I) = \sqrt{(Z \cdot Z) - 2\sum_{i=1}^{1} \alpha_i \, \mathrm{K}\,(Z \cdot x_i) + \sum_{i,j=1}^{l} \alpha_i \alpha_j \, \mathrm{K}\,(x_i \cdot x_j)} \tag{17}$$

Thus, any dataset point with $d^2(Z,I) > r^2$ are indicated as an outlier.

More specifically in the process of Phase 2, another portion of normal data will be passed through the trained FedVAE model (at the end of Phase 1, which will return a set of loss value vectors. We use these vectors to train the SVDD model in Phase 2: The hypersphere's radius as calculated by Equation (24) will be considered as the threshold of the FedVAE-SVDD model .

Each sample belonging to the hypersphere needs to satisfy the condition:

$$(x_i - I)^T (x_i - I) \leq r^2 \tag{18}$$

Where:

$x_i \in \mathbb{R}^m, i = 1, \ldots, n$ represents the training data

r: the radius that represents the decision variable

I: the center, a decision variable

The dual formulation is obtained using the Lagrange multipliers. Also, the support-vector-based boundary is usually used rather than the hypersphere, since it is more flexible to cover the data with the volume as small as possible. Thus it can reduce the amount of very sparse distribution of training observations space that can increase the probability of false positives. Any function that meets the Mercer condition ([23]) can be used as a kernel function. Therefore, the Objective Function is obtained as follows:

$$Max \sum_{i=1}^{n} \alpha_i K(x_i \cdot x_i) - \sum_{i,j=1}^{n} \alpha_i \alpha_j K(x_i \cdot x_j). \tag{19}$$

$$s.t \;\; 0 \leq \alpha_i \leq C \quad and \quad \sum_{i=1}^{n} \alpha_i = 1 \tag{20}$$

Where:

$\alpha_i \in \mathbb{R}, \; i = 1, \ldots, n$ are the Lagrange coefficients

C: the penalty constant that controls the trade-off between the volume and the errors.

$K(.)$ stands for the kernel function. Some commonly used kernels are

Gaussian kernel

$$K\left(x,y\right)=\exp\frac{-\left\|x-y\right\|^{2}}{2\sigma^{2}} \tag{21}$$

σ is the Gaussian kernel width

Exponential kernel

$$K\left(x,y\right)=\exp\frac{-\left\|x-y\right\|}{2\sigma^{2}} \tag{22}$$

The Gaussian kernel and the exponential kernel are very similar, with only the square of the norm left out.

Laplacian kernel

$$K(x,y)=exp(-\frac{\left\|x-y\right\|}{\sigma}) \tag{23}$$

In fact, the Laplace Kernel is completely equivalent to the exponential kernel, except for being less sensitive for changes in the σ parameter. Although both radial basis function kernels above give excellent performance, in our case study, we decide to use the Laplace kernel as default since it allows the SVDD model to run faster.

The threshold r is calculated as:

$$r=\sqrt{K(x_l,x_l)-2\sum_i(x_i,x_l)-\sum_{i,j}\alpha_i\alpha_j(x_i,x_j)} \tag{24}$$

using any $x_k \in S$ where S is the set of support vectors that have $\alpha_k < C$. For each new observation Z, the distance $d(Z,I)$ is calculated as

$$d(Z,I)=\sqrt{(Z\cdot Z)-2\sum_{i=1}^{1}\alpha_i\ \mathrm{K}\left(Z\cdot x_i\right)+\sum_{i,j=1}^{l}\alpha_i\alpha_j\ \mathrm{K}\left(x_i\cdot x_j\right)} \tag{25}$$

Thus, any dataset point with $d^2(Z,I) > r^2$ are indicated as an outlier.

More specifically in the process of Phase 2, another portion of normal data will be passed through the trained FedVAE model (at the end of Phase 1, which will return a set of loss value vectors. We use these vectors to train the SVDD model in Phase 2: The hypersphere's radius as calculated by Equation (24) will be considered as the threshold of the FedVAE-SVDD model .

Algorithm 2: Phase-2: FedVAE-SVDD

Input: X - NormalTrainData;
C - penalty constant;
$K(.)$ - Kernel function;
Output: Threshold;

Recontruction Data $\tilde{X} = FedVAE(X)$;
Get set of error vectors $S_X = \{\Delta X | \Delta X = \tilde{X} - X\}$;
for $x_i, x_j \in S_X$ **do**
    if *($\alpha_i < C$ and $\alpha_j < C$)* **then**
        calculate $x' = K(x_l, x_l) - 2\sum_i (x_i, x_l) - \sum_{i,j} \alpha_i \alpha_j (x_i, x_j)$;
    end if
end for
$Threshold = \sqrt{x'}$;
return *Threshold*;

3.5. Integration of Explainable Artificial Intelligence (XAI SHAP) - FedeX

In anomaly detection, although algorithms related to neural network models tend to benefit rather than signature-based methods and techniques, its drawback is insufficient interpretability. Therefore, the reason why an instance is predicted to be abnormal can not be easily discovered in such cases, which could render researchers vague in analyzing anomaly or outlier outcomes. To overcome this limitation, a widely-used approach called Explainable Artificial Intelligence (XAI) can be adopted.

