



HAL
open science

Light-Weight Federated Learning-based Anomaly Detection for Time-Series data in Industrial Control Systems

Truong Thu Huong, Ta Phuong Bac, Le Quang Anh, Dan Minh Nguyen, Cong Thanh Le, Hoang Xuan Nguyen, Ha Thu Do, Hung Tai Nguyen, Kim Phuc Tran

► **To cite this version:**

Truong Thu Huong, Ta Phuong Bac, Le Quang Anh, Dan Minh Nguyen, Cong Thanh Le, et al.. Light-Weight Federated Learning-based Anomaly Detection for Time-Series data in Industrial Control Systems. *Computers in Industry*, 2022, 140, pp.103692. 10.1016/j.compind.2022.103692. hal-03680865

HAL Id: hal-03680865

<https://hal.science/hal-03680865v1>

Submitted on 29 May 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Computers in Industry

Light-Weight Federated Learning-based Anomaly Detection for Time-Series data in Industrial Control Systems

--Manuscript Draft--

Manuscript Number:	
Article Type:	Research Paper
Keywords:	Anomaly detection, ICS, Federated Learning, Autoencoder, Transformer, Fourier
Corresponding Author:	Thu Huong Truong, PhD Hanoi University of Science and Technology Hanoi, VIET NAM
First Author:	Thu Huong Truong, PhD
Order of Authors:	Thu Huong Truong, PhD Phuong Bac Ta Anh Quang Le Minh Dan Nguyen Thanh Cong Le Xuan Hoang Nguyen Thu Ha Do Tai Hung Nguyen Kim Phuc Tran
Abstract:	<p>With the emergence of the Industrial Internet of Things (IIoT), potential threats to smart manufacturing systems are increasingly becoming challenging, causing severe damage to production operations and vital industrial assets, even sensitive information. Hence, detecting irregularities for time-series data in industrial control systems that should operate continually is critical, ensuring security and minimizing maintenance costs. In this study, with the hybrid design of Federated learning, Autoencoder, Transformer, and Fourier mixing sublayer, we propose a robust distributed anomaly detection architecture that works more accurately than several most recent anomaly detection solutions within the ICS contexts, whilst being fast learning in minute time scale. This distributed architecture is also proven to achieve lightweight, consume little CPU and memory usage, have low communication costs in terms of bandwidth consumption, which makes it feasible to be deployed on top of edge devices with limited computing capacity.</p>

Research Highlights

- A fast-learning model in 20-minute time scale that can cope with frequent updating
- A light-weight detection scheme in terms of CPU, Memory usage, and running time
- Faster system response upon attacks since detection is implemented near the sources.
- An accurate anomaly detection scheme for time-series data
- Federated learning to reduce bandwidth consumption on the link from Edge to Cloud.

Light-Weight Federated Learning-based Anomaly Detection for Time-Series data in Industrial Control Systems

Truong Thu Huong^a, Ta Phuong Bac^b, Le Anh Quang^a, Nguyen Minh Dan^a,
Le Thanh Cong^a, Nguyen Xuan Hoang^a, Do Thu Ha^c, Nguyen Tai Hung^a,
Kim Phuc Tran^d

^a*Hanoi University of Science and Technology, Vietnam*

^b*School of Electronic Engineering, Soongsil University, Seoul, Korea*

^c*International Research Institute for Artificial Intelligence and Data Science, Dong A University, Danang, Vietnam*

^d*Université de Lille, ENSAIT, GEMTEX, F-59000 Lille, France*

Acknowledgment

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

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

Light-Weight Federated Learning-based Anomaly Detection for Time-Series data in Industrial Control Systems

Abstract

With the emergence of the Industrial Internet of Things (IIoT), potential threats to smart manufacturing systems are increasingly becoming challenging, causing severe damage to production operations and vital industrial assets, even sensitive information. Hence, detecting irregularities for time-series data in industrial control systems that should operate continually is critical, ensuring security and minimizing maintenance costs. In this study, with the hybrid design of Federated learning, Autoencoder, Transformer, and Fourier mixing sublayer, we propose a robust distributed anomaly detection architecture that works more accurately than several most recent anomaly detection solutions within the ICS contexts, whilst being fast learning in minute time scale. This distributed architecture is also proven to achieve lightweight, consume little CPU and memory usage, have low communication costs in terms of bandwidth consumption, which makes it feasible to be deployed on top of edge devices with limited computing capacity.

Keywords: Anomaly detection, ICS, Federated Learning, Autoencoder, Transformer, Fourier

1. Introduction

Industrial control systems (ICS) [1] refers to many types of control systems and related instruments, which comprise devices, systems, networks, and controls used to operate and automate industrial processes. ICS devices and protocols are now deployed in every industrial sector and vital infrastructure, including manufacturing, transportation, energy, gas pipelines, water treatment, and so on. Since an ICS architecture contains valuable information that can affect the performance of the whole industry, it becomes a

9 significant target for attacks from a variety of threats. Therefore, it is cru-
10 cial to maintain a close eye on these systems' behavior for attack events by
11 using anomaly detection-based techniques. Moreover, threats are becoming
12 more complex, thus an anomaly detection solution that can promptly and
13 correctly identify attacks whilst being light-weight enough to be implemented
14 on devices with low computing capabilities in IoT-based industrial control
15 systems is required.

16 Various anomaly detection methods for time-series data in the ICS con-
17 text have been proposed, such as RNN [2], LSTM [2], and GRU [3]. However,
18 the performance of these sequence models still requires improvement. The
19 primary problem with RNNs is that gradients are propagated over multiple
20 stages, which tends to cause them to vanish or explode. For that reason,
21 Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) have
22 been widely adopted in time-series anomaly detection and have found partic-
23 ular success. Nevertheless, these algorithms still suffer from their sequential
24 nature, as with RNN and its variants. Being inherently sequential hinders
25 parallelization within samples, which slows down the training and inference
26 processes, especially with long sequences.

27 To tackle those limitations, we propose to use a Transformer-based ap-
28 proach for anomaly detection in this paper. Unlike the other existing sequence-
29 to-sequence models, Transformer, which was initiated in [4] to solve NLP
30 problems, does not employ Recurrent Networks. Instead, the Transformer
31 model deploys an attention mechanism to extract dependency between each
32 part of the incoming data simultaneously, thus making it very parallelizable.
33 However, the attention mechanism usually requires high computation cost
34 and large memory storage, so it might not be suitable for the distributed
35 ICS contexts where distributed computing is handled at edge devices with
36 limited hardware capacity. Hence, in our solution, we also make use of Au-
37 toencoder (AE) to reduce the dimension of input data whilst still preserving
38 the most vital information. In addition, the Transformer model's running
39 time is further sped up by replacing the attention layer in the Transformer
40 block with a Discrete Fourier Transform (DFT) sublayer. By transforming
41 the data from the time domain into the frequency domain, this Fourier trans-
42 formation can capture the relationship between the input sequences similar
43 to the attention mechanism [5]. DFT is a direct transformation that will
44 result in a highly efficient and rapid approach since no learnable paramet-
45 ers are required. Besides, thanks to the implementation of the Fast Fourier
46 Transform (FFT) algorithm, DFT reaches a substantially lower computa-

47 tional complexity of $\mathcal{O}(n \log n)$, compared to $\mathcal{O}(n^2)$ of the DFT calculation
48 using the conventional matrix method.

49 From another perspective, ICS is a distributed system containing multiple
50 components located on different machines that communicate and coordinate
51 actions to synchronize as a single coherent system. Therefore, in this paper,
52 we design an overall Federated learning (FL)-based anomaly detection archi-
53 tecture so called FATRAF. FL is one of the most promising and adaptive
54 candidates for communication costs in a distributed environment. FATRAF
55 is able to provide knowledge of other data patterns from other edge zones
56 to each local model through a federated global model update. Furthermore,
57 implementing anomaly detection tasks locally at each edge and federating
58 those local models with FL improve the system response time upon attack
59 arrivals since the detection model is conducted directly right at the edge,
60 which is close to attack sources. The synchronization of the FL technique
61 and an ML-based detection strategy aids in achieving both benefits of detec-
62 tion effectiveness and light-weight computing.

63 Overall, FATRAF brings the following advantages:

- 64 1. A light-weight local learning model for anomaly detection in ICS based
65 on a combination of Autoencoder - Transformer - Fourier to bring fast
66 learning time and consume hardware resources reasonably. That makes
67 FATRAF be feasibly implemented in practical distributed edge devices
68 in an IoT-based ICS architecture.
- 69 2. An unsupervised learning model based on normal data only, along
70 with a dynamical-threshold-determining scheme named Kernel Quan-
71 tile Estimation (KQE) [6] to tune the detection mechanism dynami-
72 cally. Therefore, it is able to dynamically keep track of new anomalous
73 patterns that may change over time in ICS, while yielding high de-
74 tection performance for time-series data in industrial control systems,
75 compared to state of the art solutions.
- 76 3. A federated learning (FL) framework to enable efficient distributed
77 anomaly detection near anomaly/attack sources. Hence the system
78 response time upon attacks can be improved. Distributed or edge com-
79 puting helps blocking an infected zone without affecting the common
80 operation of the entire system, therefore enhancing production effi-
81 ciency. FL allows the edge sites to share model information with each
82 other, aiming to optimize anomaly detection performance globally. In
83 practice, this solves the lack of training data in each edge site, especially

84 with multivariate and high-dimensional data sets.

85 The rest of our paper is represented as follows: Section 2 describes state
86 of the art in the field of anomaly detection for ICSs. Section 3 elaborates
87 our design and integration of the whole solution - FATRAF - that yields
88 an efficient detection performance whilst being light-weight, achieving faster
89 training time and consuming fewer hardware resources. The performance
90 evaluation and experiments are described in Section 4 in which we investi-
91 gate the detection performance of FATRAF as well as its edge computing
92 efficiency. Finally, conclusions are presented in Section 5.