The aim of XAI is to assist humans to understand the results of solutions using black-box models by the assessment of feature attributions, thereby demonstrating how much each feature participated in making a decision for each data point of the model. With simple machine learning models, such as logistic regression and linear regression, the importance of features can be assessed via the coefficient of each feature in the data set. Meanwhile, as aforementioned, for several complicated models related to neural networks such as VAE, it is difficult to measure or compute the influence level of each feature on an output decision. This is simply because there are a large number of parameters engaging in the model. In fact, with the advent of some state-of-the-art XAI frameworks, this problem has been handled.

There are several effective XAI frameworks such as Local Interpretable Model-agnostic Explanations (LIME) [24] and Deep Learning Important Fea-

tures (DeepLIFT) [25]. However, with the scope of this study, the main XAI approach is based on SHapley Additive exPlanations (SHAP) [9] method using the Shapley values, which comes from the theory of the cooperative game [26]. By computing an individual player's contribution for each coalition $\mathcal{P}$ (a possible subset from feature set) and then averaging over all of these contributions, the Shapley value tells us the payout the one is assigned fairly in received payouts. Similarly, in terms of XAI, for an individual decision, each feature value can be considered as a player, and the payouts can be treated as the decision. Mathematically, the Shapley value of a feature in a particular prediction model can be defined as:

$$\xi_n(f, a) = \sum_{\mathcal{P} \subseteq a'} \frac{|\mathcal{P}|!(m - |\mathcal{P}| - 1)!}{m!} [f_a(\mathcal{P}) - f_a(\mathcal{P} \setminus n)] \tag{26}$$

Where, ξ_n is the Shapley value for feature n

f is a "black-box" model that needs explaining

a is an input datapoint

a' is the simplified data input

$\mathcal{P}$ is one of all possible subsets of feature set, considered as a coalition

m is the number of features in the dataset

Due to the fixed input size, commonly, the features of a model that are omitted in the Eq.(26) are substituted with random input values from the background dataset. Looking at this formula, it can be seen that the total possible subsets of an m-feature set used for interpretation is 2^m, which leads to the complexity of computing Shapley values being massive if m increase more. Therefore, to deal with this issue without calculating all combinations, Kernel SHAP [9] can be employed by sampling feature subsets and then fitting them into a linear regression model:

$$K(a) = \gamma_0 + \gamma_1 a_1 + \gamma_2 a_2 + \gamma_3 a_3 + ...\gamma_m a_m \tag{27}$$

where, a_i is a encoded feature and γ_i is the corresponding coefficient representing the contribution of the feature to the model, $i = 1, 2, ..., m$. In this linear regression model, the variables a_i are encoded according to the presence or absence of ones. Thanks to this, Sharply values can be approximated as the output values of the trained linear regressing model.

In this work, our FedeX architecture is applied with SHAP to account for the impact level of features on the anomalies that are predicted from the FedVAE-SVDD model, via their SHAP values. In this case, as depicted in

Fig. 3, the Kernel SHAP is fed in with the FedVAE-SVDD model and the test data to construct a local linear regression explanation model and compute the SHAP values. Subsequently, the explanatory model computes the SHAP value of classified anomalies and displays them visually. As feature values are measured by sensors, by using this explanation, operators or domain engineers can easily determine the sensors likely causing the abnormality and make a faster detection response.

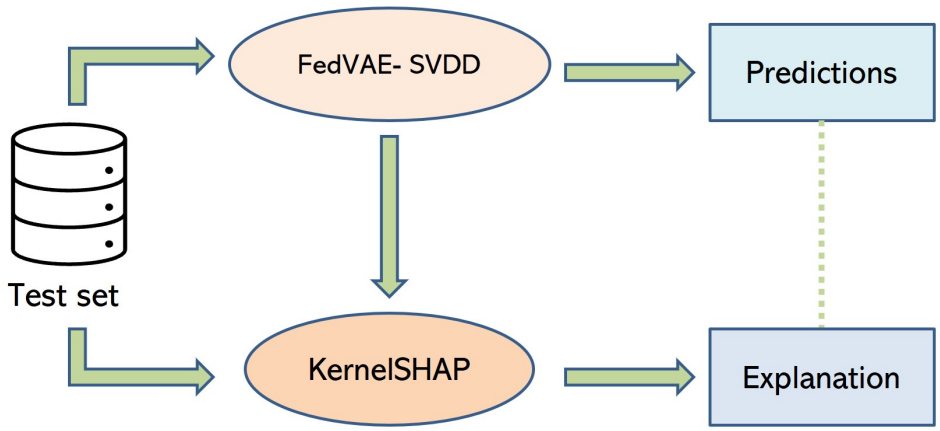


Figure 3: FedVAE-SVDD phase

4. Performance Evaluation

In this section, we evaluate the FedeX architecture in various aspects. From the detection performance perspective, the FedVAE-SVDD learning model is evaluated in comparison with different cutting-edge solutions for Anomaly detection in ICSs. The resource requirement of FedVAE-SVDD over the embedded edge device is also taken into account. Moreover, the results of using the XAI-SHAP technique to interpret the prediction results of FedVAE-SVDD are also described with the main case study in a SCADA liquid storage infrastructure dataset [27].

4.1. Experiment Setup

For the evaluation, we implement FedeX in a small-scale IoT testbed including:

- 4 Raspberry-Pi-4-Model-B kits acting as edge devices; Raspberry-Pi-4 equipped with quad-core 1.5 GHz ARM Cortex-A72 processor and 4 GB RAM with 32-bit Raspbian OS
- 01 Dell Precision 3640 Tower Workstation serves as Cloud Server; the workstation with Intel Core i710700K 3.8 GHz (up to 5.1 GHz), 16 GB RAM, working on Linux operating system.
- All edge devices and the Cloud Server are connected by a router through a WIFI interface

At the edge devices (i.e. Raspberry-Pi-4), we implement our FedeX framework in Python 3 with the Tensorflow 2 platform, which is built with the support of the FL framework - FedML[28]. In the FedeX architecture, the edge devices and cloud server exchange the weights and bias matrix of the VAE model using the standardized MQTT protocol for an IoT environment [29]. EMQ X Broker (2021) is hosted on the cloud server as an MQTT broker for better long-term performance. We discover EMQ X Broker as the most scalable open-source broker that could accept more advantageous devices linked to the server.

4.2. A Case Study for ICSs

In our main case study, we consider the SCADA liquid storage infrastructure dataset [27], which simulates a fuel storage system supplying an automated production line monitored by an ICS system. The high-level overview of the testbed system is shown in Fig. 4.

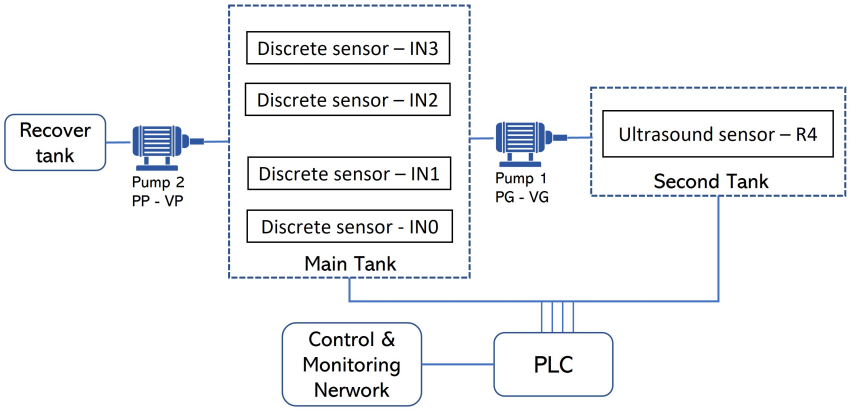


Figure 4: High level architecture of the SCADA liquid storage infrastructure system.

As depicted in Fig 4, the system is composed of the main tank and secondary tank with a capacity of 9 and 7 liters, respectively. Data is collected by connecting the sensors to a PLC; four discrete sensors in the main tank (i.e., IN0, IN1, IN2, IN3) and one in the secondary tank (i.e., R4) are used to measure the level of fuel in tanks. Besides, pump1 and pump2 control the flow of fuel between two tanks, it is also connected with two sensors (i.e., PP, PG). PLC registers 2 through 4 provided output data defining the system's state used to analyze the data obtained and register 2 provides the bits that indicate the discrete sensors' binary status. To extract the state of each sensor separately, a population count can be performed on the register. Register 3 holds the pump's active or inactive state, whereas Register 4 holds the ultrasound sensors' step value from 0 to 10,000. (e.g. Step 3,000 represents 2.1 liters of liquid in the tank).

As described in [27], the data set consists of 14 distinct scenarios. Each scenario includes one of 5 operational situations (such as sabotage, breakdown, accident, or cyber-attack) as well as six affected components. Affected components are those parts of the system that are directly impacted by the abnormality.

The data set has 10 features, and we use the normal data to perform the training model for all scenarios. In order to simulate the 4 distributed zones, we split the original data set into 4 independent subsets, each of which is used for local training corresponding to each of the 4 edge devices. After that, a test set containing both normal and abnormal data is utilized to evaluate

the model performance. Accordingly, the proportion of the training set and test set in the whole data set is 0.7 and 0.3, respectively.

4.3. FedeX Performance Evaluation

4.3.1. Detection Capability

As aforementioned, in our FedeX testbed, we set up 4 distributed edge zones. To evaluate the detection performance of the FedVAE-SVDD model, the common detection metrics such as F1 score, Accuracy, Recall, Precision are measured in various testing scenarios:

- Scenario 1: FedVAE-SVDD versus its centralized counterpart in our main case study of ICS (i.e. the SCADA liquid storage infrastructure dataset [27])
- Scenario 2: FedVAE-SVDD and its centralized counterpart versus other previously proposed AD solutions in our main case study of ICS and different SCADA datasets.

As elaborated above, the SVDD model is used to determine the optimal detection threshold for the FedVAE model depending on each different training dataset, in order to achieve good detection performance. Here, we use the Laplacian kernel as default to calculate the kernel distance. Any loss value greater than this threshold is considered an outlier. In our experiments, the optimal thresholds found for 4 distributed zones: Zone 1, Zone 2, Zone 3, and Zone 4 are 0.11, 0.09, 0.09, and 0.09 respectively.

Scenario 1: FedVAE-SVDD vs. its centralized VAE-SVDD over our main case study of a non-time series SCADA dataset

In this scenario, we measure the detection performance of our FedVAE-SVDD solution with the Centralized VAE-SVDD in which the training process is supposed to be carried out at the Central Cloud. Since the learning model converges after 3 communication rounds, the results retrieved after the rounds are shown in Table. 1.