93 2. Related Work

94 Anomaly detection has always been a critical issue in Smart Manufactur-
95 ing (SM), which requires timely detection and accuracy. Solutions have been
96 proposed to detect cyber attacks for ICS, such as [7], [8], [9],[10], and [11]. In
97 [7], the authors performed an ensemble method using a deep neural network
98 and decision tree for attack identification. The hybrid model can also ad-
99 dress commonly-encountered issues with imbalanced massive data sets. The
100 performance is evaluated over ICS data sets, and the model provides the
101 accuracy of 99.67 % and 96.00 % on the SWaT [12] and gas pipeline data
102 sets [13], respectively. However, the authors used all normal and attack data
103 for the training process, resulting in higher accuracy but lack of adaptabil-
104 ity when new attack patterns are employed. In the same direction of using
105 supervised learning, work [8] proposed a measurement intrusion detection
106 system approach to detect any common abnormal activity in an ICS system
107 with the HAI data set [14]. They applied the well-known supervised learning
108 algorithms such as K-Nearest Neighbour, Decision Tree, and Random Forest,
109 with the last one having the best performance of 99.67%. However, the data
110 in the ICS system is commonly in time series, and anomalous patterns change
111 continuously. Although the performance is proven quite high, the model in
112 [8] is incapable of extracting characteristics of time-series data; thus, the ac-
113 tual performance may differ. Furthermore, as we reproduce their proposed
114 method with the corresponding data set, we notice that the published classi-
115 fication performance appears to have been calculated with a micro-averaging
116 strategy, which is inappropriate in binary classification problems, especially
117 when the correct detection of instances of minority class (i.e., anomaly class)
118 is crucial to the overall operation. Therefore, contrary to the opinion pro-
119 vided by work [8], we presume that using unsupervised training models with

120 normal data will be a more effective method in detecting attacks and adapt-
121 ing to new abnormal behaviors. In addition, an anomaly detection frame-
122 work with the combination of k-mean and convolutional neural network was
123 proposed in work [10]. These techniques, however, do not provide optimal
124 performance for time-series data. It can reach an accuracy of 95.53% and
125 an F1-score of 89.08% with the gas pipeline data set, as the paper results
126 show. In another aspect, work [9] combined some popular machine learn-
127 ing methods such as 1D-CNN, Undercomplete Autoencoder, and Principal
128 Component Analysis with short-time Fourier transformation, transforming
129 time-domain signals into frequency representation to remove noise and han-
130 dle slow attacks. Similar to work [10], the performance of these models has
131 not been optimized with F1 scores of only 82 - 88% over the SwaT data set.

132 Since ICS data is frequently in time series, it is reasonable to use sequence
133 models such as Long Short-term Memory (LSTM) and Gated Recurrent Unit
134 (GRU) to capture temporal relationships in data sequences. Nevertheless,
135 these solutions still suffer from their sequential nature. Being inherently
136 sequential hinders parallelization within samples, which slows down train-
137 ing and inference processes, especially with long sequences. Therefore, work
138 [15], [16], [17], and [18], for example, did not yield significant anomaly de-
139 tection performance when employing LSTM and GRU for ICS time series
140 data. In work [15], the author presented a data-driven predictive modeling
141 approach for ICS systems. The models such as Recurrent Neural Network
142 (RNN), LSTM, and GRU were used to detect anomalies. The best-achieved
143 accuracy is 81.38% over the SCADA data set. To detect anomalies with de-
144 centralized on-device data, work [16] presented a Federated Learning (FL)-
145 based anomaly detection strategy for IoT networks based on the combina-
146 tion of GRUs and LSTM. However, the proposed method's performance is
147 insufficient; the accuracy in each FL client reaches the highest performance
148 of 95.5%. Work [17] provided a methodology called MADICS for Anomaly
149 Detection in Industrial Control Systems using a semi-supervised anomaly de-
150 tection paradigm. The performance of MADICS in terms of Recall is slightly
151 low over its testing data sets. In work [18], the author proposed MAD-GAN
152 to deal with the lack of labeled data using an supervised method - Genera-
153 tive Adversarial Networks (GANs) and LSTM-RNN. This model considered
154 correlation between the spatial and temporal characteristics of multivariate
155 ICS data. However, the experimental results received over the SWaT and
156 WADI data sets indicate no trade-off between the measures such as Preci-
157 sion and Recall, while the F1-score remains low in general. Furthermore,

158 the authors of [18] did not account for the model training time, which is the
159 major drawback of LSTM and GRU. Work [19] proposed a cyberattacks de-
160 tection mechanism using the combination of Variational Autoencoder (VAE)
161 and LSTM. Although the detection performance is considerably high, the
162 model in [19] faces difficulty in terms of running time since the LSTM block
163 requires a long training time. Liu *et al.* [20] presented an attention CNN-
164 LSTM model within a FL framework for anomaly detection in IIoT edge
165 devices. However, since ICS data usually contains multiple features (the
166 Gas Pipeline and SWaT data sets have 17 and 51 features respectively), this
167 complex design may necessitate more computation and training time.

168 In order to tackle those shortcomings of the LSTM and GRU networks,
169 Google launched a novel, fully promising architecture known as Transformer
170 [4] in 2017, an encoder-decoder architecture based on an attention mechanism
171 rather than RNN. Although initially evolved in the field of Natural Language
172 Processing, this architecture has been deployed in various anomaly detection
173 applications. As an example, the spacecraft anomaly detection research in
174 [21] demonstrated that their transformer-encoder-based framework, with the
175 adoption of the attention mechanism and the masking strategy, could con-
176 serve time cost up to roughly 80% in comparison with LSTM, while only
177 reaching an F1 score of 0.78 on the NASA telemetry data set. From our
178 standpoint, this figure needs to be further ameliorated since the capture of
179 all actual abnormalities should be prioritized, which aims to minimize the
180 damage in the worst case. Similarly, by introducing an Anomaly-Attention
181 mechanism so as to measure the association discrepancy between anomalous
182 and normal samples, an unsupervised time series anomaly detection proposal
183 called Anomaly Transformer in [22] shows considerable prediction capabil-
184 ity over the service monitoring data set - Pooled Server Metrics (PSM).
185 Nonetheless, regarding the application in industrial water treatment, that
186 study achieves a modest F1 score of 94.07% and Recall of 96.73% on the
187 SWaT data set, so further improvement is needed.

188 With respect to anomaly detection problems in IoT contexts, we also find
189 that the transformer-inspired solution in a recent work [23] brings remarkable
190 performance. By leveraging a transformer encoder followed by a two-layer
191 feed-forward neural network, the classifier deployed in [23] performs well on
192 the Aposemat IoT-23 data set [24] with the best F1 score of 95%. Neverthe-
193 less, as aforementioned, the performance of such a classifier becomes degraded
194 when a new abnormal pattern or behavior arises since it is trained with both
195 defined normal and abnormal observations (i.e, labeled samples). Meanwhile,

196 our proposed solution is trained with completely normal data, attempting to
197 find non-conform patterns in the data during inference, which results in ef-
198 fective anomaly detection performance, even with unknown abnormalities or
199 attacks. Work [23], furthermore, lacks a comprehensive assessment in terms
200 of runtime as well as feasibility on actual hardware devices if deployed in
201 distributed ICS systems, which prompts us to conduct evaluation scenarios
202 for our solution itself. Besides, to the best of our knowledge on the rele-
203 vant studies, our transformer-based approach is considered at the forefront
204 of anomaly detection deployment for ICS ecosystems.

205 **3. System Architecture Design**

206 *3.1. System Architecture Overview*

207 In this paper, we propose an Anomaly Detection (AD) architecture
208 for time-series data in Industrial Control Systems (ICSs) which provides a
209 fast and efficient learning model combined with Federated learning for dis-
210 tributed computing. The overall proposed architecture is named **FATRAF**
211 (**F**ederated Learning-based **A**utoencoder-**T**ransformer-**F**ourier Anomaly De-
212 tection). Our goal is to design an AD system that has a fast training time and
213 is light weight to accommodate frequent learning update, while still either
214 retaining the same or improving the detection performance in comparison
215 with some existing AD solutions for ICSs in the literature.

216 As illustrated in Figure 1, FATRAF comprises two main components:

- 217 • **Edge sites:** In a factory, there are various manufacturing zones, in
218 which sensor systems are installed to gather readings that signify op-
219 erating states over time. Subsequently, the time-series data, as the
220 local data, is transmitted wirelessly to an edge device in the vicinity of
221 the corresponding manufacturing zone. Designated to monitor anom-
222 alies, these edge devices employ the local data as inputs for training
223 its own local anomaly detection model - ATRAF (**A**E-**T**ransformer-
224 **F**ourier learning model). This deployment allows detecting anomalies
225 timely right at the edge sites, making use of the computing capacity
226 of edge devices, and distributing heavy computation tasks that could
227 overload the cloud server.
- 228 • **Cloud Server:** The cloud server undertakes two primary functions:
229 system initialization and aggregation of local models sent from differ-

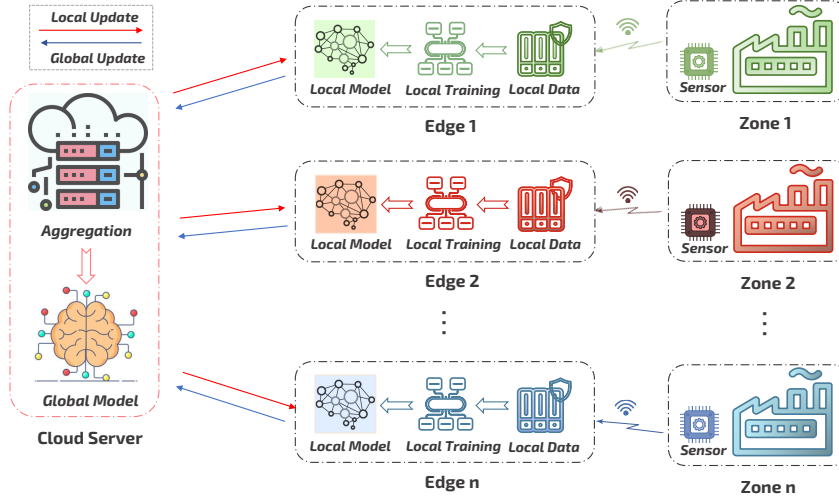


Figure 1: FATRAF Architecture

230 ent edges. The whole process of model aggregation and global model
 231 updating down to all local models are called Federated Learning.

232 As Figure 2 depicts, the Federated learning process of FATRAF comprises
 233 the following key steps:

- 234 1. *System Initialization*: At the beginning, the cloud server establishes
 235 a global model with specific learning parameters (i.e., the global FA-
 236 TRAF model), and sending it to each edge device of each corresponding
 237 edge site.
- 238 2. *Local Training*: After receiving that initial configuration, by utilizing
 239 the on-site data collected from sensors, edge devices conduct a local
 240 training process. Accordingly, anomaly detection is deployed right at
 241 these edge sites, enabling fast and timely system response upon attacks.
- 242 3. *Local Model Update*: After the local training, the edge devices send
 243 back the learned weights w_{t+1}^k to the cloud server for aggregation.
- 244 4. *Model Aggregation*: After deriving all trained weights w_{t+1}^k , the cloud
 245 server federates them and constructs a new global model version by the
 246 formula proposed by [25], as follows:

$$w_{t+1} = \sum_{k=1}^n \frac{v_k}{v} w_{t+1}^k \quad (1)$$

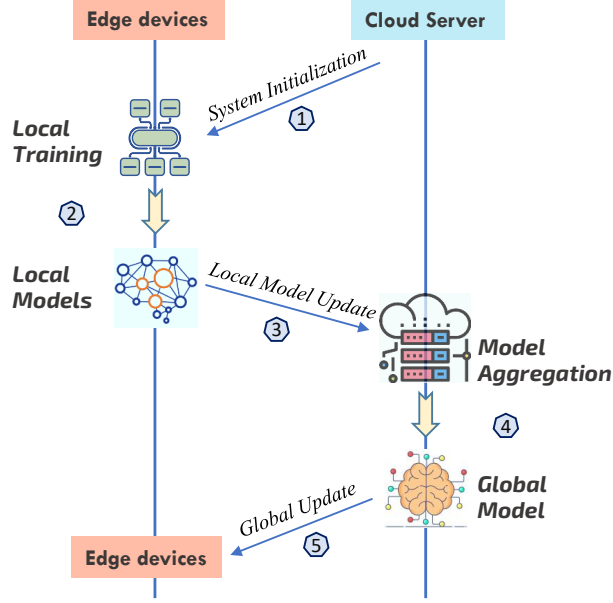


Figure 2: Flow diagram of FATRAF

Where:

n : the number of edge sites

v_k : amount of data of the k^{th} edge site

v : total amount of data of all edge sites

w_{t+1}^k : weight of the local model at the k^{th} edge site at time $t + 1$

w_{t+1} : weight of the federated global model at time $t + 1$.

Although there are several proposals on the federated learning technique to aggregate local models to a single global model such as [26, 27], through our various experiments, the approach of federating local weights presented in Equation 1 is still found to be the most efficient in terms of detection performance.

5. *Global Update*: Finally, the cloud server broadcasts back the new configuration w_{t+1} to each edge device so as to update the local models. In the next learning rounds, this communication process will repeat from the second step in order to optimize the local models until the global learning model converges.

264 In the following sections, we will describe how the local anomaly detection
265 module is designed and implemented to accelerate the learning speed, be
266 more light-weight to adapt well to the limited computing capacity of edge
267 hardware.

268 *3.2. Design of the ATRAF Detection Model at the Edge*

269 In the FATRAF architecture, the anomaly detection model deployed at
270 each edge site is designed as the hybrid learning model of Auto Encoder
271 (AE), Transformer, and Fourier - called **ATRAF**. The mission of this de-
272 sign is to create a light-weight, low-computing, and fast-learning model that
273 can yield high detection performance for time-series data, while still ensur-
274 ing fast training time to cope with the requirement of frequently re-updating
275 the learning model. This requirement comes from the fact that devices of a
276 factory can be aging, or attack/anomaly behaviors could change over time.
277 Therefore, in order to catch up with any new data pattern on time, the se-
278 curity system has to keep learning and updating the learning model quite
279 frequently. That leads to another requirement that any learning model de-
280 ployed on those distributed edge devices with limited computing capacity
281 should be light-weight to work fast and consume less computing resources.

282 Overall, ATRAF is an unsupervised learning model that relies totally on
283 normal data so as to attempt to detect abnormal patterns during inference,
284 thereby enhancing anomaly detection performance. This suits the fact that
285 novel abnormalities are unknown, or the data is not always labeled, thereby
286 solving the limitation of some classifiers such as the recent transformer-based
287 work [23] for IoT applications.

288 As illustrated in Figure 3, the AE model has the advantage of capturing
289 the distinct structural and temporal regularities of the data in each dis-
290 tributed zone. The ability to capture data temporal regularities makes this
291 model suitable for anomaly detection for time series data. In addition, AE
292 is a good method to reduce the data dimensionality that results in a shorter
293 learning time. It can be seen in Figure 3, we use the Autoencoder for the
294 local training process and only the encoder part for the testing process. As
295 the output of the AE Encoder block, compressed time-series data is then fed
296 into the Transformer-Fourier block to estimate the long-term correlation of
297 the data sequences and extract a distinguishable and meaningful criterion,
298 which is used to determine irregularities in the data.

299 Aiming to achieve high anomaly detection performance for time-series
300 data and better running time, Transformer is used as a remedy for other

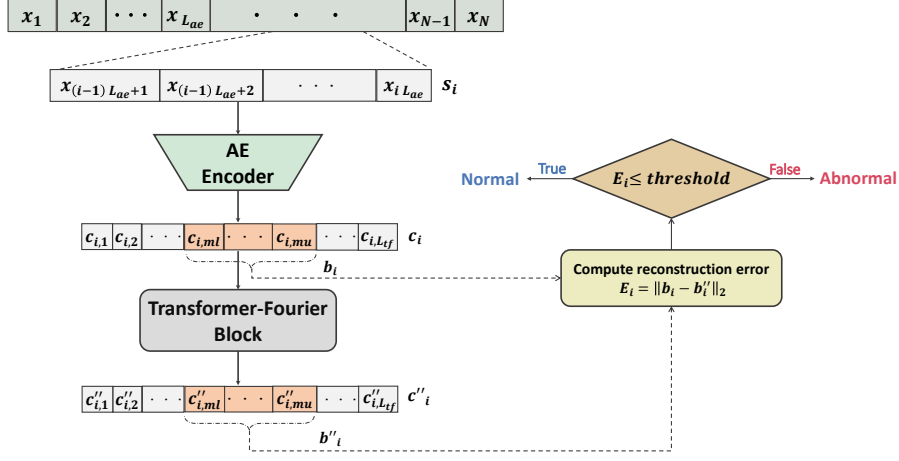


Figure 3: ATRAF Learning Model

301 time-series approaches such as LSTM and GRU since the Transformer model
 302 processes sequences in parallel. In the Transformer-Fourier block, the train-
 303 ing process is significantly accelerated by comparing to, for example, that of
 304 recurrent neural networks such as LSTMs, while retaining or even improv-
 305 ing detection performance. This is due to the fact that the Transformer-
 306 Fourier model processes sequences in parallel, instead of sequentially. The
 307 Transformer-Fourier block consists of a Transformer encoder layer followed by
 308 a Fourier layer (specifically a Fourier Transform with a position-wise feed-
 309 forward network). The Fourier layer has a lighter memory footprint since
 310 it replaces the self-attention sublayer in the Transformer encoder with an
 311 unparameterized Fourier transform sublayer. Furthermore, the Transformer-
 312 Fourier block proves to be more efficient than the Transformer encoder with
 313 the same number of layers, since it maintains the same level of performance
 314 as the Transformer encoder while shortening the training process, making it
 315 well-suited for resource-constrained edge devices. This statement is demon-
 316 strated through our experiments in Section 4.

317 In the following sections, we will describe in detail the implementation of
 318 the ATRAF model.

3.2.1. Design of the Autoencoder Block

320 Autoencoder (AE) is a symmetrical, unsupervised neural network with
 321 a "bottleneck" in the central hidden layer, which has fewer nodes than the

322 input and output layers. It is trained to reconstruct the output as closely
 323 as possible to the input. After this process, the network has learned to
 324 compress the data into a lower-dimensional code and rebuild the input from
 325 it. Generally, an AE consists of 3 components:

- 326 • **Encoder:** An encoder is a feed-forward, fully connected neural network
 327 that compresses the input into a lower-dimensional code.
- 328 • **Code:** Code, also known as the latent-space representation, is a com-
 329 pact "summary" or "compression" of the input. This code keeps the
 330 most essential information of the input while employing fewer features.
- 331 • **Decoder:** Decoder is also a feed-forward network and has a similar
 332 structure to the encoder. This network is in charge of reconstructing
 333 the input back to the original dimensions from the code.

334 In our design, the AE block is first trained locally at the edge devices
 335 for a number of epochs. After that, the decoder part is discarded while the
 336 encoder part is kept for compressing the input into a lower-dimensional code.
 337 We construct the encoder and decoder with only two fully-connected hidden
 338 layers each. The goal of this approach is to accomplish the simplicity and
 339 light-weight of the model, allowing it to be trained on edge devices with
 340 limited computing capacity, as well as to reduce communication costs while
 341 still providing sufficient detection performance. AEs, in particular, learn a
 342 map from the input to themselves via a pair of encoding and decoding stages.

$$\bar{X} = Decoder(Encoder(X))$$

343 Where:

344 X and \bar{X} are the input and output of AE

- 345 • Input $X = \{x_1, x_2, \dots, x_N\}$, where $X \subset \mathbb{R}^D$, is divided into k non-
 346 overlapping sequences $\{s_1, s_2, \dots, s_k\}$ where $s_i = \{x_{(i-1)L_{ae}+1}, \dots, x_{iL_{ae}}\}$
 347 is a sequence of length L_{ae} .
- 348 • These sequences are then used to train the local AE model. After
 349 the training process, the sequences are fed through the AE model's
 350 encoder, producing compressed code sequences $\{c_1, c_2, \dots, c_k\}$ where
 351 $c_i = \{c_{i,1}, c_{i,2}, \dots, c_{i,L_{tf}}\}$ with $L_{tf} \leq L_{ae}$.