In fact, in the context of seeking a detection scheme in a distributed manner that can provide a fast system response upon attacks or anomalies, and can work lightweight to cope with the limited computing capacity of edge devices in an IoT environment, we always have to think about the trade-off

Table 1: FedVAE-SVDD performance measured in 4 zones vs. Centralized VAE-SVDD over the SCADA liquid storage infrastructure dataset [27]

	Zone 1	Zone 2	Zone 3	Zone 4	Centralized
Threshold	0.11	0.09	0.09	0.09	0.26
Accuracy	1	0.9587	0.9992	0.9210	0.9017
Precision	1	0.9237	0.9985	0.864	0.9059
Recall	1	1	1	0.999	0.9806
F1	1	0.96	0.9992	0.9269	0.9418
AUC	1	1	1	0.92	0.9

with detection performance which is supposed to be slightly lower than the detection performance of the centralized monitoring and training manner.

However, as we can see, in our ICS main case study, the hybrid FedVAE-SVDD solution even outperforms the Centralized learning manner. It can be explained that Federated learning offers the improvement of generalizability of the VAE-SVDD model through the collaboration of multiple edge devices by taking advantage of separate data sources when compared to a single global model under data heterogeneity. FL eliminates a single point of failure due to its distributed nature. This can be considered as an advantage of Decentralized Learning, so the results when comparing our model with the Centralized learning method are slightly higher.

Considering the performance of FedVAE-SVDD only, we can see that all detection metrics are very good. Only Precision in Zone 4 gets a bit low at 0.864. However, in a smart factory, even the smallest abnormal incident can adversely affect the entire factory. So in general, we need to avoid discarding anomalies (i.e. Recall is important) and accept that sometimes the model can miss detecting a normal sample to be abnormal (i.e. Precision). Because engineers can easily test it and then operate the factory properly. Therefore, the Recall results of our model prove that this model can be a very good candidate to be deployed in a smart factory.

Scenario 2: FedVAE-SVDD vs. other Anomaly detection solutions over different SCADA data sets

First of all, as the main case study of our detection design, we compare the performance of FedVAE-SVDD with the results of other AD solutions

for ICSs found by a recent research work [8] who work in the same ICS context (i.e the same SCADA liquid storage infrastructure data set [27]). The results can be seen in Table. 2 in which our FedVAE-SVDD detection solution is shown to outperform other machine-learning algorithms LR, LDA, KNN, CART, NB, SVM in all metrics (i.e. Accuracy, Precision, Recall, and F1-Score). Even its centralized counterpart (i.e the centralized VAE-SVDD) performs better than those ones, which shows that this learning model is suitable for such a case study. Note that the detection results of our proposed solution are retrieved after 3 communication rounds as the convergence point of the learning model.

Table 2: Our proposal vs. other anomaly detection solutions over the SCADA liquid storage dataset

Learning model	Accuracy	Precision	Recall	F1-Score
LR	0.87	0.78	0.51	0.49
LDA	0.88	0.87	0.53	0.53
KNN	0.91	0.85	0.7	0.75
CART	0.94	0.86	0.86	0.86
NB	0.67	0.63	0.80	0.60
SVM	0.91	0.90	0.68	0.74
Centralized VAE-SVDD	0.9017	0.9059	0.9806	0.9418
FedVAE-SVDD @ Zone 1	1	1	1	1
FedVAE-SVDD @ Zone 2	0.9587	0.9237	1	0.96
FedVAE-SVDD @ Zone 3	0.9992	0.9985	1	0.9992
FedVAE-SVDD @ Zone 4	0.9210	0.864	0.999	0.9269

In the extended experiments, we run our FedVAE-SVDD model and the centralized VAE-SVDD model over the SWaT dataset [30] which is the case study of several other researches such as LAD-ADS [10], MADICS [12], and 1D CNN ensembled attacks [13]. As it can be seen in Table. 3, since SWaT represents a data type that is not purely time-series, therefore similar to the case of SCADA liquid storage dataset, FedVAE-SVDD and its centralized counterparts (VAE-SVDD) perform quite well in comparison with LAD-ADS [10], MADICS [12], and 1D CNN ensembled attacks [13] in terms of Recall and F1-score. In the FL-manner, the detection model FedVAE-SVDD provides higher detection performance which is varied from one zone to another

one depending on the local data of each zone.

Again, let us note that Recall and F1-score are the 2 important metrics for ICSs. In the case of imbalanced datasets like these considered datasets in which the number of abnormal samples is so much different from the number of normal samples, a good F1-score figure is necessary.