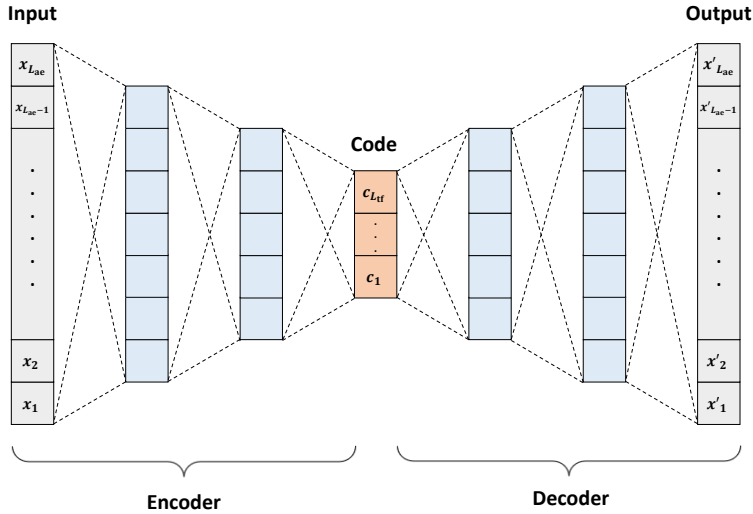


Figure 4: Autoencoder Configuration

352 In order to reduce the computation cost as well as improve the detec-
 353 tion performance, the length of the input sequence L_{ae} , the code sequence
 354 L_{tf} of the AE model are chosen through a pragmatic process of different
 355 configurations of L_{ae} and L_{tf} in order to achieve a good F1-score.

356 By fixing parameter L_{ae} at a time while varying L_{tf} , we found that the
 357 detection performance (i.e., F1 score) as well as the training time increases
 358 as L_{ae} increases and decreases with the ratio $\frac{L_{ae}}{L_{tf}}$.

359 3.2.2. Design of the Transformer-Fourier Block

360 In order to overcome the problem of sequential computing and take ad-
 361 vantage of GPU parallel computing to speed up the learning process, the
 362 attention mechanism was adopted to capture long-term dependencies.

363 In this paper, a design of the Transformer with the mixing Fourier Trans-
 364 form sublayer is proposed as a learning model that only relies on an at-
 365 tention mechanism to draw temporal dependencies in sequences. To speed
 366 up the Transformer encoder architecture, the unparameterized simple linear
 367 Fourier Transformation is integrated to replace the self-attention sublayer of
 368 the Transformer encoder. This linear mixing sublayer was proven to work
 369 efficiently in terms of learning speeds, especially at long input lengths [5].

370 The hybrid model of Transformer and Fourier transform mixing sublayer
 371 can provide a smaller memory footprint and faster runtime than an all-

372 attention-sublayer Transformer on GPU. As a result, it is much more suitable
 373 for resource-constrained edge devices. The operations of the Transformer-
 374 Fourier block in Figure 5 can be described as follows:

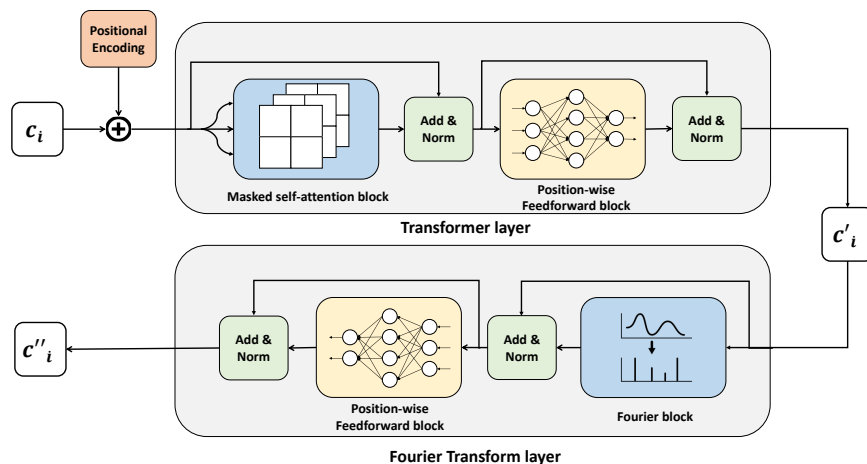


Figure 5: The Transformer-Fourier block

- 375 • Each output code sequence c_i of the local AE models are passed through
 376 the Positional Encoding layer, which injects relative positional infor-
 377 mation into the sequences. Since the sequences are processed in paral-
 378 lel, we need to ensure that the model is able to differentiate be-
 379 tween the positions in the code sequences. In this paper, we add
 380 the sine and cosine encodings of variable frequencies to each sequence
 381 $c_i = \{c_{i,1}, c_{i,2}, \dots, c_{i,L_{tf}}\}$.

$$PE_{p,2i} = \sin\left(\frac{p}{10^{\frac{si}{d_{hidden}}}}\right) \quad (2)$$

$$PE_{p,2i+1} = \cos\left(\frac{p}{10^{\frac{si}{d_{hidden}}}}\right) \quad (3)$$

383 where p is the position, i is the dimension index and d_{hidden} is the size
 384 of the code sequences' hidden dimension.

- 385 • A contiguous sub-sequence from index ml to mu (as illustrated in Fig-
 386 ure 3) in each resultant sequence is blocked from the Transformer en-
 387 coder (by assigning the respective score in the self-attention matrix

388 to minus infinity). The Transformer encoder block consists of a Self-
 389 Attention sublayer, followed by a Position-wise Feed Forward sublayer.
 390 The Transformer encoder can learn the inter-position dependencies of
 391 the non-blocked portions of the input sequences, producing an embed-
 392 ding $c'_i = \{c'_{i,1}, c'_{i,2}, \dots, c'_{i,L_{tf}}\}$ of the same dimensionality as the input
 393 sequence c_i .

394 • This embedding c'_i is then fed into the Fourier Transform layer. The
 395 Self-Attention sublayer of the Transformer encoder is replaced with
 396 an unparameterized Fourier sublayer. The Fourier sublayer applies
 397 two 1-dimensional Discrete Fourier Transforms along the hidden di-
 398 mension and the sequence dimension of the sublayer’s input and takes
 399 the real part of the result, i.e., $\Re(\mathcal{F}_{hidden}(\mathcal{F}_{sequence}(c'_i)))$. The unpa-
 400 rameterized Fourier transform is a relatively effective and light-weight
 401 mixing method and is able to retain 92-97% the performance of a Trans-
 402 former encoder while only taking 20% the training time [5]. The out-
 403 put of the Fourier transform layer is a reconstructed sequence of the
 404 input sequence c_i which is denoted $c''_i = \{c''_{i,1}, c''_{i,2}, \dots, c''_{i,L_{tf}}\}$. The re-
 405 construction errors are calculated between the masked sub-sequence
 406 of c_i , i.e., $b_i = \{c_{i,ml}, \dots, c_{i,mu}\}$ and the corresponding reconstructed
 407 sub-sequence $b''_i = \{c''_{i,ml}, \dots, c''_{i,mu}\}$ as demonstrated in Figure 3.

$$E_i = \|b_i - b''_i\|_2 \quad (4)$$

408 • An anomaly threshold λ_{Th} is determined by the Kernel Quantile Esti-
 409 mation (KQE) technique [6] that is used on the reconstruction errors
 410 to classify the input sequence $s_i = \{x_{(i-1)L_{ae}+1}, \dots, x_{iL_{ae}}\}$. Without
 411 loss of generality, assume that $E_1 \leq E_2 \leq \dots \leq E_k$. The anomaly
 412 threshold is determined by

$$\lambda_{Th} = \sum_{i=1}^k \left[\int_{\frac{i-1}{k}}^{\frac{i}{k}} \frac{1}{h} K\left(\frac{t-q}{h}\right) dt \right] E_i \quad (5)$$

413 Where K is the density function, chosen as the standard Gaussian
 414 kernel $K(x) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{x^2}{2})$; the asymptotically optimized bandwidth
 415 $h = \sqrt{\frac{p(1-q)}{k+1}}$ controls the estimator’s smoothness, and q ($0 < q < 1$) is
 416 the preset value.

417 4. Experiments and Evaluation

418 In this section, we focus on evaluating the performance of FATRAF in
419 terms of detection performance in different scenarios as well as the imple-
420 mentation feasibility in the Edge Computing environment.

421 First, we briefly introduce different time-series data sets in our exper-
422 iments and the data preprocessing. Second, we also describe our testbed
423 settings and tools. Finally, anomaly detection performance and edge com-
424 puting efficiency will be studied throughout our experiments.

425 4.1. Data Sets and Pre-processing

426 In this study, we take the data sets presented in Table 1 as the main ICS
427 use cases. In addition, to cross validate our solution detection performance
428 in diverse contexts, we utilize other time-series data sets of disparate areas
429 as listed in Table 2. In our experiments, we compare the performance with
430 some mostly up-to-date reference anomaly detection solutions for ICS that
431 run on top of some other time-series or non-purely time-series data sets.

432 In fact, all data sets need to be pre-processed before being fed into the
433 ATRAF learning model. As for the SCADA Gas Pipeline data set, it initially
434 carries many missing values (i.e., "?" values) throughout the entire data set.
435 To deal with this problem, we make use of the Last Observation Carried
436 Forward (LOCF) method [30], which uses the immediately previous value
437 within the same field to substitute the missing values. In the case of the
438 SWaT and HAI data sets, as their raw data sets contain many correlated
439 features, using all of them in our proposed learning model will increase the
440 computational burden considerably, resulting in exceptionally long training
441 time. For that reason, we perform feature selection to reduce the number of
442 features presented in each data point by investigating the correlation between
443 them: features with high correlation are removed; features with zero variance
444 are discarded.