Table 3: Our proposal vs. other anomaly detection solutions over the SWaT dataset

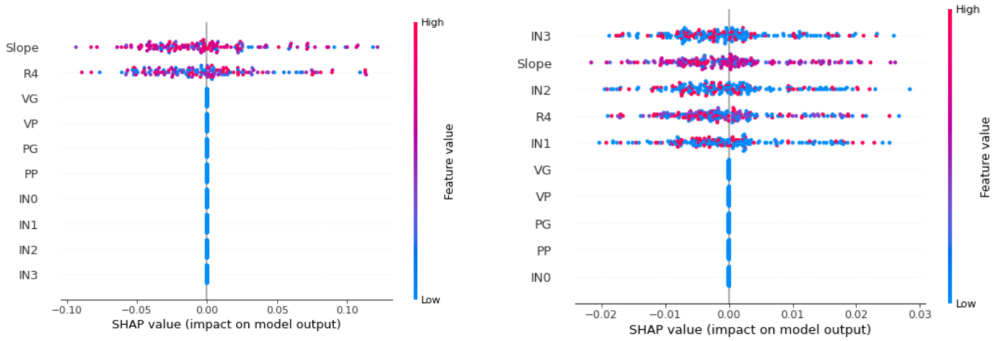
Learning model	Accuracy	Precision	Recall	F1-Score
LAD-ADS[10]	N.A	0.939	0.891	0.914
MADICS[12]	0.9659	0.984	0.75	0.851
1D CNN ensembled attacks[13]	N.A	1.0	0.853	0.920
1 FedVAE-SVDD @Zone 1	0.942	0.942	0.9999	0.97
FedVAE-SVDD @Zone 2	0.9727	0.9718	1	0.9857
FedVAE-SVDD @Zone 3	0.9427	0.9427	1	0.9705
FedVAE-SVDD @Zone 4	0.9433	0.9433	1	0.9708
Centralized VAE-SVDD	0.9725	0.9751	0.9962	0.9855

4.3.2. Explainable AI - SHAP

Although the above results demonstrate that FedeX achieves good anomaly detection performance, we want to investigate the reasons why they are predicted so. Since our case study is based on the data set gathered in a liquid storage infrastructure [27] as described in Section 4.2, we expect that FedeX could support domain engineers quickly and visually in finding and checking abnormal behavior of those sensors or actuators . Therefore, SHAP is employed to identify how features contribute to the anomalies predicted by the FedVAE-SVDD model. Thanks to this, decisions and priorities in checking and maintaining systems can be made effectively, allowing operators to save more time.

From a practical perspective, anomalies can arise from various threats such as accidents, sabotage, breakdown, and cyber-attack. This promotes us to perform two explanation scenarios, where SHAP is employed to explain two sets, corresponding to two different intervals, drawn randomly from anomalous samples predicted in the test set at Zone 1. The results of both scenarios are visualized in Fig.5, a summary plot for the distribution of SHAP values over whole computed data points, pointing out the impor-

tance of features through their impact. In the visualizations, the dots in each feature correspond to the SHAP values of each data point, piling up to depict density. The position on the x-axis is denoted by Shapley values and on the y-axis by features ordered as per importance. Besides, the value of features from low to high is displayed by color gradation.



(a) Explanation for detected anomaly 1 (b) Explanation for detected anomaly 2

Figure 5: Summary plot of SHAP values

As characterized in Fig.5(a), we can observe that overwhelmingly, Slope and R4 are critical features, while the other features do not contribute to the anomaly. Consequentially, it can be inferred that the ultrasound sensor which measures the physical values of the R4 and Slope feature may break down. This incident can come from some weather factors like humidity. Therefore, by checking the ultrasound sensor quickly, domain engineers can make reasonable solutions, without verifying other physical components in the system. On the other hand, if the sensor still works properly, i.e false alarm occurs, the operator can consider retraining the model for higher anomaly detection accuracy. For the remaining scenario, Fig.5(b) shows that IN3 is the most crucial feature, while Slope, IN2, R4, and IN1 have a remarkable influence. Based on these signs, as an engineer, we could determine that the anomaly is most likely to arise from sabotage impacting physical components such as the discrete sensors in the main tank and the ultrasound sensor, thereby prioritizing checking them.

In both of these scenarios, SHAP suggests that the R4 feature has a significant impact on predicted anomalies, similar to the analyses mentioned in a SCADA dataset research [31]. The authors confirm that the ultrasound sensor badly affects most of the abnormal scenarios in the dataset and the

value of R4 measured by this sensor is most significant. Accordingly, it can be seen that our XAI-based explanatory solution is capable of precisely identifying the primary cause related to the anomaly in reality.

In conclusion, based on these positive findings, we would like to make some comments and recommendations. Firstly, our scheme can make a comprehensive explanation for detected anomalies, boosting the reliability of FedeX. Besides, if there are the occurrence of unknown threats, FedeX will still support operators to determine affected physical components and come up with timely responses rather than inspecting the entire system. This issue may not be solved by other multi-class classification-based anomaly detection solutions. Furthermore, based on data records, we recommend that domain engineers should run SHAP periodically, for example, once per week, to check and schedule system maintenance depending on attack types, or to retrain the model for higher detection performance.

4.3.3. Edge Computing Capacity

Deploying a learning model at the edge is challenging due to limited capacity of those embedded devices. Therefore, to get insight into the efficiency and feasibility of the FedeX architecture, we conduct a few experiments for the FedVAE-SVDD training phase to evaluate the edge performance during the training, based on some metrics such as: power consumption, CPU usage, memory usage, and model running time.