445 Finally, all data sets should be normalized to have all processed features to
446 be in the same scale between 0 and 1 according to the min-max normalization
447 described in Equation 6:

$$x'_i = \frac{x_i - x_{min}}{x_{max} - x_{min}} \quad (6)$$

448 where x_i , x'_i are before and after values of the feature. x_{max} , x_{min} are the
449 maximum and minimum values of that feature.

Table 1: List of the data sets used for main use cases

Data sets	Description
Gas Pipeline [13]	Time-series data set was collected in 2015 from the Supervisory Control and Data Acquisition (SCADA) gas pipeline system by Mississippi State University Lab. Each data point has 17 features, containing network information as well as a payload which gives details about the gas pipeline’s state and settings.
SWaT [12]	Secure Water Treatment, launched in 2015, is the data set collected from a scaled down water treatment plant. Being applied in secure Cyber Physical Systems research, SWaT collects the plant’s continuous operation data in 11 days, in which 7 days of normal activities and 4 days under attacks. Each sample the SWaT data set contains 51 features from different sensors and actuators.
HAI [14]	HIL-based Augmented ICS data set stemmed from a practical ICS testbed augmented with a Hardware-In-the-Loop (HIL) simulator, introduced for ICS anomaly detection research. The testbed aims to emulate steam-turbine power and pumped-storage hydropower generation. Initially published in 2020, the time-series data set includes 59 features recorded under normal and abnormal (in case of attacks or system failures) behaviors.
Power Demand [28]	Univariate time-series data set, including 1-year-long power consumption readings of a Dutch research facility in 1997.

Table 2: List of the data sets used for cross validation

Data sets	Description
ECGs [28]	Time-series of the heartbeat electrical signals
Respiration [28]	Measurements of patient’s respiration when waking up by thorax extension
Gesture [28]	2-feature data set indicates the right hand’s coordinates while performing different actions
Space shuttle [28]	Solenoid current’s measurements of a Marotta MPV-41 series valve cycled on and off
NYC taxi [29]	Information on New York taxi passenger data stream (Jul 2014 - Jun 2015)

450 *4.2. Testbed Setup and Implementation*

451 In our experiments, the ATRAF learning model is implemented in two
 452 learning modes: Centralized Learning and Federated Learning. As the whole
 453 FATRAF architecture is designed to be light-weight, parallel computing of
 454 the ATRAF detection model should be leveraged. Therefore, the training
 455 and inference tasks are implemented in GPU-equipped devices.

456

457 In the Centralized-learning mode, the experiment is conducted using:

- 458 • a Dell Precision 3640 Tower workstation featuring an NVIDIA Quadro
 459 P2200 GPU and an Intel Core i7-10700K CPU with 16GB RAM.

460 In Federated-learning mode, the experiment is conducted using:

- 461 • 4 NVIDIA Jetson Nano B01 boards to emulate 4 edge devices (standing
 462 for 4 different distributed zones). Local learning models are trained
 463 with local data subsets at each edge.
- 464 • A ThinkSystem SR550 (Intel Xeon Silver 4210, 64GB RAM) to serve
 465 as the Cloud Server for aggregation tasks (i.e., Federated learning).
- 466 • WiFi interfaces for the edges and the Cloud to communicate their
 467 weight matrices via the MQTT protocol. MQTT is proven to be a
 468 light-weight, reliable and scalable protocol for IoT networks. For this

469 purpose, the open-source EMQ X Broker is hosted in the Aggregation
470 Server.

471 For the performance measurement purposes, some tools are used as follows:

- 472 • Tool *bmon* [31] is used to measure bandwidth occupation in each link
473 between the Edge–Server link in the Federated Learning–based archi-
474 tecture.
- 475 • Built-in utility *tegrastats* keeps track of computational resources usage
476 as well as energy consumption on edge devices during training and
477 testing tasks. This program extracts real-time information about the
478 usage of CPU, RAM, GPU, and the energy consumption of the NVIDIA
479 Jetson Nano board.

480 The implementation of the ATRAF model is written in Python 3.8.10 us-
481 ing the PyTorch framework (release 1.9.0). We also leverage FedML [32]
482 framework to realize the Federated Learning setting.

483 4.3. Performance Evaluation

484 4.3.1. Detection Performance Evaluation

485 To assess the detection performance of FATRAF, some common standard
486 metrics are used such as Precision, Recall and F1-Score. The definition of
487 these metrics are defined as follows:

$$488 \text{Precision} = \frac{TP}{TP+FP}$$

$$490 \text{Recall} = \frac{TP}{TP+FN}$$

$$492 \text{F1 - Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

494 where:

- 496 • *TP*: True Positive represents samples which are correctly classified as
497 positive class.
- 498 • *TN*: True Negative represents samples which are correctly classified as
499 negative class.
- 500 • *FP*: False Positive represents samples which are incorrectly classified
501 as positive class.

- 502 • *FN*: False Negative represents samples which are incorrectly classified
503 as negative class.

504 To calculate the performance metrics in our experiments, we apply the
505 simple point-adjust approach proposed in [33], which assumes it is adequate
506 to trigger a malfunction or attack alert within any subset of an anomaly
507 window. As we detect anomalies in windows of time-series data points, a
508 full window is viewed as anomalies in the ground truth label if it contains at
509 least one anomalous point. When the model detects any part of this window
510 as anomalous, the whole window is also considered as correctly detected as
511 anomalies.

512 It is also noted that during the evaluation process, we prioritize Recall
513 and overall F1-Score metric over Precision. This priority can be explained
514 that in real-life scenarios, detecting all actual attacks and malfunctions in
515 the system is often more critical. The cost of the system operator in case
516 of being attacked is too high; thus, a small number of false alarms is tolerable.

517

518 **Experiment 1 - Federated Learning vs. Centralized learning**

519

520 At first, in this experiment, we will compare the detection performance
521 of FATRAF with its centralized computing mode (i.e. implementing the
522 ATRAF learning model in the centralized cloud mode). Table 3 illustrates
523 the performance of the two modes over the 4 time-series data sets of ICSs
524 such as Power Demand, HAI, SWaT, and Gas Pipeline data sets. The detec-
525 tion performance of the centralized mode slightly outperforms the Federated
526 Learning mode. This can be obviously explained that the learning-based
527 model in the Federated Learning mode just learns from its own smaller local
528 data set, so we will have to trade off a bit of detection performance with the
529 benefits of Federated learning.

530 For cross-validation, the detection performance of both learning manners
531 is verified with other time-series data sets which do not belong to the ICS con-
532 text such as Space shuttle, Respiration, Gesture, NYC taxi, ECG. As Table
533 3 shows, the performance of FATRAF mostly approximates the Centralized-
534 learning mode of ATRAF in all cross-validation data sets. However, ATRAF
535 in both modes is proven to detect anomalies efficiently.

Table 3: FATRAF vs. its Centralized Learning mode

Data set	Centralized			FATRAF		
	Precision	Recall	F1	Precision	Recall	F1
Power Demand	0.9339	0.9797	0.9563	0.9285	0.9442	0.9363
Gas Pipeline	0.9699	1	0.9847	0.9683	1	0.9839
HAI	0.9039	0.9973	0.9483	0.8939	1	0.9440
SWaT	0.9404	0.9871	0.9632	0.9389	0.9775	0.9578
Space shuttle - TEK14	0.9874	1	0.9936	0.9738	1	0.9867
Space shuttle - TEK16	0.9851	1	0.9925	0.9296	1	0.9635
Space shuttle - TEK17	0.9728	1	0.9862	0.9728	1	0.9862
Respiration - nprs43	0.9567	0.9627	0.9597	0.9578	0.9730	0.9654
Respiration - nprs44	0.9174	0.9224	0.9199	0.9195	0.9799	0.9488
Gesture	0.9339	0.9921	0.9621	0.9331	0.9906	0.9610
Nyc taxi	0.8837	1	0.9382	0.9419	1	0.9701
ECG - Chfdb_chf01_275	0.9761	1	0.9879	0.9761	1	0.9879
ECG - chfdb_chf13_45590	0.9810	1	0.9904	0.9773	1	0.9885
ECG - chfdbf15	0.9118	1	0.9538	0.9118	1	0.9538
ECG - ltstdb_20221_43	0.9841	1	0.9920	0.9841	1	0.9920
ECG - ltstdb_20321_240	0.9558	1	0.9774	0.9714	1	0.9855
ECG - mitdb_100_180	0.9536	1	0.9763	0.9474	1	0.9730
ECG - qtddbse102	0.7990	1	0.8883	0.7891	1	0.8821
ECG - stdb_308_0	0.9547	1	0.9768	0.9521	1	0.9755
ECG - xmitdb_x108_0	0.9795	1	0.9896	0.9856	1	0.9927

536 **Experiment 2 - Performance of Federated Learning based ap-**
537 **proaches**

538

539 In this experiment, the detection performance of FATRAF is compared
540 with a recent detection work for ICS that applies Federated Learning [19]
541 - Federated Learning called FL-VAE-LSTM. As Table 4 shows, FATRAF
542 improves the detection performance in all metrics: Precision, Recall, F1-
543 score with both of the ICS data sets: Power Demand and Gas Pipeline. It
544 also outperforms FL-VAE-LSTM in most of the cross-validation time-series
545 data sets. In conclusion, the results show that FATRAF can detect anomalies
546 efficiently and stably in ICS systems that have time-series data. Later in the
547 following subsection, the running time of FATRAF and FL-VAE-LSTM [19] is
548 also compared to show that the training time of FATRAF much outperforms
549 FL-VAE-LSTM.