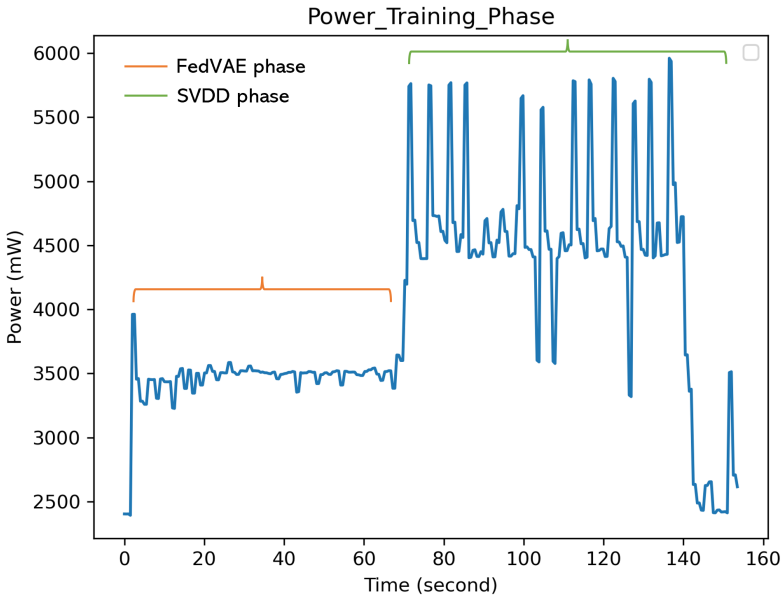


Figure 6: Power consumption of an edge device in one communication round

- **Power consumption**: Fig.6 illustrates the consumed power level during the FedVAE-SVDD training process in one communication round at an edge device (i.e the Raspberry-Pi-4), comprising two successive phases, namely FedVAE and SVDD. The measurement shows that the power consumption ranges from under 3500 mW to 6000 mW in the whole training process. Power consumption at the SVDD phase fluctuates strongly and is much higher than the FedVAE phase. As a result, to develop FedeX in IoT industrial systems, the edge devices should load the power consumption of the range from 5000 to 6000 mW on average to afford the worst case. These real-world metrics give us a better idea of how deploying distributed machine learning models on edge devices will consume more energy for that computation.

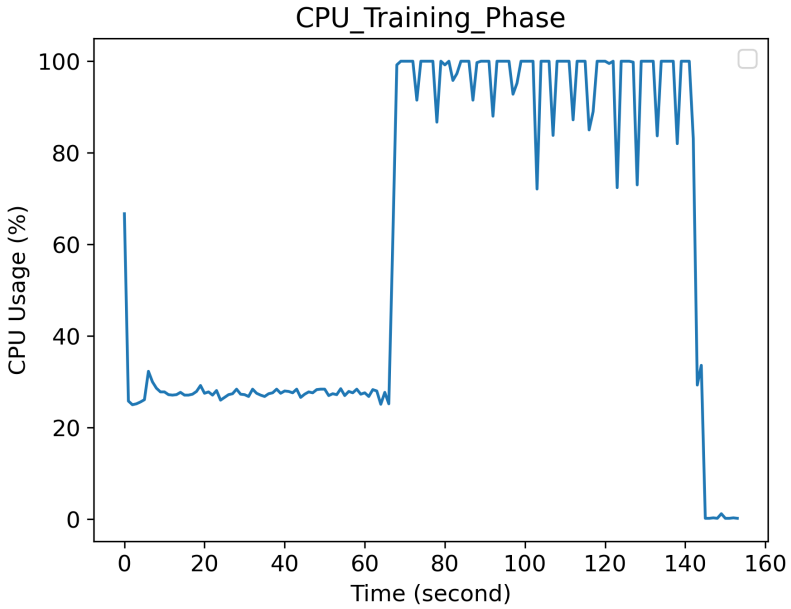


Figure 7: CPU usage of an edge device in one communication round

- **CPU usage**: Fig.7 shows the proportion of CPU usage during the FedVAE-SVDD process in one communication round at the edge device. It is conspicuous that the running time in the whole process is very fast, but in the worst case, the SVDD phase accounts for 100% of the CPU usage while this ratio of the FedVAE phase is just over 20%. Based on these findings, we would like to make a few recommendations. Firstly, in reality, with a runtime of only 70 seconds, the threshold update process in the SVDD phase can be retrained during system maintenance time or the night on schedule, rather than implemented on a real-time scale (i.e minutes or seconds scale). Thanks to this, other services would not be interrupted on the edge device every update time. From another perspective, these findings seem to be an acceptable trade-off between the running time for high detection performance and the hardware resource. Furthermore, it is possible to consider upgrading to edge hardware devices with higher processing capacity than Raspberry-Pi-4. With more powerful edge hardware, the FedVAE-SVDD model will be the effective detection model for such a factory.