Table 4: Federated-Learning approaches over the time-series data sets

Data set	FL-VAE-LSTM [19]			FATRAF		
	Precision	Recall	F1	Precision	Recall	F1
Power Demand	0.7355	0.9100	0.8135	0.9285	0.9442	0.9363
Gas Pipeline	0.9609	0.9982	0.9792	0.9683	1	0.9839
Space shuttle - TEK14	0.8623	0.8431	0.8536	0.9738	1	0.9867
Space shuttle - TEK16	1	1	1	0.9296	1	0.9635
Space shuttle - TEK17	0.9650	1	0.9822	0.9728	1	0.9862
Respiration - nprs43	0.9313	0.5530	0.6939	0.9578	0.9730	0.9654
Respiration - nprs44	0.5347	0.5027	0.5182	0.9195	0.9799	0.9488
Gesture	0.5278	1	0.6910	0.9331	0.9906	0.9610
Nyc taxi	0.9606	1	0.9799	0.9419	1	0.9701
ECG - Chfdb_chf01_275	0.9175	1	0.9570	0.9761	1	0.9879
ECG - chfdb_chf13_45590	0.9489	1	0.9738	0.9773	1	0.9885
ECG - chfdbf15	0.9458	1	0.9721	0.9118	1	0.9538
ECG - ltstdb_20221_43	1	1	1	0.9841	1	0.9920
ECG - ltstdb_20321_240	1	1	1	0.9714	1	0.9855
ECG - mitdb_100_180	1	1	1	0.9474	1	0.9730
ECG - qtbsel102	0.9604	1	0.9797	0.7891	1	0.8821
ECG - stdb_308_0	0.6073	0.6373	0.6220	0.9521	1	0.9755
ECG - xmitdb_x108_0	1	0.7628	0.8654	0.9856	1	0.9927

550 **Experiment 3 - Comparison with other existing AD solutions**
 551 **for ICS over different contexts**

552

553 The third experiment focuses on the comparison of ATRAF in both the
 554 Centralized and Federated Learning mode and several most recent works
 555 [7, 8, 22] in the field of anomaly detection in ICS.

556 Figure 6 depicts the detection performance of the 3 solutions: SAE pro-
 557 posed in [7], our FATRAF and its centralized computing mode on 2 different
 558 data sets: Gas Pipeline [13] and SWaT [12]. As these data sets are primary
 559 case studies in [7], using them in the detection performance comparison will
 560 yield better validation results. Whilst FATRAF gets marginally lower results
 561 than its centralized counterpart as shown in Experiment 1, it can be seen
 562 that on the Gas Pipeline data set, FATRAF outperforms SAE on all three
 563 metrics with Precision, Recall, F1-Score being 0.9683, 1, 0.9839, respectively,
 564 as opposed to 0.9463, 0.9372, 0.9383 achieved by SAE.

565 On the contrary, the experiment on the SWaT data set shows that FA-
 566 TRAF a bit underperforms SAE’s roughly perfect performance in Precision,
 567 Recall, F1-Score of 0.97, 0.99, and 0.99, respectively. This can be explained
 568 as SWaT is not a purely time-series data set whilst FATRAF is designed to
 569 cope with time-series effectively.

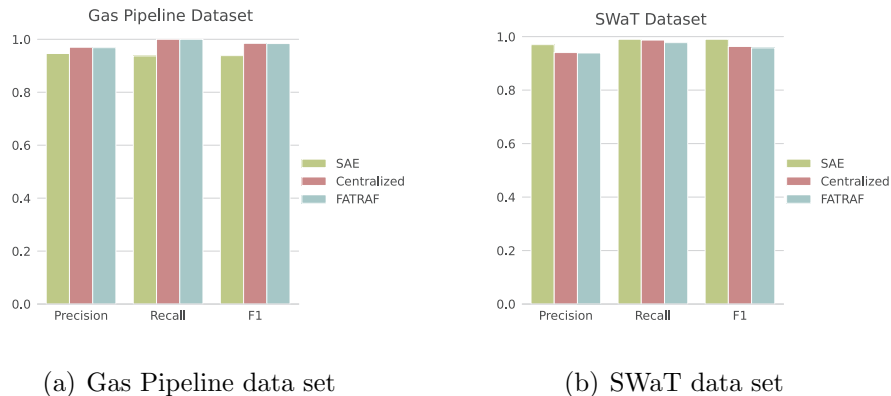


Figure 6: Detection performance of SAE [7] vs. FATRAF vs. ATRAF’s centralized mode on two datasets

570 Another comparison is made between our proposed solution and the mea-
571 surement intrusion detection system (MIDS) introduced in [8], which utilizes
572 3 machine learning techniques such as KNN, Decision Tree, and Random
573 Forest. As can clearly be seen from Table 5, the detection performance of
574 both FATRAF and its centralized mode remarkably outperforms MIDS with
575 all three different detection techniques. One point worth mentioning is that
576 Precision, Recall, and F1-Score values are yielded 0 when using MIDS over
577 the HAI data set. This outcome can be explained as the classification algo-
578 rithms are not able to detect the minority class, which is the anomaly class in
579 the test set. In fact, the original paper [8] presented much higher classifica-
580 tion performance, approximately ideal results, than our shown experimental
581 results. The reason could be that those performance metrics may have been
582 calculated when classifying an imbalanced test set with a micro-averaging
583 strategy which highly encourages the classifier to focus on the dominant
584 class for the trade-off of the minority class. However, this strategy is irrel-
585 evant in the context of binary classification tasks. Also, correctly detecting
586 minority classes should be the priority when dealing with anomaly detection
587 problems.

Table 5: FATRAF and ATRAF’s centralized mode vs. Measurement Intrusion Detection System [8]

Data set		HAI			Gas Pipeline		
		Precision	Recall	F1	Precision	Recall	F1
MIDS [8]	KNN	0.7254	0.1685	0.2735	0.4236	0.6791	0.5218
	Decision Tree	0	0	0	0.5113	0.7076	0.5937
	Random Forest	0	0	0	0.4913	0.7587	0.5964
Centralized ATRAF		0.9039	0.9973	0.9483	0.9699	1	0.9847
FATRAF		0.9404	0.9871	0.9632	0.9683	1	0.9839

588 Finally, Figure 7 shows the detection performance differences of proposed
 589 solutions and Anomaly Transformer [22] which also applies the Transformer
 590 model for time-series anomaly detection tasks. The comparison is drawn
 591 from their performance on the SWaT data set. It can be noted from the
 592 chart that our proposed detection mechanism is slightly better than Anomaly
 593 Transformer whose Precision, Recall and F1-Score metrics are 0.9155, 0.9673,
 594 0.9407, respectively.

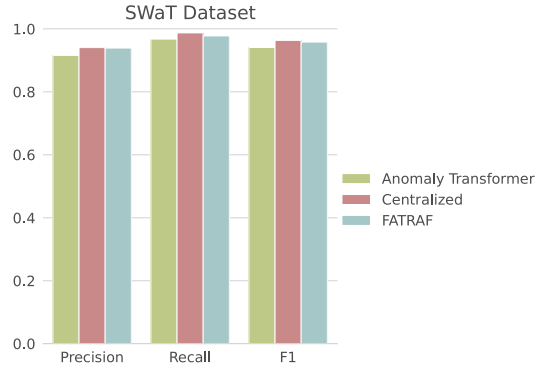


Figure 7: FATRAF and ATRAF’s centralized mode vs. Anomaly Transformer [22]

595 *4.3.2. Edge Computing Evaluation*

596 Since our proposal aims to be deployed on resource-constrained IoT edge
 597 devices, we should validate if FATRAF not only reaches a remarkable perfor-
 598 mance as demonstrated above but also saves edge resources effectively, partly
 599 thanks to its light-weight feature. Consequently, this section dives into eval-
 600 uating the edge computing performance in terms of memory usage, GPU,
 601 CPU usage, power consumption, running time, and bandwidth occupation.

602

Memory usage

In this measurement, we can see how the memory of each edge device is used during the training phase of the FATRAF on the Gas Pipeline data set. As described above, the training phase comprises two continuous phases: local training of Autoencoder and training of Transformer-Fourier in the federated environment. The edge computing assessment is carried out with 10 local epochs of training the autoencoder and 10 communication rounds of training the Transformer-Fourier block. Figure 8 demonstrates the memory usage of an NVIDIA Jetson Nano representing an edge device during the whole operation of 1155 seconds (roughly 19 minutes). As it can be noticed, the autoencoder phase takes about 410 seconds or nearly 7 minutes, consuming around 70% of total memory. After that, the client continues to feed data through the trained encoder to create input for the Transformer-Fourier training phase. For the latter phase, the Jetson client increases its constant memory usage to approximately 75% of its total of 4GB RAM.

It should be noted that the memory space is common between GPU and CPU, there is no dedicated GPU memory in NVIDIA Jetson Nano. Therefore, this memory usage is not for training model purposes only, it is also used by the operating system and other general-purpose programs.

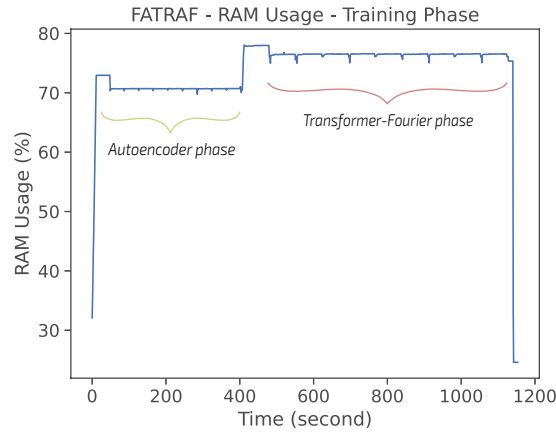


Figure 8: Memory Usage of an Edge device (NVIDIA Jetson Nano) during the FATRAF Training Phase

CPU usage

625 In order to see how CPU is occupied for the training phase of the learning
 626 model, and to analyze how light-weight our proposed solution is, CPU usage
 627 is measured and presented in Figure 9. As the proposed detection mecha-
 628 nism leverages parallel computing and GPU training, the learning model just
 629 makes a spike of around 40% of the CPU usage, but mostly consumes less
 630 than 20% of the hardware CPU during the training phase. This result shows
 631 that our proposed learning model is quite light-weight which is suitable for
 632 being deployed in an edge-computing environment.

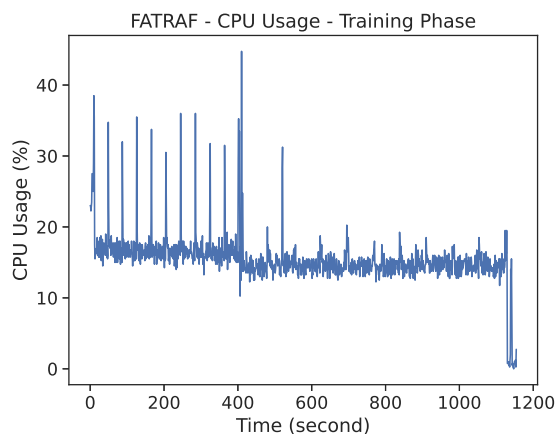


Figure 9: CPU Usage of an Edge device (NVIDIA Jetson Nano) during the FATRAF Training Phase

633 GPU Usage

634

635 The GPU usage of the edge hardware based on NVIDIA Jetson Nano is
 636 illustrated in Figure 10. Since the GPU hardware is primarily responsible
 637 for the training detection model, the GPU usage of Jetson is utilized to its
 638 maximum during the FATRAF operation in order to accelerate the learning
 639 process of both the Autoencoder and Transformer-Fourier block.

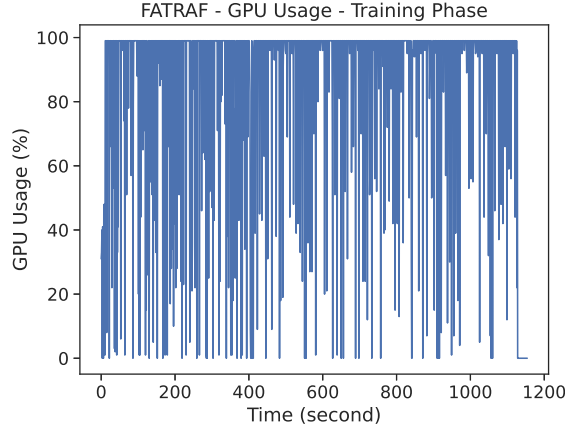


Figure 10: GPU Usage of NVIDIA Jetson Nano during FATRAF Training Phase

640 **Running time evaluation**

641

642 In this experiment, the training time of the FATRAF learning model will
 643 be compared with the FL-VAE-LSTM model proposed in the recent work
 644 [19] of the same field and context.

645 Whilst the settings for the FATRAF operation remains the same as de-
 646 scribed the above measurements, the training of the detection mechanism
 647 FL-VAE-LSTM is configured with 10 communication rounds in the VAE
 648 and LSTM phase each (as the convergence point), on top of the same Gas
 649 Pipeline data set.

650 It is also worth mentioning that both of the detection learning models are
 651 trained on the GPU device of NVIDIA Jetson Nano. As Figure 11 illustrates,
 652 when training the 2 models on the same hardware, the training time of the
 653 hybrid of AE - Transformer - Fourier takes 1200 seconds overall, whilst the
 654 combination of VAE - LSTM needs around 5000 seconds to finish the training
 655 process.

656 In conclusion, in this research, we have achieved our goal of reducing the
 657 training time of the learning model, which, as the results, fits much better
 658 to the IoT environment.

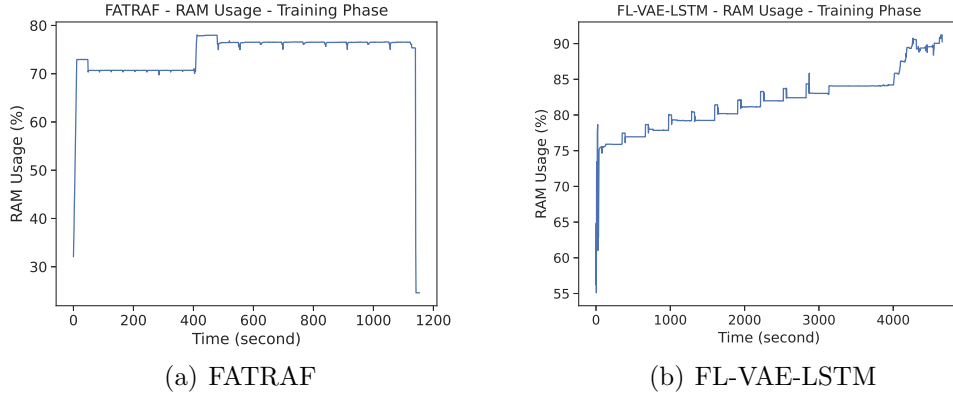


Figure 11: Memory Usage and Training time of FATRAF vs. FL-VAE-LSTM [19]

659 In addition, we extend our experiment to demonstrate the running time
 660 performance of the Hybrid Transformer-Fourier that replaces the attention
 661 sublayer with the Fourier transform sublayer. The Fourier transform is ap-
 662 plied to reduce the training time as well as the complexity, making the mech-
 663 anism more light-weight. As dealing with diverse problems, more Fourier
 664 layers may be employed to enhance the detection performance instead of
 665 traditional transformer encoder layers. To better understand the benefit of
 666 hybrid Transformer-Fourier over all-attention Transformer, Figure 12 illus-
 667 trates the relationship of how training time of these models is increased in
 668 proportion to the number of layers ranging from 2 to 10. The recorded time
 669 involves 10 rounds of training over the Gas Pipeline data set, after the au-
 670 toencoder phase. As the number of layers increases, learning time of the
 671 hybrid Transformer-Fourier model is proven to be more efficient than one of
 672 the all-attention transformer model.

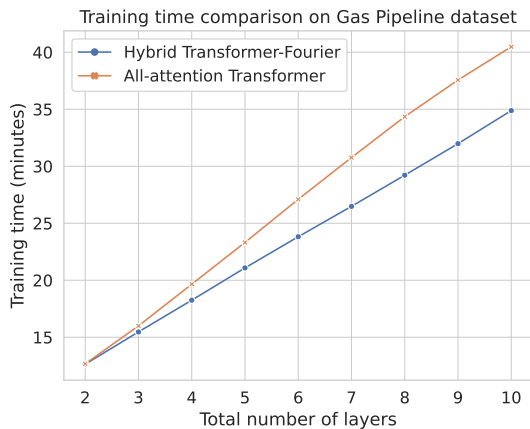


Figure 12: Training time of Hybrid Transformer-Fourier model vs. All-Attention-Sublayer Transformer model

Bandwidth Consumption

673

674

675 Leveraging the Federated Learning technique, in FATRAF, each client
676 sends all learnable parameters of the Transformer-Fourier block to the cloud
677 server in each round. Figure 13 and Figure 14 depict the bandwidth occupied
678 in the upstream and downstream link by each client in the edge-cloud trans-
679 mission. Because the autoencoder training phase is entirely implemented
680 within the local site, the clients do not exchange any information in this
681 phase as illustrated in the first part of the graph. On the contrary, when
682 the client enters the Transformer-Fourier training phase, in every communi-
683 cation round, the model’s weight matrices are transferred to the cloud for
684 federation (i.e., aggregation) when the local training is done. As can be seen
685 in Figure 13 and Figure 14, there are 10 spikes of bandwidth occupation of
686 around 45 KiB/s. Each spike spans no more than 2 seconds and corresponds
687 to one transmission or reception of weight matrices in one round at the edge
688 device. Such low communication cost will spare bandwidth for other infor-
689 mation transmission in Industrial IoT systems, thus increasing the feasibility
690 of the FATRAF architecture.

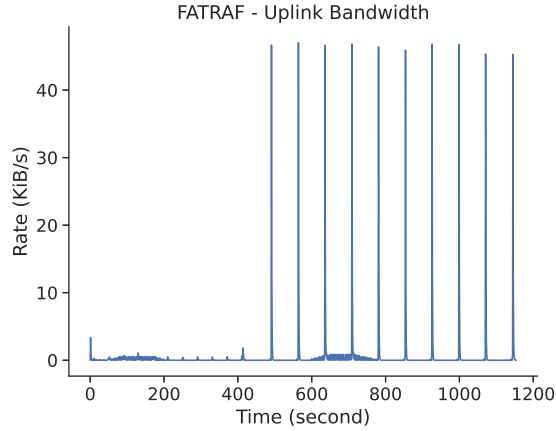


Figure 13: Bandwidth occupation in the Edge-Cloud upstream link of FATRAF

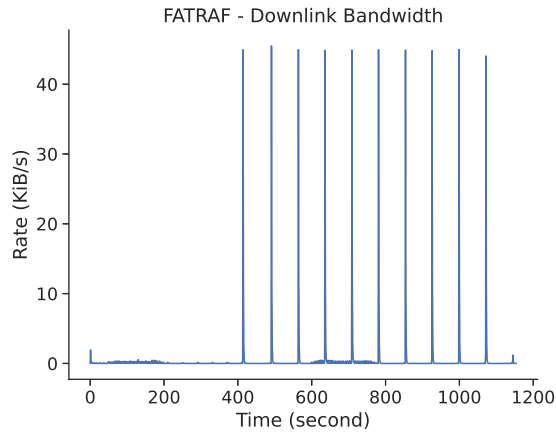


Figure 14: Bandwidth occupation in the Edge-Cloud downstream link of FATRAF

691 **Power Consumption**

692

693 To have a deeper insight on the performance and influence of our proposed
 694 model from the various aspects, power consumption at each edge device dur-
 695 ing the training phase is taken into account. In fact, power consumed in each
 696 electronic component within Industrial systems could also be a critical factor
 697 too, taking into account energy efficiency and keeping the earth green. Fig-
 698 ure 15 shows a sketch of how FATRAF consumes power during each phase in

699 the whole training duration. With an energy consumption baseline of around
 700 1000 mW when the Jetson client is not processing any task, the hardware
 701 extensively consumes around 5000-6000 mW during the autoencoder phase.
 702 Then during the course of training the Transformer-Fourier block, the power
 703 consumption of Jetson Nano declines and remains in the range of 4000-5000
 704 mW until the training process is done.