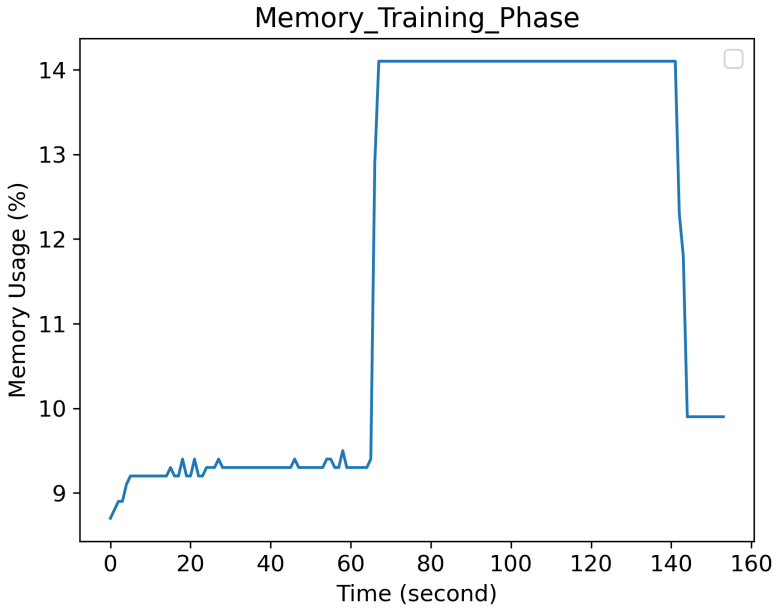


Figure 8: Memory usage of an edge device in one communication round

- **Memory usage**: In the same experimental setup with the power measurement, the percentage of memory usage in the VAE-SVDD phase at an edge device is demonstrated in Fig.8. Obviously, throughout the period of 150 seconds of the VAE-SVDD training process with one communication round, the memory usage of the VAE and SVDD phase is quite steady, with just over 9% and 14%, respectively. With these ratios, it can be inferred that in the training process, the memory resource is still available for other tasks.

- **Model running time**: Using deep learning to detect anomalies inside ICS is common worldwide, but we always have to face the fact that training models take quite a long time to train with the whole data set, even with smaller data sets from distributed zones. We have usually observed the computing running time on the scale of an hour or a few hours; and those figures mean that the system should be only retrained periodically on the scale of an hour, day, or week, since it can not capture any sudden change of traffic patterns in real-time. For example, if IoT devices inside a factory are aging over time, resulting

in a change in the data set characteristic, we will need to wait until the next period of the training model operation to retrain it again. In contrast to this setback, our outcomes in the testbed indicate that in terms of time, Raspberry-Pi-4 takes relatively little total time of 150 seconds to run the FedVAE-SVDD model (with just about 70 and 80 seconds in the FedVAE phase and SVDD phase respectively) in each communication round, whereas still ensuring the high detection performance depicted in Table.1. In our case study, it just needs to run 3 communication rounds for the training to converge and the detection results shown in the previous section are retrieved after 3 rounds. Therefore, the FedVAE-SVDD model takes only roughly 450 seconds (i.e 7.5 minutes) overall to produce such high detection performance. Basically, this shows that FedeX not only responds to anomalies in real-time but also handles the running time problem in a trade-off for a high performance described in a previous work [32], especially for the smart factory scenario as presented in this paper.

5. Conclusion

In this paper, we have elaborated our proposed hybrid model which combines an effective and fast detection scheme based on VAE and SVDD with the Federated-Learning technique that enables the hybrid model to perform efficiently on the weak computing hardware of distributed edge devices installed in the IoT-based system of a Smart Factory. With the FL architecture design, the detection task is distributed to smaller local zones located in the last premise of traffic senders. Therefore, anomalies or attacks can be quickly identified and quarantined in each separate zones. This FL architecture also helps to deal with Big Data created from a variety of devices inside a huge smart Factory 4.0 of the future.

In addition, this detection model stands out with a very fast training time in the minute-time scale (i.e 7.5 minutes). In case IoT devices in a Factory are aging, leading to changes of data patterns over time (i.e. concept drift), FedeX still works well since it can be retrained quickly every 7.5 minutes.

Moreover, the scheme has been proved to achieve high anomaly detection metrics such as Accuracy, Precision, Recall, F1-score, especially in the context of distributed training environment where the different edge of numerous zones trains their own model from different datasets.

In this paper, we also introduce XAI to help enhancing anomaly detection by interpreting how features contribute to the anomalies predicted by a "black-box" ML-based learning model. This way paves the way for engineers to have a deeper outlook on checking the systems more effectively.

Acknowledgment

This work was supported by Hanoi University of Science and Technology (HUST) under Project T2021-PC-010.

References

[1] Y. Lu, X. Xu, L. Wang, Smart manufacturing process and system automation a critical review of the standards and envisioned scenarios, Journal of Manufacturing Systems 56 (2020) 312–325.

[2] H. HaddadPajouh, A. Dehghantanha, R. M. Parizi, M. Aledhari, H. Karimipour, A survey on internet of things security: Requirements, challenges, and solutions, Internet of Things 14 (2021) 100129.

[3] N. Tuptuk, S. Hailes, Security of smart manufacturing systems, Journal of Manufacturing Systems 47 (2018) 93–106.

[4] S. Abdulrahman, H. Tout, H. Ould-Slimane, A. Mourad, C. Talhi, M. Guizani, A survey on federated learning: The journey from centralized to distributed on-site learning and beyond, IEEE Internet of Things Journal 8 (2021) 5476–5497.

[5] T. Miller, Explanation in artificial intelligence: Insights from the social sciences, Artificial Intelligence 267 (2019) 1–38.

[6] D. P. Kingma, M. Welling, Auto-encoding variational bayes (2014).

[7] D. M. J. Tax, A. Ypma, R. P. W. Duin, Support vector data description applied to machine vibration analysis, 1999.

[8] G. E. I. Selim, E. E.-D. Hemdan, A. M. Shehata, N. A. El-Fishawy, Anomaly events classification and detection system in critical industrial internet of things infrastructure using machine learning algorithms, Multimedia Tools and Applications 80 (2021) 12619–12640.

[9] S. M. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions, in: Proceedings of the 31st international conference on neural information processing systems, pp. 4768–4777.

[10] T. K. Das, S. Adepu, J. Zhou, Anomaly detection in industrial control systems using logical analysis of data, Computers & Security 96 (2020) 101935.

[11] H. R. Ghaeini, D. Antonioli, F. Brasser, A.-R. Sadeghi, N. O. Tippenhauer, State-aware anomaly detection for industrial control systems, in: Proceedings of the 33rd Annual ACM Symposium on Applied Computing, SAC '18, Association for Computing Machinery, New York, NY, USA, 2018, p. 16201628.

[12] . L. Perales Gmez, L. Fernndez Maim, A. Huertas Celdrn, F. J. Garca Clemente, Madics: A methodology for anomaly detection in industrial control systems, Symmetry 12 (2020).

[13] M. Kravchik, A. Shabtai, Detecting cyber attacks in industrial control systems using convolutional neural networks, in: Proceedings of the 2018 Workshop on Cyber-Physical Systems Security and PrivaCy, pp. 72–83.

[14] Q. P. Nguyen, K. W. Lim, D. M. Divakaran, K. H. Low, M. C. Chan, Gee: A gradient-based explainable variational autoencoder for network anomaly detection, in: 2019 IEEE Conference on Communications and Network Security (CNS), pp. 91–99.

[15] Y. Liu, S. Garg, J. Nie, Y. Zhang, Z. Xiong, J. Kang, M. S. Hossain, Deep anomaly detection for time-series data in industrial iot: A communication-efficient on-device federated learning approach, IEEE Internet of Things Journal 8 (2021) 6348–6358.

[16] V. Mothukuri, P. Khare, R. M. Parizi, S. Pouriyeh, A. Dehghantanha, G. Srivastava, Federated learning-based anomaly detection for iot security attacks, IEEE Internet of Things Journal (2021) 1–1.

[17] C.-P. Chang, W.-C. Hsu, I. Liao, Anomaly detection for industrial control systems using k-means and convolutional autoencoder, in: 2019 International Conference on Software, Telecommunications and Computer Networks (SoftCOM), pp. 1–6.

[18] A. Al-Abassi, H. Karimipour, A. Dehghantanha, R. M. Parizi, An ensemble deep learning-based cyber-attack detection in industrial control system, IEEE Access 8 (2020) 83965–83973.

[19] K. Amarasinghe, K. Kenney, M. Manic, Toward explainable deep neural network based anomaly detection, in: 2018 11th International Conference on Human System Interaction (HSI), pp. 311–317.

[20] C. Hwang, T. Lee, E-sfd: Explainable sensor fault detection in the ics anomaly detection system, IEEE Access 9 (2021) 140470–140486.

[21] D. M. J. Tax, A. Ypma, R. P. W. Duin, Pump failure detection using support vector data descriptions, in: D. J. Hand, J. N. Kok, M. R. Berthold (Eds.), Advances in Intelligent Data Analysis, Third International Symposium, IDA-99, Amsterdam, The Netherlands, August 1999, Proceedings, volume 1642 of *Lecture Notes in Computer Science*, Springer, 1999, pp. 415–426.

[22] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).

[23] B. Scholkopf, S. Mika, C. Burges, P. Knirsch, K.-R. Muller, G. Ratsch, A. Smola, Input space versus feature space in kernel-based methods, IEEE Transactions on Neural Networks 10 (1999) 1000–1017.

[24] M. T. Ribeiro, S. Singh, C. Guestrin, "why should I trust you?": Explaining the predictions of any classifier, CoRR abs/1602.04938 (2016).

[25] A. Shrikumar, P. Greenside, A. Kundaje, Learning important features through propagating activation differences, CoRR abs/1704.02685 (2017).

[26] L. A. Petrosyan, N. A. Zenkevich, Game theory (second edition), World Scientific Publishing Company, 2 edition, 2016.

[27] P. M. Laso, D. Brosset, J. Puentes, Dataset of anomalies and malicious acts in a cyber-physical subsystem, Data in Brief 14 (2017) 186–191.

[28] G. Li, Y. Shen, P. Zhao, X. Lu, J. Liu, Y. Liu, S. C. Hoi, Detecting cyberattacks in industrial control systems using online learning algorithms, Neurocomputing 364 (2019) 338–348.

[29] R. A. Light, Mosquitto: server and client implementation of the mqtt protocol, Journal of Open Source Software 2 (2017) 265.

[30] J. Goh, S. Adepu, K. N. Junejo, A. P. Mathur, A dataset to support research in the design of secure water treatment systems, in: CRITIS.

[31] H. Hindy, D. Brosset, E. Bayne, A. Seeam, X. Bellekens, Improving siem for critical scada water infrastructures using machine learning, Lecture Notes in Computer Science (2019) 319.

[32] T. T. Huong, T. P. Bac, D. M. Long, T. D. Luong, N. M. Dan, L. A. Quang, L. T. Cong, B. D. Thang, K. P. Tran, Detecting cyberattacks using anomaly detection in industrial control systems: A federated learning approach, Computers in Industry 132 (2021) 103509.