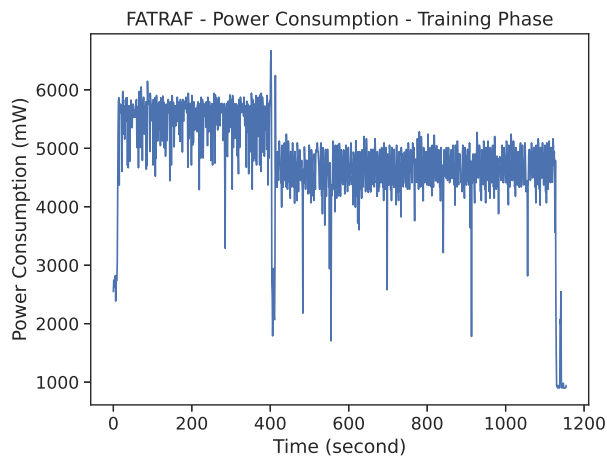


Figure 15: Power Consumption of NVIDIA Jetson Nano during the FATRAF Training Phase

705 5. Conclusion

706 In this paper, we have proposed an anomaly detection method that ap-
 707 plies the Federated Learning technique to make use of both advantages of
 708 distributed learning in different local areas and global updating for all local
 709 learning models - FATRAF. FATRAF has been proven to provide high detec-
 710 tion performance for time-series data in ICSs in comparison with cutting-edge
 711 proposed AD solutions, whilst achieving a remarkable improvement in run-
 712 ning time. The reduction of the training time of the learning model down
 713 to 1200 seconds paves the way for this AD solution to be re-trained more
 714 frequently during the factory operation. In turn, it helps the security archi-
 715 tecture in ICS to be able to frequently update changes in normal behaviors
 716 of the smart devices installed in a smart factory 4.0.

717 **References**

- 718 [1] D. Bhamare, M. Zolanvari, A. Erbad, R. Jain, K. Khan, N. Meskin,
719 Cybersecurity for industrial control systems: A survey, *Computers &*
720 *Security* 89 (2020) 101677.
- 721 [2] A. Sherstinsky, Fundamentals of recurrent neural network (rnn) and long
722 short-term memory (lstm) network, *Physica D: Nonlinear Phenomena*
723 404 (2020) 132306.
- 724 [3] C. Xu, J. Shen, X. Du, F. Zhang, An intrusion detection system using a
725 deep neural network with gated recurrent units, *IEEE Access* 6 (2018)
726 48697–48707.
- 727 [4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez,
728 L. u. Kaiser, I. Polosukhin, Attention is all you need, in: I. Guyon, U. V.
729 Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett
730 (Eds.), *Advances in Neural Information Processing Systems*, volume 30,
731 Curran Associates, Inc., 2017.
- 732 [5] J. Lee-Thorp, J. Ainslie, I. Eckstein, S. Ontanon, FNet: Mixing tokens
733 with fourier transforms (2021).
- 734 [6] S. Sheather, J. Marron, Kernel quantile estimators, *Journal of The*
735 *American Statistical Association - J AMER STATIST ASSN* 85 (1990)
736 410–416.
- 737 [7] A. Al-Abassi, H. Karimipour, A. Dehghantanha, R. M. Parizi, An en-
738 semble deep learning-based cyber-attack detection in industrial control
739 system, *IEEE Access* 8 (2020) 83965–83973.
- 740 [8] S. Mokhtari, A. Abbaspour, K. K. Yen, A. Sargolzaei, A machine learn-
741 ing approach for anomaly detection in industrial control systems based
742 on measurement data, *Electronics* 10 (2021).
- 743 [9] M. Kravchik, A. Shabtai, Efficient cyber attack detection in industrial
744 control systems using lightweight neural networks and pca, *IEEE Trans-*
745 *actions on Dependable and Secure Computing* (2021) 1–1.
- 746 [10] C.-P. Chang, W.-C. Hsu, I. Liao, Anomaly detection for industrial con-
747 trol systems using k-means and convolutional autoencoder, in: 2019

- 748 International Conference on Software, Telecommunications and Com-
749 puter Networks (SoftCOM), pp. 1–6.
- 750 [11] S. D. D. Anton, S. Sinha, H. D. Schotten, Anomaly-based intrusion
751 detection in industrial data with svm and random forests, 2019.
- 752 [12] itrust, centre for research in cyber security, singapore university of
753 technology and design, [https://itrust.sutd.edu.sg/itrust-labs_](https://itrust.sutd.edu.sg/itrust-labs_datasets/)
754 [datasets/](https://itrust.sutd.edu.sg/itrust-labs_datasets/), Accessed: 2022-01-01.
- 755 [13] I. P. Turnipseed, A new scada dataset for intrusion detection research.
- 756 [14] Hil-based augmented ics security dataset, [https://github.com/](https://github.com/icsdataset/hai)
757 [icsdataset/hai](https://github.com/icsdataset/hai), Accessed: 2022-01-01.
- 758 [15] K. Zarzycki, M. Ławryńczuk, Lstm and gru neural networks as models of
759 dynamical processes used in predictive control: A comparison of models
760 developed for two chemical reactors, *Sensors* 21 (2021).
- 761 [16] V. Mothukuri, P. Khare, R. M. Parizi, S. Pouriye, A. Dehghantanha,
762 G. Srivastava, Federated learning-based anomaly detection for iot secu-
763 rity attacks, *IEEE Internet of Things Journal* (2021) 1–1.
- 764 [17] A. L. Perales Gómez, L. Fernández Maimó, A. Huertas Celdrán, F. J.
765 García Clemente, Madics: A methodology for anomaly detection in
766 industrial control systems, *Symmetry* 12 (2020).
- 767 [18] D. Li, D. Chen, L. Shi, B. Jin, J. Goh, S. Ng, MAD-GAN: multivari-
768 ate anomaly detection for time series data with generative adversarial
769 networks, *CoRR* abs/1901.04997 (2019).
- 770 [19] T. T. Huong, T. P. Bac, D. M. Long, T. D. Luong, N. M. Dan, L. A.
771 Quang, L. T. Cong, B. D. Thang, K. P. Tran, Detecting cyberattacks
772 using anomaly detection in industrial control systems: A federated learn-
773 ing approach, *Computers in Industry* 132 (2021) 103509.
- 774 [20] Y. Liu, S. Garg, J. Nie, Y. Zhang, Z. Xiong, J. Kang, M. S. Hos-
775 sain, Deep anomaly detection for time-series data in industrial iot: A
776 communication-efficient on-device federated learning approach, *IEEE*
777 *Internet of Things Journal* 8 (2021) 6348–6358.

- 778 [21] H. Meng, Y. Zhang, Y. Li, H. Zhao, Spacecraft anomaly detection via
779 transformer reconstruction error.
- 780 [22] J. Xu, H. Wu, J. Wang, M. Long, Anomaly transformer: Time series
781 anomaly detection with association discrepancy, 2021.
- 782 [23] R. Kozik, M. Pawlicki, M. Choraś, A new method of hybrid time win-
783 dows embedding with transformer-based traffic data classification in iot-
784 networked environment, *Pattern Analysis and Applications* 24 (2021).
- 785 [24] S. Garcia, A. Parmisano, M. J. Erquiaga, IoT-23: A labeled dataset
786 with malicious and benign IoT network traffic, 2020. More details here
787 [https://www.stratosphereips.org /datasets-iot23](https://www.stratosphereips.org/datasets-iot23).
- 788 [25] H. McMahan, E. Moore, D. Ramage, B. Agüera y Arcas, Federated
789 learning of deep networks using model averaging (2016).
- 790 [26] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, A. T. Suresh,
791 SCAFFOLD: Stochastic controlled averaging for federated learning, in:
792 H. D. III, A. Singh (Eds.), *Proceedings of the 37th International Con-
793 ference on Machine Learning*, volume 119 of *Proceedings of Machine
794 Learning Research*, PMLR, 2020, pp. 5132–5143.
- 795 [27] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, V. Smith,
796 Federated optimization in heterogeneous networks, 2020.
- 797 [28] E. Keogh, J. Lin, A. Fu, Hot sax: efficiently finding the most unusual
798 time series subsequence, in: *Fifth IEEE International Conference on
799 Data Mining (ICDM'05)*, pp. 8 pp.–.
- 800 [29] Nyc taxi and limousine commission, Last accessed on May, 2021.
- 801 [30] J. Shao, B. Zhong, Last observation carry-forward and last observation
802 analysis, *Statistics in Medicine* 22 (2003) 2429–2441.
- 803 [31] bmon, <https://github.com/tgraf/bmon>, Last accessed on May, 2021.
- 804 [32] C. He, S. Li, J. So, M. Zhang, H. Wang, X. Wang, P. Vepakomma,
805 A. Singh, H. Qiu, L. Shen, P. Zhao, Y. Kang, Y. Liu, R. Raskar, Q. Yang,
806 M. Annavaram, S. Avestimehr, Fedml: A research library and bench-
807 mark for federated machine learning, arXiv preprint arXiv:2007.13518
808 (2020).

- 809 [33] H. Xu, W. Chen, N. Zhao, Z. Li, J. Bu, Z. Li, Y. Liu, Y. Zhao, D. Pei,
810 Y. Feng, J. Chen, Z. Wang, H. Qiao, Unsupervised anomaly detection
811 via variational auto-encoder for seasonal kpis in web applications, in:
812 Proceedings of the 2018 World Wide Web Conference, WWW '18, In-
813 ternational World Wide Web Conferences Steering Committee, Republic
814 and Canton of Geneva, CHE, 2018, p. 187–196.

Declaration of interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: