IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 20 December 2022, accepted 11 January 2023, date of publication 16 January 2023, date of current version 24 January 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3237554

APPLIED RESEARCH

Distributed Anomaly Detection in Smart Grids: A Federated Learning-Based Approach

J. JITHISH, (Member, IEEE), BITHIN ALANGOT, NAGARAJAN MAHALINGAM, (Senior Member, IEEE), AND KIAT SENG YEO, (Fellow, IEEE)

Engineering Product Development Pillar, Singapore University of Technology and Design, Singapore 487372

Corresponding author: J. Jithish (jithish_jayarajan@sutd.edu.sg)

This work was supported by the Singapore's National Research Foundation, under Award NRF-CRP20-2017-0006.

ABSTRACT The smart grid integrates Information and Communication Technologies (ICT) into the traditional power grid to manage the generation, distribution, and consumption of electrical energy. Despite its many advantages, it faces significant challenges, such as detecting abnormal behaviours in the grid. Identifying anomalous behaviours helps to discover unusual user power consumption, faulty infrastructure, power outages, equipment failures, energy thefts, or cyberattacks. Machine learning (ML)-based techniques on smart meter data has shown remarkable results in anomaly detection. However, traditional ML-based anomaly detection requires smart meters to share local data with a central server, which raises concerns regarding data security and user privacy. Server-based model training faces additional challenges, such as the requirement of centralised computing power, reliable network communication, large bandwidth capacity, and latency issues, all of which affect the real-time anomaly detection performance. Motivated by these concerns, we propose a Federated Learning (FL)-based smart grid anomaly detection scheme where ML models are trained locally in smart meters without sharing data with a central server, thus ensuring user privacy. In the proposed approach, a global model is downloaded from the server to smart meters for on-device training. After local training, local model parameters are sent to the server to improve the global model. We secure the model parameter updates from adversaries using the SSL/TLS protocol. Using standard datasets, we investigate the anomaly detection performance of federated learning and observe that FL models achieve anomaly detection performance comparable to centralised ML models while ensuring user privacy. Further, our study shows that the proposed FL-based models perform efficiently in terms of memory, CPU usage, bandwidth and power consumption at edge devices and are suitable for implementation in resource-constrained environments, such as smart meters, for anomaly detection.

INDEX TERMS Anomaly detection, cybersecurity, data analysis, federated learning, Internet of Things, machine learning, neural networks, smart grid, smart meter.

I. INTRODUCTION

The traditional power grid suffers from several disadvantages, such as ageing infrastructure, reliability issues, lack of consumer participation in power distribution, and limited support for distributed energy sources [1], [2], [3]. The Smart grid represents the next stage in the power grid development that integrates technological advancements such as artificial intelligence, big data, cloud computing, and 5G cellular technology to manage the generation, distribution, and consumption of electrical energy [4]. Smart grids use Advanced Metering Infrastructure (AMI) or smart meters that collect user data to monitor power flows and adapt to energy demand and supply variations accordingly. Smart meters facilitate additional benefits, such as rapid outage detection, faster service restoration capabilities, and greater control over billing by providing detailed information on power usage, thus enabling customers to make informed decisions [5], [6].

Despite its significant benefits, smart grids still face many technical challenges [7], [8]. Smart grids are susceptible

The associate editor coordinating the review of this manuscript and approving it for publication was Mostafa Rahimi Azghadi.

to various vulnerabilities, such as faulty equipment, outages, equipment failures, energy thefts, and cyberattacks, which cause Non-Technical Losses (NTL) [9]. In particular, adversaries may exploit the security vulnerabilities in the grid to launch sophisticated cyber attacks (interrupt services, damage infrastructure, and steal user data) [10] that may affect the normal functioning of the grid. According to a recent study, NTLs cause significant revenue losses globally, which amounts to 96 billion dollars every year [11]. NTLs can be identified by detecting abnormalities in the grid, which gets reflected in the data collected by smart meters.

In conventional grids, abnormal usage patterns are usually identified by regular analysis of utility bills or onsite inspections, both of which are time-consuming and inefficient. Smart meters enable the large-scale collection of customer data by frequent sampling, which results in fine-grained data available for data analytics. Since the energy consumption data is readily available in smart meters, identifying anomalous usage patterns can be performed automatically using Machine Learning (ML) models. ML-based techniques for anomaly detection can identify energy-theft attacks, abnormal electricity consumption behaviours, faulty smart meters, malfunctioning appliances, and NTLs, such as meter tampering and meter modification [12], [13]. Anomaly detection techniques can assist utility companies in minimising energy wastage, reducing power imbalance, preventing power outages, and identifying malicious user activities.

ML-based anomaly detection techniques use historical power consumption data from smart meters to train ML models. Traditionally, there are two major approaches for building smart meter anomaly detection models. The first approach builds dedicated ML models for each smart meter in the central server by collecting data from individual meters. The second approach aggregates data from smart meters to build a single global ML model to detect anomalies in smart meters. Both methods involve transferring data to a centralised server for model training, contributing to substantial network traffic. In addition, centralised solutions suffer from the following disadvantages

- **Connectivity**– Centralised ML solutions require stable connection to continuously transmit data to the server. Since IoT devices are often deployed in remote environments, maintaining a stable internet connection could be challenging.
- **Bandwidth** – When there are thousands of IoT devices, participating in the ML task, the bandwidth required to transmit data to the centralised server may be prohibitively large.
- **Latency** -Transmitting data to server, running ML algorithms in the cloud, retrieving ML predictions lead to high latency affecting real-time applications
- **Privacy and Security**- Sending data to server makes it vulnerable to cyberattacks and may violate local privacy regulations such as California Consumer Privacy Act (CCPA) and General Data Protection Regulation (GDPR) [14].

To address these concerns, in this work, we propose a novel distributed strategy for anomaly detection in smart grids using Federated Learning (FL), which has comparable anomaly detection accuracy to centralised models [15]. Since bulk of the computation in FL happens in the local device, the impact of connectivity, bandwidth and latency issues in FL is reduced compared to centralised training. In FL, ML models are trained in a decentralised manner, where a copy of the global ML model is sent to the clients, which gets trained on clients using the local dataset. After training is completed, each client sends only the updated model parameters to the central server, which are then incorporated into the original model. Since data for model training stays on client devices, this approach ensures data privacy. Furthermore, FL reduces network traffic as only incremental model parameter updates are transmitted to the central server instead of sending large volumes of data from client devices.

In this work, we investigate the anomaly detection performance of federated learning models. We develop seven ML models towards this goal: Logistic Regression, Feed-forward Neural Network Classifier, 1D-CNN Binary Classifier, Autoencoder Binary Classifier, Vanilla RNN Classifier, LSTM Binary Classifier, and GRU Binary Classifier. These models are evaluated on three standard anomaly detection datasets for 100 clients using the Tensorflow Federated framework [16]. Further, we train these models on the same datasets using conventional centralised training. In addition, the anomaly detection performance of federated models is compared with their corresponding centralised models.

Next, we use the developed FL models for identifying anomalous energy consumption in smart grids. Similar to the anomaly detection datasets, we train these models on a server in the traditional centralised manner. Then, we federate the proposed ML models using Raspberry Pi as a prototype for smart meter hardware. Using a workstation as server and Raspberry Pi as client devices, we set up a federated learning environment to detect anomalous energy consumption patterns in publicly available smart meter datasets. After completing the FL training process, we evaluate the average performance of the FL-trained models with that of their corresponding centrally-trained ones.

Further, we investigate the feasibility of deploying the proposed FL models for anomaly detection in terms of their memory, CPU usage, bandwidth, and power consumption at edge devices. Hardware evaluation of FL models is critical as detection algorithms are required to identify anomalies while being lightweight enough to be deployed in resource-constrained devices with limited processing capabilities. We observe that federate anomaly detection models operate efficiently in terms of memory usage, CPU usage, and power consumption on edge hardware with accuracy comparable to server-trained models and are suitable for deployment in smart meters.

The following are the contributions of the paper:

- We investigate the anomaly detection performance of federated learning by evaluating FL models on standard anomaly detection datasets.
- We compare the anomaly detection performance of federated models with corresponding centralised models for standard datasets.
- We develop novel ML models for smart grid anomaly detection using centralised training and evaluate their performance on a real-world smart meter dataset.
- We develop a federated learning testbed for smart grids by prototyping smart meters using a Raspberry Pi device. We perform FL-based on-device anomaly detection without data sharing using the prototype devices as clients and a workstation as the server.
- Finally, we investigate the resource utilisation, memory, CPU, power, and bandwidth requirements for training FL models at the smart meter prototype.

The following are the key novelties of our work:

- We compare the anomaly performance of federated and centralised ML models for KDD, NSL-KDD and CIDDS datasets. To the best of our knowledge, this is the first work that compares the anomaly detection performance of centralised and federated ML models for different datasets.
- Another part of the novelty of this work is that, so far, this is first work to apply federated ML models for anomaly detection in a smart grid context.
- To authors' knowledge, ours is the first research that quantifies the resource usage of federated models for potential application in resource-constrained IoT devices.
- Finally, we perform the performance evaluation on federated models for smart grid anomaly detection on real hardware using smart meter prototypes. We believe this has not been performed previously.

The rest of the paper is organized as follows: Section II explains the related work, Section III explains the federated learning process for anomaly detection, Section IV describes federated learning for anomaly detection datasets, Section V discuss federated anomaly detection for smart grids, Section VI explains the findings of our study, and finally Section VII discusses the conclusion and future work.

II. RELATED WORK

This section examines the recent research articles that use federated learning in different applications across various domains. We also provide a comprehensive overview of the recent works that use machine learning for anomaly detection in smart grids. Finally, we discuss the latest research works on FL-based anomaly detection. Table 1 gives the comparison of our proposed work with existing works.

A. FEDERATED LEARNING APPLICATIONS

FL is an emerging discipline that has shown promising results in various applications such as smartphones, smart homes, image processing, health care, and electric vehicles. Hard et al. [17] proposed a next-word predictor for smartphone keyboards that used Recurrent Neural Networks (RNN). The RNN model was trained using FL with the participation of client smartphones. The model achieved significant improvements in next-word prediction but was constrained by high communication costs. Leroy et al. [18] proposed a smartphone voice assistant based on Convolutional Neural Networks (CNN) trained using FL. The model was successfully used for wake-word detection. Ramaswamy et al. [19] used FL to train a Long Short Term Memory (LSTM) based model for predicting emojis on smartphone keyboards. The model achieved better performance compared to server-trained models.

In image processing, Han et al. [20] proposed an FL-based framework for product defect detection in factories. The proposed framework used CNN for identifying faulty products. Silva et al. [21] performed the analysis of MRI images from multiple hospitals for brain disorders in a federated setting. The authors demonstrated the importance of FL in medical image analysis, where privacy and security are of the foremost concern. Feki et al. [22] proposed a CNN-based FL for COVID-19 screening using chest X-Ray images. In the proposed approach, multiple hospitals collaboratively trained a CNN model without transferring their private data. Lee et al. [23] used FL for analysis of thyroid ultrasound images. The performance of the FL approach was found to be comparable with centralised training with enhanced privacy and significant savings in bandwidth.

In clinical research, Pfohl et al. [24] used FL on patient records from multiple hospitals for clinical risk prediction with guaranteed patient privacy. The performance of the FL model was found to be comparable to the centralised model. Brisimi et al. [25] utilised FL to train a Support Vector Machine (SVM) classifier to predict cardiovascular disease hospitalisations. The FL model converged faster with less communication overhead than the centralised model. Sanchez et al. [26] used FL to train a CNN model for breast cancer detection. The authors observed better classification performance for the FL model compared to the centralised version. Tedeschini et al. [27] propose FL for brain tumour segmentation. The authors implemented the proposed approach on a real-time test bed with high training accuracy.

In other applications, Mowla et al. [28] utilised FL to detect jamming attacks in Flying Ad Hoc Networks (FANETs) formed by Unmanned Aerial Vehicles (UAVs). Saputra et al. [29] proposed predicting the energy demand of electric vehicles using a novel clustering algorithm trained using FL. Yu et al. [30] successfully applied FL for obstacle avoidance in autonomous mobile robots.

B. ML-BASED SMART GRID ANOMALY DETECTION

This section describes recent works anomaly detection techniques based on machine-learning for smart grids. In the

TABLE 1. Comparison with existing works.

Article	Datasets	ML Algorithms	FL Framework	ML Framework	Hardware	Simulation	Domain
Troung et al. (2022) [31]	Gas pipeline, SWaT, HAI, Power Demand	Autoencoder	FedML	Pytorch	NVDIA Jetson nano	✗	Industrial Control Systems
Cui et al. (2022) [32]	KDD cup 1999	Generative Adversarial Network	Custom Framework	Not Specified	Raspberry Pi	✗	IoT
Wang et al. (2022) [33]	Not specified	Deep Reinforcement Learning	Custom Framework	Not Specified	✗	✓	Industrial IoT
Huong et al. (2021) [34]	Gas pipeline, ECG, Space shuttle, NYC taxi Power Demand	Autoencoder	FedML	Pytorch	Raspberry Pi	✗	Industrial Control Systems
Liu et al. (2021) [35]	Space shuttle, ECG, Engine, Power Demand	Long Short Term Memory	PySyft	Pytorch	✗	✓	Industrial IoT
Mothukuri et al. (2021) [36]	Modbus dataset	Long Short Term Memory, Gateway Recurrent Unit	PySyft	Pytorch	✗	✓	Industrial Control Systems
Sater et al. (2021) [37]	Sensor event log dataset, Energy usage dataset	Long Short Term Memory	PySyft	Pytorch	✗	✓	Smart Buildings
Nguyen et al. (2019) [38]	Custom dataset	Gateway Recurrent Unit	Custom Framework	Tensorflow	✗	✓	IoT
This work	Ausgrid Dataset	Logistic Regression, Feed-Forward Neural Network Classifier, 1D-CNN Binary Classifier, Autoencoder Binary Classifier	Flower	Tensorflow	Raspberry Pi	✗	Smart Grid Anomaly Detection

following subsections, we categorise these works based on the types of machine techniques used.

1) UNSUPERVISED TECHNIQUES

Clustering, dimensionality reduction, and one-class learning are the major unsupervised techniques widely used for smart grid anomaly detection. In clustering-based techniques, Yeckle et al. [39] successfully investigated the detection of energy theft in smart grids using several unsupervised algorithms such as K-Nearest Neighbour (KNN), Local Outlier Factor (LOF), and Local Density Factor (LDF). Rossi et al. [40] used categorical clustering to detect anomalous customer behaviour from the smart grid dataset. The proposed approach provided a visual representation of anomalies in the smart grid. Arjunan et al. [41] proposed a novel algorithm for detecting anomalous smart grid power consumption using anomaly scores. The anomaly score of users is periodically updated to identify abnormal behaviour. Toshpulatov et al. [42] successfully used Hierarchical Self-Organising Maps (HSOM) for detecting anomalies in smart grids with an average *F1-score* exceeding .9%.

Among anomaly detection techniques based on One-Class Support Vector Machines (OCSVM), Himeur et al. [43] introduced an OCSVM-based anomaly detection approach to identify abnormal energy usage in buildings. However, the proposed approach has a limitation as it does not account for information about user occupancy. Harrou et al. [44] used OCSVM to identify anomalies in data generated from photovoltaic arrays. The authors reported that the proposed OCSVM-based approach yielded higher accuracy than other clustering-based techniques.

Among dimensionality reduction-based techniques, Sial et al. [45] proposed Principal Component Analysis (PCA) and clustering to identify abnormal energy consumption from normal behaviour. Anomalies were calculated based on an anomaly consumption score. Zhang et al. [46] utilised Gaussian Mixture Model (GMM) and Linear Discriminant Analysis (LDA) for detecting abnormal power consumption. The evaluation of the proposed GMM-LDA approach showed higher accuracy than SVM-based approaches.

2) SUPERVISED TECHNIQUES

Among supervised ML techniques, Neural Networks (NN) have been gaining popularity recently for anomaly detection. Abnormal energy consumption patterns in industrial plants can be used as a metric for the early identification of plant

degradation. Santolamazza et al. [47] used NNs to identify system failure using anomaly detection from abnormal power usage. Zheng et. al [48] used CNNs for detecting energy-thefts in smart grids. The proposed approach combines both wide and deep CNNs for electricity theft detection. Evaluation of the approach on realistic datasets showed that it outperforms existing methods. Zhou et al. [49] developed a real-time anomaly detector for smart grids using LSTM network. The proposed approach detected grid anomalies by analysing voltage measurements.

Wang et al. [50] used LSTM neural network for identifying abnormal smart grid power consumption. The proposed model significantly reduced forecasting error compared to the ARIMA (Autoregressive Integrated Moving Average) algorithm. Pereira et al. [51] introduced an unsupervised framework based on Variational Recurrent Autoencoder (VRAE) time series anomaly detection on data collected from solar panels. The evaluation of solar energy generation data showed that the model detected anomalies with high accuracy. Fenza et al. [52] introduced an LSTM-based power anomaly detection technique that coniders the phenomenon of *concept drift*. The proposed technique profiled user behaviour based on their historical power consumption.

Among traditional supervised techniques, Fahim et al. [53] used support vector regression to detect abnormal energy consumption patterns in smart buildings. Saqaeeyan et al. [54] used Bayesian Networks to detect abnormal occupant behaviours in smart homes. Chen et al. [55] proposed an approach based on the Autoregressive Conditional Heteroscedastic Model (ARCH) for abnormal energy usage detection in smart homes. The proposed approach is validated using real-world building energy consumption datasets. Korba et al. [56] developed an SVM-based approach for detecting anomalous power consumption patterns in smart meters. Cody et al. [57] utilised decision trees to detect fraudulent activities by analysing smart meter datasets.

3) ENSEMBLE METHODS

In ensemble learning, the dataset is split into multiple subsets, and various ML models are applied to these subsets to identify anomalies. Consequently, the anomaly detection scores of these models are calculated. The score of the best-performing model is selected as the final score of the ensemble model. Touzani et al. [58] used gradient boosting to model the baseline energy consumption of smart buildings. This baseline was used to detect abnormal power consumption. De Guia et al. [59] used bagging for anomaly detection in photovoltaic systems. Liang et al. [60] introduced a novel anomaly detection framework for power grids. The proposed framework identifies abnormal usage patterns by analysing the communication protocol of smart meters using an ensemble of SVM and kNN algorithms.

4) HYBRID METHODS

Hybrid methods combine multiple ML algorithms to improve anomaly detection performance. Wang et al. [61] proposed a rule-based anomaly detector that combined SVM and KNN methods for anomaly detection in residential buildings. The authors reported that the proposed approach demonstrated better predictive accuracy than existing works. Chahla et al. [62] developed a hybrid technique that combined LSTM and K-means algorithms to detect unusual behaviour in a real-world power consumption dataset. Takkidin et al. [63] designed an attack detector for electricity thefts. The proposed approach used a combination of an autoencoder and LSTM to train the ML model on benign data. Electricity thefts were detected by analysing the deviation from the normal pattern.

C. ANOMALY DETECTION USING FEDERATED LEARNING

Recent research has demonstrated the promising prospect of FL as a privacy-preserving alternative to centralised model training in various domains. This section highlights the latest works that use FL for anomaly detection across various applications.

Nguyen et al. [38] used federated learning for network intrusion detection in IoT devices. The proposed approach was successfully implemented in a smart home setting with a higher attack detection rate and faster response time. Sater et al. [37] used an LSTM-based FL model for activity prediction and anomalous energy usage detection in smart buildings. The proposed FL-based method performed better than centralised methods with faster training convergence. Mothukuri et al. [36] used federated learning to identify and classify attacks in IoT networks. The authors developed a federated Gated Recurrent Unit-based approach with outperformed centralised ML algorithms. Liu et al. [35] developed an FL-based anomaly detection scheme for time-series IoT data. Further, the authors developed a communication-efficient compression algorithm for FL gradients to reduce network utilisation. Huong et al. [34] developed an FL-based, lightweight anomaly detection framework based on autoencoder architecture for industrial IoT. The scheme detected anomalies in time series data for IIoT with minimal CPU and memory usage. Cui et al. [32] introduced an FL framework for anomaly detection in IoT that used differential privacy to enhance user privacy. Further, the proposed framework used blockchain technology to secure FL algorithms against model poisoning attacks. Troung et al. [31] proposed a fast, FL framework for identifying anomalies in industrial control systems which is both memory and computation efficient. The authors reported that the proposed approach outperformed other anomaly detection solutions. Pei et al. [64] proposed an FL-based anomaly detection framework using LSTM for network traffic anomaly detection. The proposed method produced higher detection accuracy compared to existing methods. Guo et al. [65] developed an FL-based framework called FLOG for anomaly detection in distributed log data. Based on 1D-convolution, the authors proposed an FL model named FLOGCNN, which outperformed baseline methods in anomaly detection.

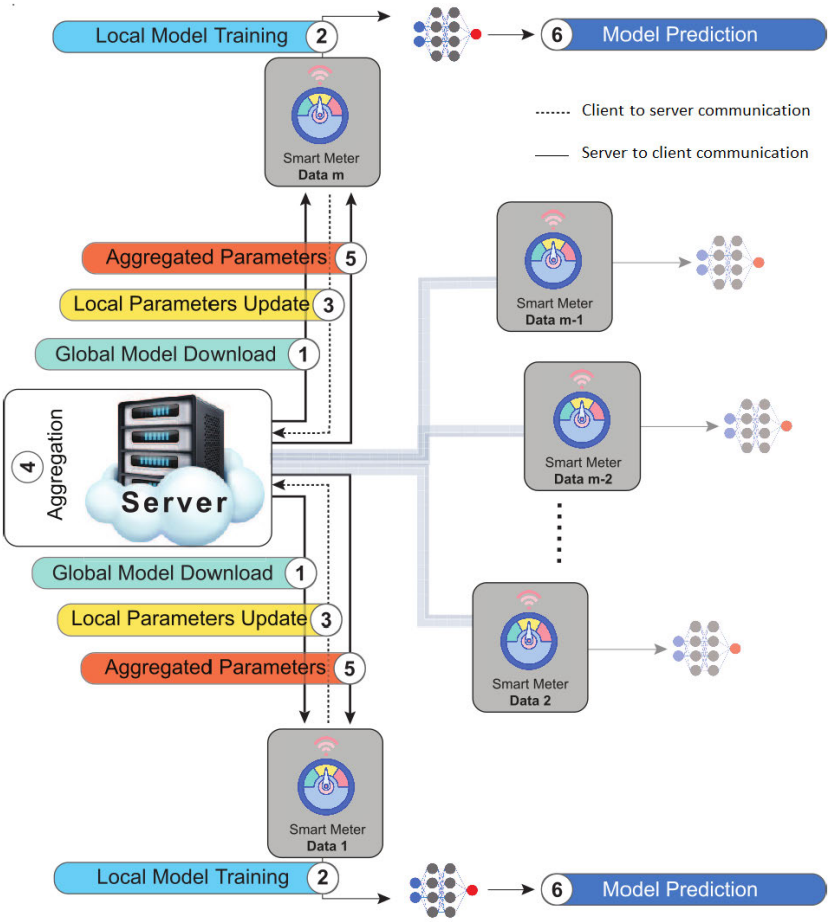


FIGURE 1. Anomaly detection on smart meter data using federated learning.

D. PRIVACY IN FEDERATED LEARNING

As model training is performed in the device on local data, FL inherently provides privacy guarantees compared to centralised training. Upon closer inspection, it becomes evident that some model parameters sent to the server strongly correlate with training data. Therefore, this scenario of inadvertent information leakage may be exploited by adversaries to infer information regarding training data, thus making FL susceptible to inference attacks [66]. Subsequently, it is necessary to further strengthen the privacy and security of FL by employing other techniques. Motivated by these concerns, we use the secure SSL/TLS communication protocol to prevent adversaries from eavesdropping or tampering with model parameter updates sent from client devices to the server.

Among other techniques to enhance privacy in FL, Truex et al. [67] present a privacy-preserving FL system using local differential privacy with formal privacy guarantees. Further, Truex et al. [68] introduce an approach that utilises differential privacy and secure multi-party (SMC) computation. SMC is at risk of inference attacks privacy may not offer good accuracy. The proposed approach balances both techniques and produces highly accurate models resilient against inference attacks. Xu et al. [69] introduce *HybridAlpha*, a privacy-preserving FL framework that uses SMC and functional encryption. The proposed approach is faster and reduces network traffic and model training time compared to existing methods. Zhao et al. [70] propose a blockchain-based FL framework for IoT for predicting customers' requirements and behaviours in smart homes. In addition, blockchain is used to trace the activities of malicious customers.

III. FEDERATED LEARNING FOR ANOMALY DETECTION

Traditional ML-based anomaly detection techniques collect energy consumption data from individual smart meters in a centralised facility such as a server. The ML model is trained for anomaly detection using collected data. In traditional ML, a global machine learning model M is trained by N number of smart meters (SM) represented by $SM_1,\ldots, SM_N$ by aggregating their local data $D_1,\ldots, D_N$ in a server S. The local smart meter data are aggregated to a form global dataset D such that $D = D_1\ U\ D_2\ U,\ldots, D_N$. The dataset D is finally used for global model training.

In contrast, in FL-based anomaly detection, the server dispatches a copy of the global ML model to smart meters,

which get trained using the local smart meter dataset. Here, smart meters transfer only the updated parameters of the global model to the central server, and the data remains localised in smart meters. In FL, smart meters $SM_1,\ldots,SM_N$ collaboratively train global model M without them sharing their respective data $D_1,\ldots,D_N$ to a centralised server S. Each smart meter SM_i trains the global model M using their local data D_i such that only the parameter updates of model M are sent to the central server S.

A. FEDERATED LEARNING TRAINING

Figure 1 describes the FL training process for anomaly detection. As described in the figure, a single global ML model is transferred from the central server to smart meters. In this work, we develop seven global models: Logistic Regression, Feed-Forward Neural Network Classifier, 1D-CNN Binary Classifier, Autoencoder Binary classifier, RNN Classifier, LSTM Binary Classifier, and GRU Classifier for federated anomaly detection in smart grids. Each model is trained by each smart meter using its local dataset. The FL process for one round of training proceeds through the following steps.

- **Step 1:** In the first round of training, the weights of the global model are initialised randomly in the server. Out of a total of N smart meters $(SM_1,\ldots,SM_n)$, a subset of devices, say K, are chosen at random. A copy of the global model M is sent from the server S to K devices.
- **Step 2:** Smart meters train this model using the local smart meter data after receiving the copy of the global model M. Before model training, the local data are preprocessed.
- **Step 3:** Once the local FL training is completed, the K smart meters that participate in training send new model parameters of the trained model to the server S. Since the local data in each device is unique, the model parameters sent to the server are different.
- **Step 4:** Using the model updates sent by smart meters, the server S constructs an improved version of the original global model by aggregating the updated model parameters. The server uses the **FedAVG** algorithm to aggregate the model updates. The process is repeated from Step 1 to Step 4 until model convergence.
- **Step 5:** After model convergence, the updated global model is sent to all smart meters.
- **Step 6:** Smart meters replace their outdated local model with the updated global model. This updated model is now used for detecting anomalous instances in smart meter data.

Centralised model training assumes that the data and model prediction are Independent and Identically distributed (IID). FL does not adhere to this assumption as the data generated in FL corresponds to different users with unique behavioural patterns. Thus the dataset is non-IID, which leads to differences in the statistical distribution of datasets among each smart meter. The violation of IID assumption in FL may impose challenges for FL training. However, we show in the evaluation section that FL achieves comparable accuracy to centralised training for anomaly detection.

Algorithm 1 Federated Averaging Algorithm (FedAvg)

At the Server Side: Initialize weights ω_0 of the global model M
for *each round t* **do**
  $S_t \leftarrow$ randomly selected m clients
  Send model M to S_t clients
  for *each client k* **do**
    $\omega_{t+1}^{k} \leftarrow$ Update client (w_t, k)
    $\omega_{t+1} \leftarrow \sum_{m=1}^{M} \frac{n_m}{n} L_m(\omega)$
  end
  Send model M to all clients
  At the Client Side:
  ClientUpdate(k, w_t) **procedure**
  $B \leftarrow$ Split P_k into batches B of size b_S
  for *each epoch* $e < E$ **do**
    for *batch* $b \in B$ **do**
      $\omega \leftarrow \omega - \eta \Delta L(\omega)$
    end
    send ω to server
  end
end

B. FEDERATED LEARNING ALGORITHM

The federated learning problem of M client devices can be formulated mathematically as an optimization problem as follows

$$\min_{w} L(\omega) = \sum_{m=1}^{M} \frac{n_m}{n} L_m(\omega) \tag{1}$$

where,

$$L_m(\omega) = \frac{1}{n_m} \sum_{i \epsilon P_m} L_i(\omega) \tag{2}$$

In the above equations (1) and (2), $L(\omega)$ represents the loss function of the global model whereas $L_m(\omega)$ denotes the loss function of m^{th} device. The term $L_i(\omega)$ represents the loss of i^{th} data sample. In addition, the term P_m denotes the data partitions in the client device m. $\sum_{m=1}^{M} n_m$ represents the aggregate data from all devices. The objective of the FL algorithms is to find ω which minimizes $L(\omega)$ over P such that for 2 devices i and j, $P_i \neq P_j$.

1) FEDERATED AVERAGING (FedAVG)

The FedAVG process is described by Algorithm 1. The process begins by selecting a random subset of client devices

(denoted as S_t) and sending the global ML model to the selected devices (lines 3 and 4). The selected devices train the received ML model with their local dataset for multiple epochs. The local data in client devices (denoted by P_k) are split into B batches, each of size s_b (line 12). Lines 14 to 16 describe the local training of ML models in client devices. After the training, the client devices send the newly updated weights to the server. The server then computes the average of the received weights from the client devices. The process is repeated until model convergence, after which it is transferred to client devices.

C. SSL FOR FEDERATED LEARNING

The ML model parameters exchanged between client devices and the central server is confidential, but the client-server communication channel is insecure and vulnerable to cyberattacks. The data exchanged in the channel is unencrypted and susceptible to eavesdropping, Man-in-the-middle (MiTM) attacks, and data tampering. To secure the FL training process against cyberattacks, we use SSL/TLS protocol.

The SSL/TLS protocol provides the confidentiality, integrity, and authentication of data transferred between devices. SSL/TLS encrypts data being transmitted between devices to guarantee data privacy. SSL starts a handshake process for device authentication to ensure that interacting device are confident of one other's identities. The protocol digitally signs data to maintain data integrity and prevent data from being changed while transmitted to the destination. Using SSL/TLS to secure the FL training process has various benefits, including

- **Security:** SSL/TLS protects sensitive communication by encrypting the model updates between the server and client devices such that only intended recipients can decrypt the data.
- **Authentication:** The model updates from the client devices must be verified such that they reach only the intended server. SSL/TLS ensures this in our FL training by using server certificates for authentication.
- **Man-in-the-Middle (MiTM) Attacks:** In MiTM attacks, the attacker positions himself between the server and client devices and secretly relays data to impersonate or eavesdrop on the communication. SSL prevents MiTM attacks using certificates.
- **Phishing:** In phishing, malicious actors impersonate the server and may send unauthorised messages to client devices. Since SSL/TLS ensures authentication, the FL training process is secure against phishing attacks.
- **Data Validation:** SSL/TLS ensures data validation through the process of handshaking. The data exchanges between server and client devices are validated. If data validation is not successful, the operations are aborted.

In this work, we implement SSL/TLS using the OpenSSL library. Specifically, we used the openssl version 1.1.1n released in March 2022. This version of openssl implements SSL/TLS version 1.3.

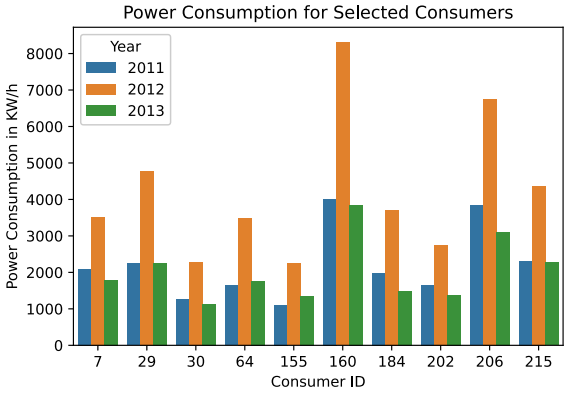


FIGURE 2. Aggregated power consumption of ten consumers over three years (2011-2013).

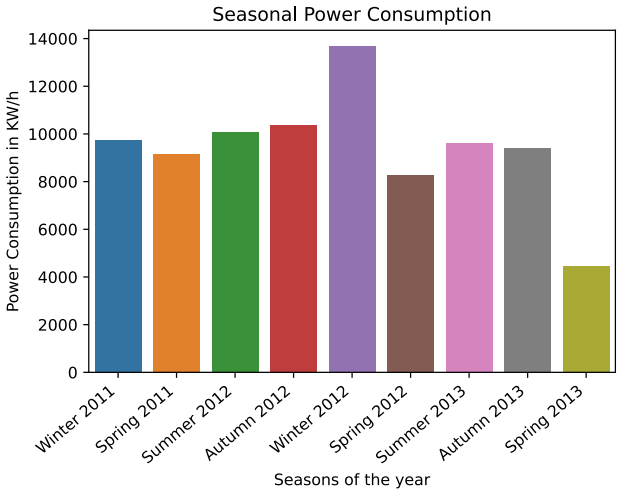


FIGURE 3. Aggregated power consumption of ten consumers categorized over four seasons.

1) SELF-SIGNED SSL/TLS CERTIFICATES

SSL/TLS protocol is implemented using self-signed certificates in our FL training setup. The certificate serves as an identifier that validates the identity of a device. It contains the devices' public keys used for data encryption. The device also has a secret private key used for data decryption. A publicly trusted third party, the Certificate Authority (CA), is responsible for creating, signing, and issuing SSL certificates. In our FL setup, we develop a local CA for signing certificates. These certificates are called self-signed certificates that do not require the validation of a trusted third party. As our FL training setup is an internal network, self-signed certificates are sufficient for implementing SSL.

IV. FEDERATED LEARNING FOR ANOMALY DETECTION DATASETS

We use the KDD 99 [71], NSL-KDD [72], and CIDDS-001 [73] datasets to study the detection performance of federated learning models. After preprocessing, each dataset is divided into train and test sets. ML models are developed

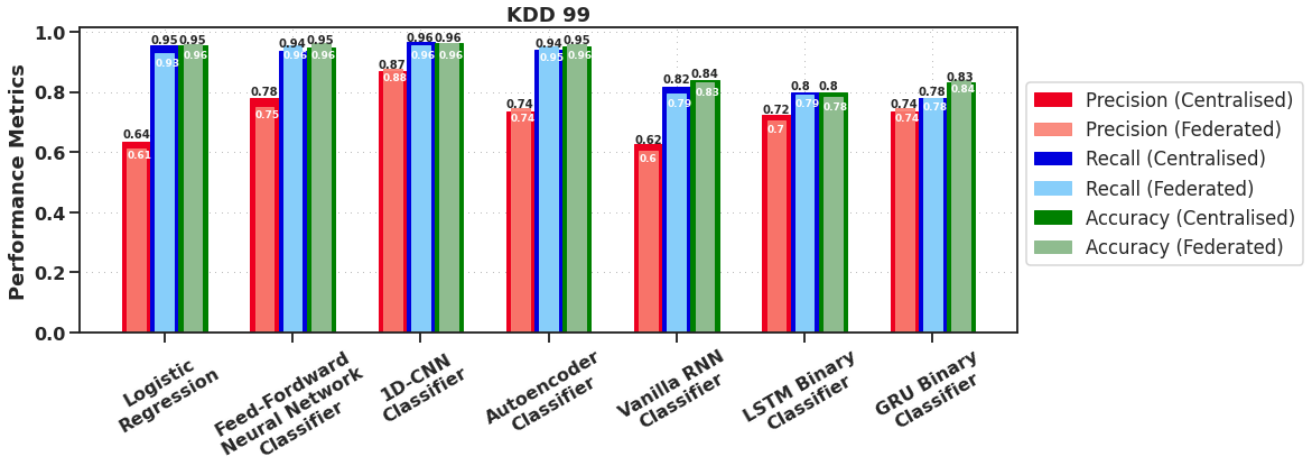


FIGURE 4. Comparison between Centralised and Federated models for KDD 99 dataset.

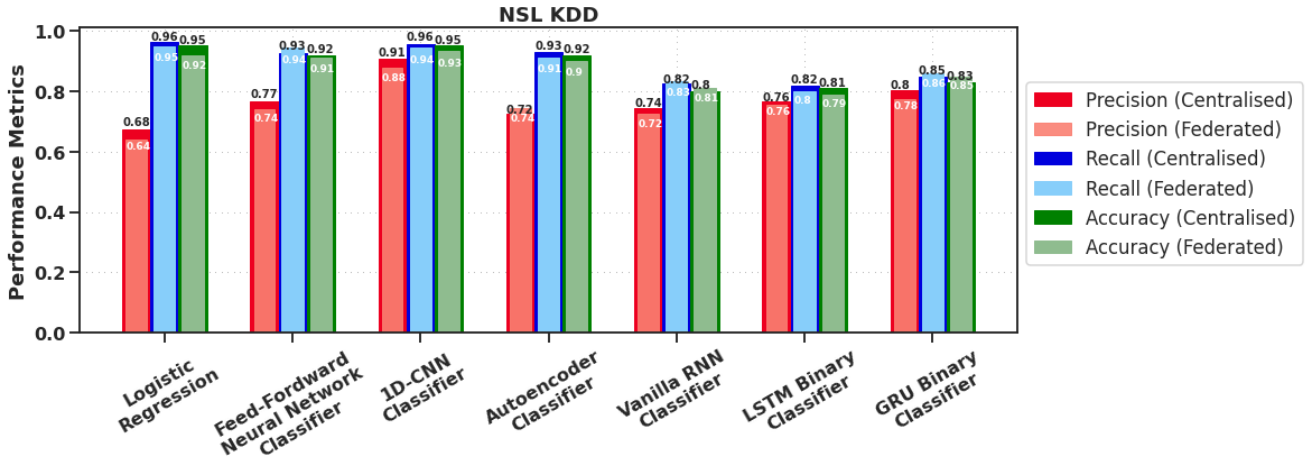


FIGURE 5. Comparison between Centralised and Federated models for NSL KDD dataset.

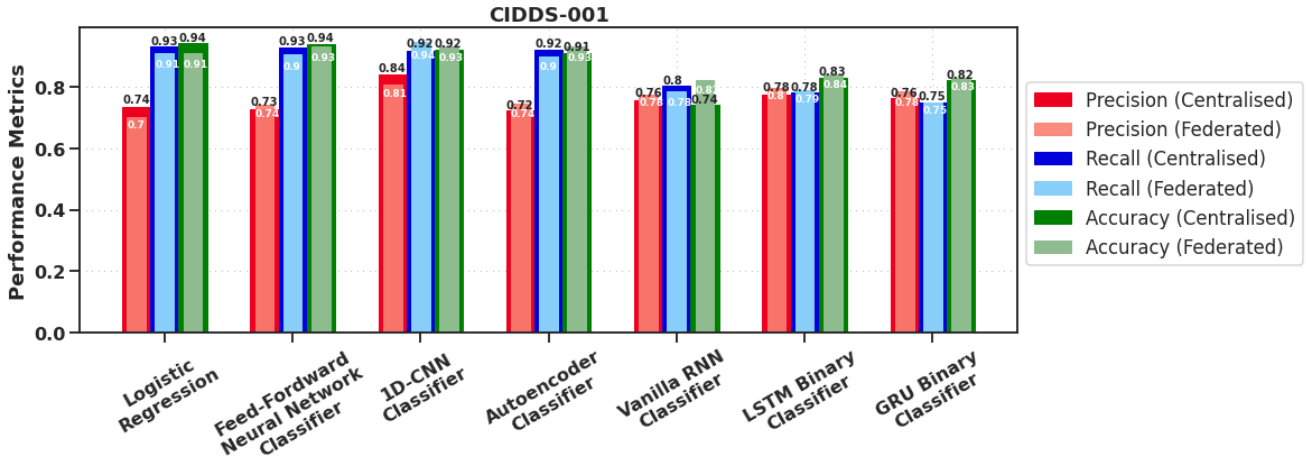


FIGURE 6. Comparison between Centralised and Federated models for CIDDS dataset.

using the train sets. These models are developed using centralised training in the local workstation. Next, the models were trained in a federated manner using the TensorFlow federated framework for 100 clients. Finally, we compare the average performance of federated models with corresponding centralised models in terms of *Precision*, *Recall*, and *Accuracy* scores. Further, we use the smart meter dataset provided by Ausgrid to analyze the performance of federated learning for anomaly detection in smart grids.

In this section, we compare the performance of federated learning-based models to that of centralised ML models for anomaly detection. We formulate anomaly detection as a classification problem, wherein various centralised ML and federated learning techniques are applied to classify the network data into normal and anomalous. We develop seven ML models: Logistic Regression, Feedforward Neural Network Classifier, 1D-CNN Binary Classifier, Autoencoder Binary Classifier, Vanilla RNN Classifier, LSTM Binary Classifier, and GRU Binary Classifier for anomaly detection in three widely used datasets. Using *Precision*, *Recall*, and *Accuracy* metrics, we compare the performance of centralised models to that of the corresponding federated models. The optimized model hyperparameters for each dataset are estimated using five-fold cross-validation and are listed in Table 2. For the simulations, we chose 100 clients and a central server. The simulations were conducted on a Google Colab instance with the following specifications:

- CPU: Intel (R) Xeon (R) processor clocked at 2.20 GHz
- RAM: 13 GB
- Storage: 145 GB
- GPU: Tesla T4 GPU with 16 GB memory

For Federated learning simulation, we use Tensorflow Federated 0.36.0 with Tensorflow version 2.10 backend. Details of the anomaly detection datasets used are described in the following section.

TABLE 2. Hyperparameter settings for ML models.

Model	Hyperparameters	Range Explored	KDD 99	NSL KDD	CIDDS-001
Logistic Regression	Penalty	[L1, L2, Elasticnet, None]	L2	L2	L2
	Inverse Regularization	[.001, .01, 1, 5, 10]	5	1	5
	Solver	[lbfgs, newton-cg, liblinear, sag]	lbfgs	lbfgs	lbfgs
	Max. Iterations	[200, 500, 1000, 5000]	1000	1000	500
Feedforward Neural Network Classifier	Number of Hidden Layers	[1, 2, 3, 4]	1	1	1
	Batch Size	[50, 100, 200, 300]	50	50	100
	Epochs	[5, 10, 25, 20]	5	10	25
	Optimizer	[SGD, RMSProp, Adam]	Adam	Adam	Adam
1D-CNN Binary Classifier	Learning Rate	[.001, .01, .1, .2]	.001	.01	.001
	Batch Size	[100, 500, 1000, 2000]	100	500	500
	Epochs	[5, 10, 15, 20, 25]	5	10	25
	Optimizer	[SGD, RMSProp, Adam]	Adam	Adam	Adam
Autoencoder (AE) Binary Classifier (BC)	Epochs (AE)	[5, 10, 25, 20]	10	10	25
	Encoding Dimensions (AE)	[5, 10, 25, 20]	5	5	10
	Epochs (BC)	[10, 50 , 100, 200]	50	50	100
	Batch Size (BC)	[100, 200, 500, 1000]	200	200	500
Vanilla RNN Classifier	Learning Rate	[.001, .01, .1, .2]	.001	.001	.01
	Batchsize	[8, 16, 32, 48]	16	32	32
	Epochs	[10, 20, 30, 40, 50]	40	50	30
	Units	[10, 20, 30, 40, 50]	20	20	20
	Optimizer	SGD, RMSProp, Adam	Adam	Adam	Adam
	Dropout	[.1, .2, .3, .4]	.2	.1	.1
LSTM Binary Classifier	Learning Rate	[.001, .01, .1, .2]	.001	.01	.001
	Batchsize	[8, 16, 32, 48]	16	32	32
	Epochs	[10, 20, 30, 40, 50]	50	30	40
	Units	[10, 20, 30, 40, 50]	20	20	20
	Optimizer	SGD, RMSProp, Adam	Adam	Adam	Adam
	Dropout	[.1, .2, .3, .4]	.2	.2	.1
GRU Binary Classifier	Learning Rate	[.001, .01, .1, .2]	.001	.001	.001
	Batchsize	[8, 16, 32, 48]	16	32	16
	Epochs	[10, 20, 30, 40, 50]	50	40	30
	Units	[10, 20, 30, 40, 50]	20	20	20
	Optimizer	SGD, RMSProp, Adam	Adam	Adam	Adam
	Dropout	[.1, .2, .3, .4]	.2	.1	.2

A. DATASETS

1) KDD 99 DATASET

The KDD 99 dataset [71] is the most popular dataset used for IDS research. It is a subset of the DARPA-98 dataset, comprising 41 feature vectors, including numeric and categorical attributes. The dataset has five classes: *Normal, DoS (Denial-of-Service), U2R (User to Root), R2L (Remote to Local), and Probe (Probing Attack)*. The other four classes, except the Normal class, correspond to attack instances. It includes 4, 898,000 cases, of which 1, 074,900 are unique records. Due to redundancy in the records, the dataset was pruned to 311,000 records. The data preprocessing steps include removing/imputing NULL values, removing redundant records, and vectorising categorical features using one-hot encoding. To normalize the feature vectors of the KDD 99 dataset, we use *z score* normalization given by the following expression

$$x_i^{standardised} = \frac{x_i - Mean(X)}{Std.Dev(X)} \tag{3}$$

In the above expression, X is a feature vector, x_i is the i^{th} instance of the feature vector, and $x_i^{standardised}$ is the transformed feature vector. The denominator of the expression computes the standard deviation of X.

2) NSL-KDD DATASET

The NSL-KDD dataset [72] was proposed to improve the performance of the KDD 99 dataset for intrusion detection. It is composed of selected records of the KDD 99 data set. The advantages of the NSL-KDD dataset over the KDD 99 dataset are 1.) No redundant records in the train and test set. 2.) The reduced dataset size removes the need for random sampling. 3.) The selected records in each class of the NSL-KDD dataset are inversely proportional to the percentage of records in the KDD99 dataset. The accuracy of distinct machine learning methods varies over a broader range, resulting in more accurate evaluation for different models. The training dataset comprises 125,970 instances, whereas the test dataset contains 22,5440 samples. It comprises four attack categories: *R2L, U2R, Probe, DoS, and a Normal class.* The data preprocessing steps for the NSL-KDD dataset include removing NULL values, removing redundant records, and vectorising categorical features using one-hot encoding. This method transforms the magnitude of feature vectors to the [0, 1] scale using the following mathematical expression:

$$x_i^{transformed} = \frac{x_i - Min(X)}{Max(X) - Min(X)} \tag{4}$$

In the above expression, X is a feature vector, x_i is the i^{th} instance of the feature vector, and $x_i^{transformed}$ is the transformed feature vector.

3) CIDDS-001 DATASET

The CIDDS dataset [73] is a recently reported dataset growing in popularity for IDS research. It contains both normal data and several types of cyberattacks, such as *Port scans, Ping scans, DoS attacks, and Brute Force attacks*. The dataset comprises thirteen feature vectors, including source and destination ports, source and destination IDs, packet size, and duration. Non-numeric features, such as the protocol type and flags, were transformed into numeric forms for model training. The dataset has five classes: attack, suspicious, victim, normal and unknown. The dataset was divided into two sets: training and testing sets. In our case, 80% of the data was used for model training and the rest for model testing. The training set contained 536,992 instances, whereas the testing set comprised 134, 248 cases. The data preprocessing steps for the CIDDS-001 dataset include removing NULL values and redundant records, vectorising categorical features using one-hot encoding, and normalisation using the *MinMax* scaling method.

B. PERFORMANCE COMPARISON: CENTRALISED VERSUS FEDERATED MODELS

The performance of centralised and federated models for anomaly detection in KDD 99, NSL-KDD, and CIDDS-001 datasets are quantified in terms of *Precision*, *Recall*, and *Accuracy* metrics. A clustered bar chart (Figures 4, 5 and 6) is used for quantifying ML model performance in which red, blue, and green bars correspond to *Precision*, *Recall*, and *Accuracy*, respectively. To compare the performance of federated and centralised models, we use an overlapping bar chart. We overlay the clustered bar chart of federated models (lighter shaded and thinner bars) over the corresponding clustered bar chart of centralised models (darker shaded and thicker bars) as described in Figure 4 (for KDD 99), Figure 5 (for NSL-KDD) and Figure 6 (for CIDDS). Such an arrangement allows us to visually compare the different performance metrics of federated models to that of centralised models. From Figures 4, 5, and 6, we observe that in some cases, federated models marginally outperform centralised models and vice versa in other cases, but the difference is marginal. We conclude that there is no substantial difference in performance between centralised and corresponding federated models across all three datasets.

The performance difference between centralised and federated learning models is attributed to the variation in the distributions of datasets [74]. This is especially true for federated learning, where it has been found that the distribution of client data has a direct impact on FL model performance. When data distribution varies between clients, FL performs poorly than centralized training. The performance of the federated model is equivalent to that of its centralized version when data is close to being independently and identically distributed (IID).

V. FEDERATED ANOMALY DETECTION FOR SMART GRIDS

A third of the world's total energy consumption is estimated to be consumed by households [75]. This trend is expected to grow as new buildings are constructed and more devices are connected to the grid. Anomalies in the smart grid can be brought about by faulty sensors, abnormal customer behavior, broken equipment, abnormal appliance usage, or cyberattacks. By analysing the power usage habits of homes, it is possible to reduce electricity waste and save millions of dollars. Studies also show that detecting these anomalies can cut energy use by 20% and prevent power failures.

Household energy usage is affected by many factors, such as weather, abnormal usage events, temperature variation, daily, weekly, and yearly seasonality, and holidays. In this section, we propose federated learning to detect anomalies in smart grids using an open dataset provided by Ausgrid Corporation, Australia. Similar to our approach used in the datasets of the previous section, we develop seven ML classifiers for centralised and federated learning and compare their anomaly detection performance. Here, in contrast to the evaluation in the Google Colab instance for anomaly detection datasets, the federated models are evaluated on a low-cost smart meter prototype developed on a Raspberry Pi device. Besides measuring the performance of federated models, we also quantify the hardware resource utilization of federated models on a Raspberry Pi device to investigate the feasibility of widespread adoption of federated learning for on-device smart grid anomaly detection for resource-constrained devices. The details of the dataset used are described in the following section.

1) AUSGRID DATASET

We use the dataset provided by Ausgrid, a power utility company in Australia [76]. The data was sourced from smart meters belonging to 300 Ausgrid customers for three years between July 1, 2010, and June 30, 2013. The energy readings were recorded every 30 minutes for this period in KWh. For customers, there were different meters for three separate categories of measurements. The different categories were Gross Generation (GG), which recorded electricity generated per customer from rooftop solar panels; Gross Consumption (GC), which measured the electricity consumed per customer; and Controlled Load (CL), which recorded off-peak controlled consumption. Each customer had GG and GC measurements, as CL was only recorded for a few customers. Since our work focuses on identifying anomalous power consumption patterns, we use the category GC.

A. EXPLORATORY DATA ANALYSIS

Exploratory Data Analysis (EDA) is a critical step for performing preliminary studies on the dataset to detect

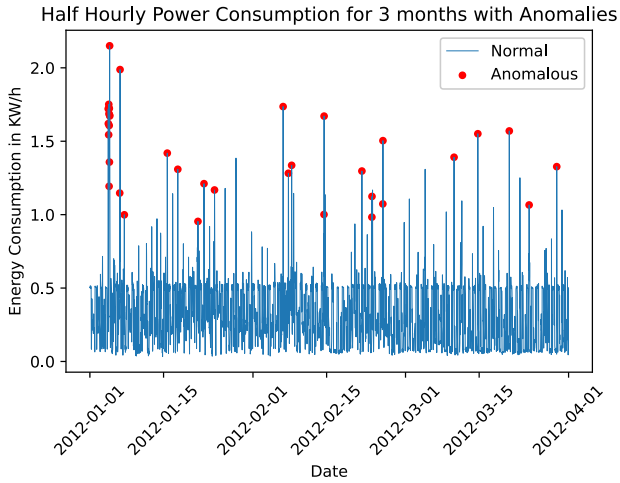


FIGURE 7. Normal and anomalous power consumption patterns for customer number 206 over three months.

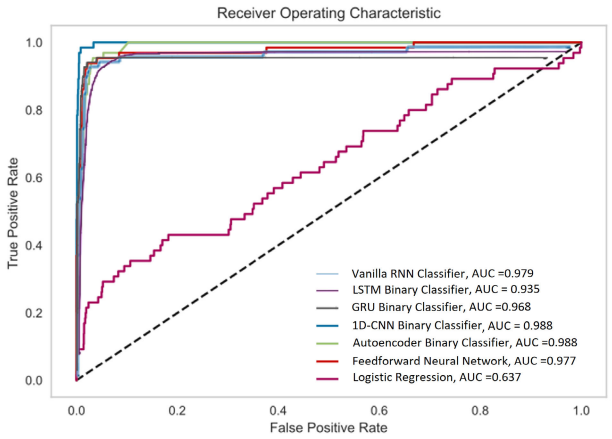


FIGURE 8. Receiver operating characteristic (ROC) of centralised ML models.

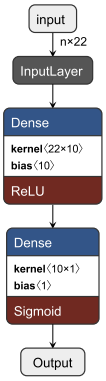


FIGURE 9. Architecture of neural network classifier.

specific patterns, spot anomalies, test hypotheses, identify features and check assumptions with the help of statistical graphs and data visualisation methods. This section describes EDA of the Ausgrid dataset.

Figure 2 compares the total yearly power consumption for the selected customers for three years starting from 2011. We observe that the highest power consumption for all consumers was in 2012. The increased power consumption in 2012 could be attributed to the relatively low temperatures during that year's winter. Failure to forecast such unusual power consumption behaviour in advance might lead to outages or grid imbalances.

Figure 3 describes the seasonal power consumption, starting from the winter (June to August) of 2011 to the spring (September-November) of 2013 for the selected consumers. The figure also shows power consumption for the summer (December-February) and autumn (March-May) seasons. We observe that power consumption for the winter of 2012 was unusually high compared to 2011 and 2013.

Figure 7 describes the power consumption pattern of customer 206 over three months, starting from January 1, 2012, to April 1, 2012. In the figure, the curve in blue represents regular power consumption, whereas the points in red denote abnormal power usage.

B. FEATURE SELECTION

In the Ausgrid dataset, 96.57% of the total data points correspond to normal behaviour, whereas only 3.43% correspond to anomalous behaviour. The significant majority of normal data points over anomalous ones leads to class imbalance. The performance of standard classification algorithms drops because of imbalanced classes. Various techniques have been developed to address the class imbalance problem, which includes under-sampling the majority class (normal data) or oversampling the minority class (anomalous data). Synthetic Minority Over-sampling Technique (SMOTE) is a popular technique to balance imbalanced datasets. SMOTE is designed to work with datasets with numerical features. Since our dataset comprises numerical and categorical variables, we use SMOTE-NC, a variant of the SMOTE algorithm, which works with datasets with numerical and categorical features. We use the *MinMax* scaling method to normalize the feature vectors.

After re-sampling using the SMOTE-NC algorithm, we observe that the training dataset is equally split between normal and anomalous cases and therefore is well balanced.

C. ML ALGORITHMS FOR ANOMALY DETECTION

In this section, we describe the various ML algorithms developed in this work for detecting anomalous energy consumption patterns among consumers using the Ausgrid smart meter dataset. Using three months of user energy consumption data (January 1, 2012, to April 1, 2012) of ten consumers, we perform ML model training using a workstation. Since data from ten consumers are aggregated at the workstation for training, the resulting ML models are referred to as *Centralised Models*.

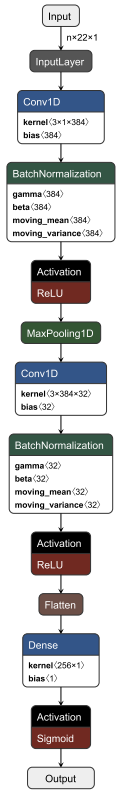


FIGURE 10. Architecture of 1D-CNN binary classifier.

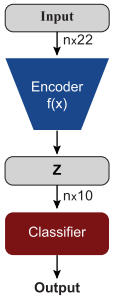


FIGURE 11. Architecture of Autoencoder binary classifier.

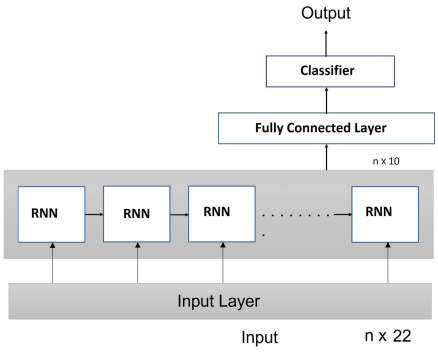


FIGURE 12. Architecture of RNN classifier.

After evaluating the performance of *Centralised Models*, we use Federated Learning to delegate anomaly detection

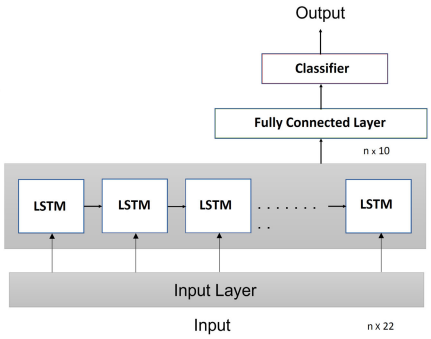


FIGURE 13. Architecture of LSTM binary classifier.

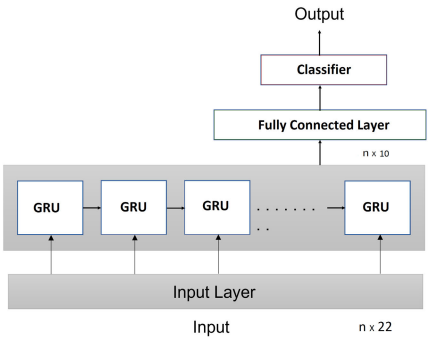


FIGURE 14. Architecture of GRU binary classifier.

task to smart meters. Distributed anomaly detection using smart meters positioned at the edge of the smart grid network would enable the system to respond faster to anomalous usage patterns. Finally, we compare the performance of *Federated* and *Centralised* ML models.

1) LOGISTIC REGRESSION

Logistic regression is a statistical model used for classification tasks. For a dataset, logistic regression estimates the probability of the occurrence of an event based on independent variables. It uses logit transformation to give probabilities as the outcome of the algorithm. The output Z of logistic regression can be described as

$$Z = \log \frac{p_i}{1 - p_i} = \beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip} \tag{5}$$

Here, p_i denotes the probability of success of i^{th} instance, x_i represents predictor variables, p corresponds to the total number of predictors, and β denotes the co-efficient of predictors.

2) FEED-FORWARD NEURAL NETWORK (FFNN) CLASSIFIER

Feed-Forward Neural Network (FFNN) comprises an input layer, several hidden layers, and an output layer. The neurons in one layer are connected to the next layer through weights. The magnitude of weight represents the strength of the connection between neurons. Each neuron receives inputs from neurons in the previous layer and generates a

FIGURE 15. Experimental setup for FL-based smart grid anomaly detection using smart meter dataset.

weighted summation of its inputs. Next, the neuron applies an activation function to the generated sum and passes on the result to other neurons. For each neuron l in the hidden layer, its output s_l is calculated using the following mathematical expression

$$s_l = \sum_{i=1}^{N} s_i w_{il} \tag{6}$$

where w represents the weight of neurons and N denotes the total number of neurons. Similarly, for each neuron p in the output layer, the output of the neuron o_p is computed using the following expression.

$$o_p = \sum_{i=1}^{N} s_l w_{lp} \tag{7}$$

Since our problem is a binary classification task, we develop an FFNN classifier with a single output neuron with a sigmoid activation function. Figure 9 describes the architecture of the FFNN classifier used in this work.

3) 1D-CNN BINARY CLASSIFIER

The Convolutional Neural Network model develops an internal representation of the 2D input for generating predictions. CNNs are also widely used for classifying one-dimensional data as they can learn from raw time series data without requiring feature engineering. Since the energy consumption data is essentially time-series data, we utilise one-dimensional CNN (1D-CNN) to identify anomalous samples. 1D convolution operation can be expressed as

$$y_j = \sum_{k=-m}^{m} x_{j-k} w_k \tag{8}$$

where x is the input sequence, w represents the kernel of length $2m + 1$, and y denotes the output sequence. Figure 10 represents the architecture of the 1D-CNN model used in this work.

4) AUTOENCODER BINARY CLASSIFIER

Autoencoder is an unsupervised neural-network architecture that compresses high dimensional input data to a low dimensional vector. The architecture of an autoencoder consists of three components: the encoder, the hidden or bottleneck layer, and the decoder. The encoder layer compresses the input data to lower dimensions at the hidden layer, whereas the decoder layer reconstructs the original input data from the compressed code. Autoencoder can be used for compressing higher dimensional data to a preferred dimensionality. The following function represents the encoder

$$z = f(x) \tag{9}$$

whereas the decoder is expressed as

$$r = g(z) \tag{10}$$

Here, x is the input, z is the compressed code vector, and r represents the reconstructed output. The data compression at the encoder can be used as a feature extraction technique to train a machine learning classifier. This feature extraction technique saves the encoder layer and discards the decoder layer after model training of the autoencoder. The compressed data at the hidden/ bottleneck layer is used to train a classifier. In our approach, we train the autoencoder with our smart meter dataset, discard the decoder and use a logistic regression classifier as the final layer to classify normal and anomalous samples. Figure 11 describes the architecture of the Autoencoder binary classifier used in this work.

5) VANILLA RNN CLASSIFIER

A Recurrent Neural Network (RNN) is a type of neural network which differs marginally in architecture from conventional feed-forward neural networks (FFNNs). In contrast to the traditional FFNN architecture, RNN architecture has a directional loop that is used to compare the error of a hidden layer to that of the previous layer and adjust the weights between layers. If $I = (i_1, i_2, i_3, \cdots, i_T)$, $H = (h_1, h_2, h_3, \cdots, h_T)$ and $O = (o_1, o_2, o_3, \cdots, o_T)$ represent the input vector, hidden layers, and the output vector of an

RNN with time sequence $t = 1, 2, \cdots, T$. The output of the RNN at time t is calculated as:

$$o_t = h_t W_{ho} + b_y \tag{11}$$

where h_t is given by

$$h_t = f(W_{ih} + h_{t-1} W_{ii} + b_h) \tag{12}$$

In the above equations, f denotes a nonlinear activation function, W represents a weight matrix, b corresponds to a bias term and h_t and h_{t-1} denote the hidden layers at t^{th}, and $(t-1)^{th}$ time steps. Traditionally, RNNs were used for time series prediction problems. In [18], [19], and [20], researchers reported that RNN-based classifiers provide excellent performance in intrusion detection. Figure 12 describes the architecture of the RNN binary classifier used in this work.

6) LSTM BINARY CLASSIFIER

RNN-based models are prone to vanishing and exploding gradient problems due to improperly assigned weights. Long Short-Term Memory Units (LSTMs) were developed to address this issue. An LSTM unit comprises several regulating gates which control the information flow. It also has a forget gate between input and output gates to reset the memory if the information is no longer required. The output $o^{c_j}(t)$ of an LSTM unit is computed as:

$$o^{c_j}(t) = h(s_{c_j}(t)) o^{out_j}(t) \tag{13}$$

In the above equation, $o^{out_j}(t)$ is output from the activation function of the output gate, $s_{c_j}(t)$ denotes the internal state of the output gate, and h represents the output of the hidden layer at time t. Figure 13 describes the architecture of the LSTM binary classifier used in this work.

7) GRU BINARY CLASSIFIER

The Gated Recurrent Units (GRU) were introduced as an alternative to LSTM units with a comparatively simpler architecture and faster training time. A typical GRU unit comprises two gates, reset and update gates. The reset gate in GRU is analogous to the forget gate in LSTM. Similar to the LSTM unit, the output of the hidden state h_t at time t is computed using the hidden state h_{t-1} at time $t-1$ as follows:

$$h_t = f(o_t, h_{t-1}) \tag{14}$$

In the above equation, f denotes a nonlinear activation function, and o_t represents the output at time t. Figure 14 describes the architecture of the GRU binary classifier used in this work.

D. ASSESSING MODEL PERFORMANCE

Predicting anomalous power consumption from smart meter data is a binary classification problem. The minority group (anomalous data points) is the positive class, and the majority group (normal data points) is the negative class. A classification algorithm's effectiveness is evaluated by comparing its projected and actual class labels. The training set, which makes up 80% of the dataset, is used to train the classification algorithm. The remaining 20% of the dataset is used to assess the model performance. Four unique outcomes are likely for the model's predictions. They are defined as follows:

- **True Positive (TP)**: Outcomes when the ML model correctly predicts the positive (anomalous) class.
- **True Negative (TN)**: Outcomes when the ML model correctly predicts the negative (anomalous) class
- **False Positive (FP)**: Outcomes when the ML model incorrectly predicts the positive (anomalous) class (True label is negative).
- **False Negative (FN)**: Outcomes when the ML model incorrectly predicts the negative (anomalous) class (True label is positive).

The standard metrics used for evaluating classification models are *Precision, Recall, Accuracy, F1-score,* and *AUC* (Area Under the Receiver Operating Characteristic (ROC) curve). These metrics are defined as follows:

- **Precision** = $\frac{TP}{TP+FP}$
  Precision measures the proportion of positive predictions made by the classifier that is actually correct
- **Recall** = $\frac{TP}{TP+FN}$
  Recall measures the proportion of actual positive predictions that are identified correctly
- **F1-score** = $\frac{2Precison*Recall}{Precision+Recall}$
  F1-score is the harmonic mean of precision and recall metrics which combines them into a single metric
- **Accuracy**= $\frac{TP+FP}{TP+TN+FP+FN}$
  Accuracy quantifies the proportion of predictions correctly made by the classifier out of its total predictions
- **AUC**= $\int_0^1 TPR\, d(FPR)$
  Here, *TPR* denotes the True Positive Rate, and *FPR* represents the False Positive Rate. ($TPR = \frac{TP}{TP+FN}$ and $FPR = \frac{FP}{FP+TN}$).
  AUC metric is a measure of the performance of the model across all possible classification thresholds. Higher values of *AUC* indicate better model performance. The maximum value of *AUC* is 1, indicating that the model perfectly distinguishes between positive and negative classes.

E. CENTRALIZED MODEL TRAINING

We use the resampled Ausgrid dataset in centralized model training to train the proposed anomaly detection models. We divide the dataset into a training set of 50, 000 samples (9.78 MB) and a test set of 20, 000 samples (3.03 MB). The training set is used for model training, and the test set is used for performance evaluation. We apply the *Grid Search*

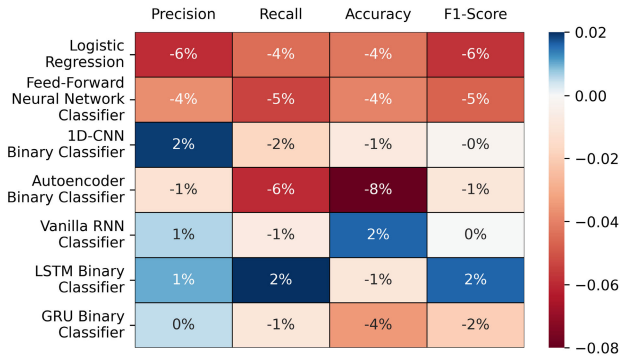


FIGURE 16. Percentage change (Δ%) in performance metrics between federated and centralised models (Δ= Performance metric of federated model - Performance metric of centralised model).

hyperparameter optimization for hyperparameter tuning. The optimum hyperparameters of the models are given in Table 2. After model training, we evaluate the model performance metrics on the test dataset. Table 3 gives the performance metrics of the proposed anomaly detection models. Figure 8 describes the ROC curves of centralised models with *AUC* values. From the table, we observe that the 1D-CNN Binary Classifier performs better compared to other models with higher accuracy.

F. FEDERATED MODEL TRAINING

In this section, we investigate the performance of the proposed FL-based smart meter anomaly detection models on actual edge devices (client devices) operating in an FL environment. In addition, using the Ausgrid dataset, we evaluate the computational and communication overheads, power consumption, and time overhead incurred by running FL anomaly detection models on client devices. Measuring the resource utilisation of FL models on edge devices is critical, as this provides a benchmark for FL-based anomaly detection under real-world settings with limited hardware capabilities. Figure 15 shows the experimental setup for the federated anomaly detection on smart meter data. The figure has 3 panels. The first panel shows the whole experimental setup, the second panel shows the workstation server and client devices, and the third panel shows the setup for power measurement of client devices using the USB voltmeter.

(a) *Smart meter prototype:* Figure 18 shows the smart meter prototype used in our work. We use four Raspberry Pi 4 model B devices with the following specifications as the smart meter prototype (client device):
- Operating System: Raspbian GNU/Linux 11 (bullseye)
- Python: Python version 3.7.12
- Tensorflow: Tensorflow version 2.5.0
- Processor: ARM Cortex-A72
- Memory: 4 GB
- Storage: 64 GB

TABLE 3. Hyperparameter settings for ML models.

Model	Hyper-parameters	Range Explored	Best Value
Logistic Regression	Penalty	[L1, L2, Elasticnet, None]	L2
	Inverse Regularization	[.001, .01, 1, 5, 10]	10
	Solver	[lbfgs, newton-cg, liblinear, sag]	lbfgs
	Max. Iterations	[200, 500, 1000, 5000]	3000
Feedforward Neural Network Classifier	Number of Hidden Layers	[1, 2, 3, 4]	1
	Batch Size	[50, 100, 200, 300]	100
	Epochs	[5, 10, 25, 20]	5
	Optimizer	[SGD, RMSProp, Adam]	Adam
1D-CNN Binary Classifier	Learning Rate	[.001, .01, .1, .2]	.001
	Batch Size	[100, 500, 1000, 2000]	1000
	Epochs	[5, 10, 15, 20, 25]	5
	Optimizer	[SGD, RMSProp, Adam]	Adam
Autoencoder (AE) Binary Classifier (BC)	Epochs (AE)	[5, 10, 25, 20]	20
	Encoding Dimensions (AE)	[5, 10, 25, 20]	10
	Epochs (BC)	[10, 50 , 100, 200]	100
	Batch Size (BC)	[100, 200, 500, 1000]	500
Vanilla RNN Classifier	Learning Rate	[.001, .01, .1, .2]	.001
	Batchsize	[8, 16, 32, 48]	32
	Epochs	[10, 20, 30, 40, 50]	50
	Units	[10, 20, 30, 40, 50]	20
	Optimizer	[SGD, RMSProp, Adam]	Adam
	Dropout	[.1, .2, .3, .4]	.1
LSTM Binary Classifier	Learning Rate	[.001, .01, .1, .2]	.001
	Batchsize	[8, 16, 32, 48]	32
	Epochs	[10, 20, 30, 40, 50]	50
	Units	[10, 20, 30, 40, 50]	20
	Optimizer	[SGD, RMSProp, Adam]	Adam
	Dropout	[.1, .2, .3, .4]	.1
GRU Binary Classifier	Learning Rate	[.001, .01, .1, .2]	.001
	Batchsize	[8, 16, 32, 48]	32
	Epochs	[10, 20, 30, 40, 50]	50
	Units	[10, 20, 30, 40, 50]	20
	Optimizer	[SGD, RMSProp, Adam]	Adam
	Dropout	[.1, .2, .3, .4]	.1

(b) *Server:* The server is a laptop with the following specifications

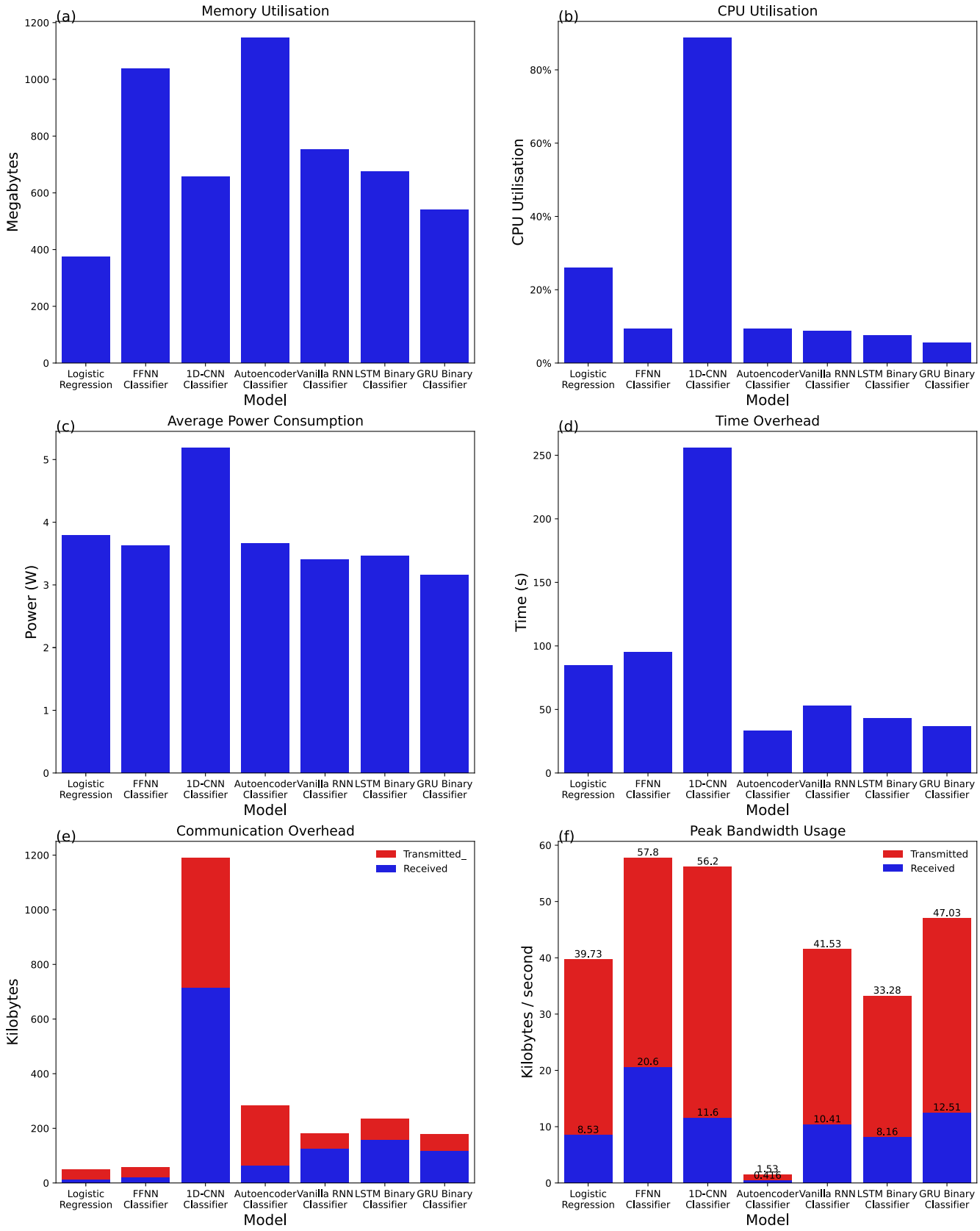


FIGURE 17. Resource usage comparison of federated models in Raspberry Pi 4 device.

TABLE 4. Performance metrics of centralised ML models.

Model	Precision	Recall	Accuracy	F1-score
Logistic Regression	.421	.943	.953	.584
Feed-Forward Neural Network Classifier	.732	.956	.971	.829
1D-CNN Binary Classifier	.855	.966	.981	.907
Autoencoder Binary Classifier	.675	.951	.966	.790
Vanilla RNN Binary Classifier	.630	.708	.801	.670
LSTM Binary Classifier	.650	.690	.830	.670
GRU Binary Classifier	.700	.730	.890	.710

TABLE 5. Average performance metrics of FL models.

Model	Precision	Recall	Accuracy	F1-score
Logistic Regression	.396	.902	.913	.550
Feed-Forward Neural Network Classifier	.705	.904	.932	.792
1D-CNN Binary Classifier	.871	.950	.989	.909
Autoencoder Binary Classifier	.687	.891	.886	.776
Vanilla RNN Binary Classifier	.636	.701	.817	.670
LSTM Binary Classifier	.660	.710	.720	.686
GRU Binary Classifier	.705	.720	.855	.690

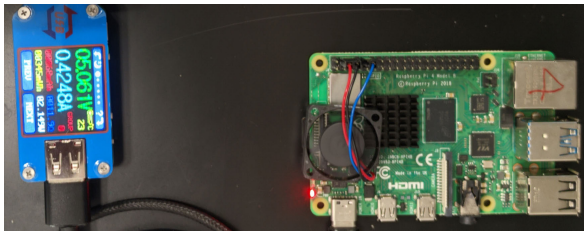


FIGURE 18. Smart meter prototype.

- Operating System: Ubuntu/Linux 20.04.4 LTS (Focal Fossa)
- Python: Python version 3.8.10
- Tensorflow: Tensorflow version 2.8.0
- Processor: AMD A10-4600M APU
- Memory: 8 GB
- Storage: 1024 GB

(c) *Communication Protocol:* Raspberry Pi client devices and the laptop (server) communicate using a router through WiFi interface.

We implement our proposed FL-based smart meter anomaly detection algorithms on four Raspberry Pi client devices and a laptop server using the Flower federated learning framework [77].

For the performance evaluation of FL anomaly detection models on Raspberry Pi devices, we randomly sample a training dataset of 20000 samples (3948 KB) and a test dataset of 6000 samples (910 KB) from the resampled Ausgrid dataset. Random sampling is performed four times for each Raspberry Pi device such that the resulting dataset is unique after each sampling. Using Flower framework and TensorFlow 2.0, we train seven anomaly detection models: Logistic Regression, Feed-Forward Neural Network Classifier, 1D-CNN Binary Classifier, Autoencoder Binary Classifier, Vanilla RNN Classifier, LSTM Binary Classifier and GRU Binary Classifier in a federated setting for local anomaly detection in each Raspberry Pi device.

For each of the proposed anomaly detection models, the Raspberry Pi client devices collaboratively train the model using their local dataset, without sharing data, while periodically updating model updates to the central server. For evaluation, we trained the FL models for 10 rounds. After 10 rounds of model training, we evaluate the model performance metrics on the test dataset. The performance metrics of centralised models are described in Table 4. We compute the average performance metrics of FL models of four raspberry devices in Table 5. Table 5 shows that similar to centralised model training, federated 1D-CNN Binary Classifier outperforms other federated models.

G. PERFORMANCE COMPARISON: FEDERATED VERSUS CENTRALISED MODEL TRAINING

In this section, we compare the performance of federated and distributed ML models in smart grid anomaly detection. Towards this goal, we define a metric Δ, which is the difference between the performance metric of a federated model and that of a centralised model (Δ= Performance metric of the federated model - Performance metric of the centralised model). This metric is computed for all the performance indicators (*Precision, Recall, F1-Score*, and *Accuracy*). We calculate this metric for all the ML models and plot the percentage change in Δ as $\Delta\%$ in a heat map, as shown in Figure 16.

In the heat map, rows represent ML models, and columns represent performance metrics. The cells represent the percentage change ($\Delta\%$) in performance metrics. Cells in red indicate that the performance of the federated model was lesser than the corresponding centralised model, and for cells in blue, vice versa. From the heat map, we observe that *Precision* of the federated 1D-CNN Binary Classifier, *Accuracy* of federated Vanilla RNN classifier, *F1-Score* and *Recall* of federated LSTM Binary Classifier increased by 2% compared to the centralised models. We also observe that the

Accuracy of the centralised Autoencoder Binary Classifier decreased by 8%, whereas *Precision*, *F1-score* of logistic regression decreased by 6% compared to the federated model. We conclude that the variation in the performance of federated models over centralised models is minimal. The performance of centralised and federated models for the smart meter dataset is broadly comparable. Depending on the specific application scenario, FL can replace centralised models, mainly when data security and privacy are the foremost concern. The diversity in dataset distributions is thought to cause performance variation between centralised and federated models. This is especially true for federated learning, where it has been discovered that the distribution of client datasets affects the effectiveness of FL models. When data distribution changes across clients, FL performs worse than centralised training. When data is almost Independently and Identically Distributed (IID), the performance of the federated model is roughly equivalent to that of its centralized variant.

1) MEASUREMENT OF PERFORMANCE METRICS

(a) *Memory Utilization:* Measuring device memory usage while running federated ML models is critical for resource-constrained IoT devices. In our experiment, we profile the memory usage of a Raspberry Pi device as it runs federated anomaly detection models. The Raspberry Pi 4 device used in our experiment has 3.7 GB of usable RAM available. We use *Conky* [78], a free, lightweight system monitoring program to measure the memory utilization of Raspberry Pi devices as they run different federated anomaly detection models on the Ausgrid dataset. Panel *(a)* of Figure 17 compares the memory utilization of different federated anomaly detection models. From the plot, we observe that memory utilization is maximum (1147 MB) for Autoencoder binary classifier and minimum for the logistic regression classifier (375 MB).

(b) *CPU Utilization:* Assessing the computational load of federated ML models is a crucial issue of consideration for devices with lesser computation capabilities, such as IoT devices. In our experiment, we profile the CPU utilization of Raspberry Pi devices as they run federated anomaly detection models. The Raspberry Pi 4 device used in our experiment uses a *Broadcom BCM2711, Quad-core Cortex-A72* processor clocked at 1.5 GHz. Again, we use *Conky* to measure the CPU utilization of Raspberry Pi devices as they run different federated anomaly detection models. Panel *(b)* of Figure 17 compares the CPU utilization of federated anomaly detection models. We observe that 1D-CNN uses 88.78% (the maximum) CPU, whereas GRU Binary Classifier uses 5.56% (the minimum) CPU.

(c) *Power Consumption:* IoT devices in many applications, such as agriculture and disaster monitoring, operate on battery power with limited recharging facilities. The energy-constrained nature of IoT devices necessitates the use of power-efficient ML algorithms. In panel *(c)* of Figure 17, we measure the average energy consumption of Raspberry Pi devices as they run federated anomaly detection models. We use *MakerHawk USB Power Meter UM25C* [79] to measure the power of Raspberry Pi devices running different federated anomaly detection models. The third panel of Figure 15 shows the third panel shows the setup for power measurement of client devices using the USB voltmeter. The idle power consumption of the Raspberry Pi device is calculated to be 2.85 W. We observe that power consumption is the highest for 1D-CNN (5.19 W, including the idle power consumption) and the lowest for GRU Binary Classifier (3.16 W, including the idle power consumption)

(d) *Time Overhead:* In addition to the computational cost of training federated ML models, estimating the time required for model training is critical. For example, extended training periods may negatively impact the practical usability of ML models. In panel *(d)* of Figure 17, we compare the average training times of different federated anomaly detection ML models. We use *Conky* to calculate the training times of federated models. We observe that training time is the highest for 1D-CNN (256 s) and the lowest for Autoencoder Classifier (33 s).

(e) *Communication Overhead:* In FL, the communication between client devices and the server may suffer from network-related issues such as limited data plans and poor or unreliable connections. Even though training data remains in the client devices, the clients continuously send model updates to the server during each training round. Therefore, minimizing network usage for FL is of primary interest. In panel *(e)* of Figure 17, we compare the communication overhead of different federated anomaly detection models using overlay bar plots. We use *iftop* [80], a network bandwidth monitoring tool, to calculate the bandwidth utilization of different federated models. We observe that communication overhead is the highest for 1D-CNN (1191 KB) and the lowest for Logistic Regression (48.8 KB) for a single round of training.

(f) *Peak Bandwidth Usage:* In addition to measuring the communication overhead, we also measure the peak bandwidth usage of FL models using *iftop*. In panel *(f)* of Figure 17, we compare the peak bandwidth usage of different FL models using overlay bar plots. We observe that bandwidth overhead is the highest for Feed-Forward Neural Network Classifier (57.8 KB/s) and the lowest for Autoencoder Classifier (1.53 KB/s).

VI. DISCUSSION

The growing concerns regarding data security and user privacy have motivated the development of ML techniques for training ML models locally in edge devices without sending data to the centralised server. Federated learning has

emerged as a promising technique in this direction, where edge or client devices send only local model updates to the server, such that user data remains locally in the client device. In this work, we investigated the feasibility of using federated learning to detect anomalous energy consumption patterns in smart grid data.

To examine the anomaly detection performance of federated learning models for anomaly detection, we use the KDD 99, NSL-KDD, and CIDDS-001 datasets. Each dataset is separated into train and test sets after preprocessing. ML models, including Logistic Regression, Feed-forward Neural Network Classifier, 1D-CNN Binary Classifier, Autoencoder Binary Classifier, Vanilla RNN Classifier, LSTM Binary Classifier, and GRU Binary Classifier, were developed. These models were trained centrally, followed by federated training for 100 clients using the TensorFlow federated framework. Finally, the average scores for *Precision*, *Recall*, and *Accuracy* of federated and centralised models were compared. Based on our study, we concluded that the anomaly detection performance of federated models was comparable to centralised models.

Next, we analysed the anomaly detection performance of the developed ML model for smart meter data. We first evaluate our proposed models in the conventional way, where the models are trained centrally in a workstation using the aggregate data collected from all smart meters. The 1D-CNN Binary Classifier had the highest accuracy, followed by Feed-forward Neural Network Classifier.

Finally, we federate the proposed ML models using the Flower federated learning framework. Using Raspberry Pi devices as a prototype for smart meters (client/edge device) and a workstation as the server, we set up a federated learning environment for training ML models for local anomaly detection at Raspberry Pi edge devices. The FL setup comprises four Raspberry Pi devices, each with its dataset collaboratively training the local ML model for anomaly detection at the edge.

The FL training process presents another challenge: the client-server communication channel is vulnerable to cyberattacks. The FL model updates among clients and the server are susceptible to eavesdropping, Man-in-the-middle (MiTM) attacks, and tampering. Therefore, we use SSL/TLS protocol to secure the FL training process against cyberattacks, ensuring that client-server communication is encrypted. SSL encryption ensures authentication, data validation, protection from phishing, and data security. After the FL training process is completed, we evaluate the average performance of the FL models. Similar to centralised training, the 1D-CNN Binary Classifier had the highest accuracy, followed by Feed-forward Neural Network Classifier.

Then, we compare the performance of centralised and federated models in terms of *Precision, Recall, Accuracy*, and *F1-Score*. We observe that the anomaly detection of federated models was comparable to their corresponding centralised models. For smart grid applications, where user privacy and data security issues are crucial, the marginal variations in the performance of FL are not a significant disadvantage.

However, implementing FL for anomaly detection in smart grids is subject to resource constraints at the edge device. Therefore, we investigate the memory, CPU, power, and bandwidth requirements for training FL models at the edge device. Our experiments show that Logistic regression and GRU Binary classifier consume fewer CPU resources for FL training, whereas the 1D-CNN Binary Classifier is the most compute-intensive model.

A. CHALLENGES IN FEDERATED LEARNING

We identify the following as the challenges for the practical implementation of federated learning for smart grid anomaly detection.

- **Lightweight ML models:** Since models are deployed at resource-constraint edge devices, FL requires the development of lightweight, less compute-intensive models
- **Improving communication efficiency:** Since FL involves client-server communication between many devices, communication-efficient protocols that reduce the size of data packets or the number of communication rounds may be developed.
- **Handling device heterogeneity:** Client devices taking part in the FL process may vary in computational capabilities, network connectivity, and battery capacity. The FL system must be able to address such heterogeneity among clients.
- **Data heterogeneity:** Centralised machine learning models assume data to be Independent and Identically Distributed (IID). FL does not adhere to this assumption, and hence we observe marginal variation between centralised and federated model performance. FL systems must account for this statistical heterogeneity in datasets.

VII. CONCLUSION AND FUTURE WORK

Traditional ML-based anomaly detection involves transferring data to a central server where the ML model gets trained to identify abnormal patterns in data. However, this approach of sending data over an open channel makes data vulnerable to increased privacy and security risks from adversaries. In addition, traditional model training demands extensive centralised computation resources with stable network infrastructure and large bandwidth requirements. Furthermore, offloading model training to a centralised location introduces latency, affecting the real-time anomaly detection performance of the model.

We formulate anomaly detection as a classification problem, wherein various centralised ML and federated learning techniques are applied to classify the network data into normal and anomalous. We develop seven ML models: Logistic Regression, Feedforward Neural Network Classifier, 1D-CNN Binary Classifier, Autoencoder Binary Classifier, Vanilla RNN Classifier, LSTM Binary Classifier, and GRU

Binary Classifier for anomaly detection in three widely used datasets. Using precision, recall, and accuracy metrics, we compare the performance of centralised models to that of the corresponding federated models.

We also proposed a Federated Learning (FL)-based approach for anomaly detection in smart grids where ML models are trained in a distributed manner by each smart meter device without requiring to share its local data with a central server. In the proposed approach, a global model is downloaded from the server to smart meter devices for on-device training using their local dataset. Parameters of the local models are sent to the server after training to improve the global model. We secured the server-client communication using the SSL/TLS protocol to safeguard model updates from adversaries.

Next, we used the centrally trained ML models for smart grid anomaly detection and compared their performance with corresponding federated implementations in edge hardware. We found that FL models provide strong centrally-trained models while enhancing security and privacy. In addition, our work demonstrated that the proposed FL-based anomaly detection models operate efficiently in terms of memory usage, CPU usage, and power consumption on edge hardware. The memory utilisation of proposed FL-based models ranges from 10% to 31% at edge devices, whereas CPU utilisation ranges from 5.56% to 88.78%. Furthermore, FL power consumption at the edge hardware ranges from.33 W to 2.34 W. Moreover, compared to centralised solutions, the overall communication overhead ranges from 48.8 kbps to 1191 kbps, resulting in significant bandwidth savings. These indicators demonstrate that the proposed FL-based anomaly detection models are suitable for deployment at smart meters as part of smart grid infrastructure.

In future work, we aim to optimize our FL models further so that the edge devices consume fewer computational resources as possible. Additionally, we plan to evaluate the performance of the proposed approach after deployment at a large number of smart homes.

REFERENCES

[1] A. Ipakchi and F. Albuyeh, "Grid of the future," *IEEE Power Energy Mag.*, vol. 7, no. 2, pp. 52–62, Mar./Apr. 2009.

[2] R. Piacentini, "Modernizing power grids with distributed intelligence and smart grid-ready instrumentation," in *Proc. IEEE PES Innov. Smart Grid Technol. (ISGT)*, Jan. 2012, pp. 1–6.

[3] D. Tan and N. Grumman, "Energy challenge, power electronics & systems (PEAS) technology and grid modernization," *CPSS Trans. Power Electron. Appl.*, vol. 2, no. 1, pp. 3–11, Apr. 2017.

[4] M. H. Rehmani, M. E. Kantarci, A. Rachedi, M. Radenkovic, and M. Reisslein, "Smart grids: A hub of interdisciplinary research," *IEEE Access*, vol. 3, pp. 3114–3118, 2015.

[5] S. S. S. R. Depuru, L. Wang, V. K. Devabhaktuni, and N. Gudi, "Smart meters for power grid—Challenges, issues, advantages and status," in *Proc. IEEE/PES Power Syst. Conf. Expo.*, Mar. 2011, pp. 1–7.

[6] J. Zheng, D. W. Gao, and L. Lin, "Smart meters in smart grid: An overview," in *Proc. IEEE Green Technol. Conf. (GreenTech)*, Apr. 2013, pp. 57–64.

[7] A. Bari, J. Jiang, W. Saad, and A. Jaekel, "Challenges in the smart grid applications: An overview," *Int. J. Distrib. Sensor Netw.*, vol. 10, Feb. 2014, Art. no. 974682.

[8] G. W. Arnold, "Challenges and opportunities in smart grid: A position article," *Proc. IEEE*, vol. 99, no. 6, pp. 922–927, Jun. 2011.

[9] B. K. Hammerschmitt, A. D. Rosa Abaide, F. C. Lucchese, C. C. Martins, A. S. da Silveira, J. Rigodanzo, J. V. Maccari Brabo Castro, and J. A. D. A. Rohr, "Non-technical losses review and possible methodology solutions," in *Proc. 6th Int. Conf. Electric Power Energy Convers. Syst. (EPECS)*, Oct. 2020, pp. 64–68.

[10] M. Z. Gunduz and R. Das, "Cyber-security on smart grid: Threats and potential solutions," *Comput. Netw.*, vol. 169, Mar. 2020, Art. no. 107094.

[11] *Electricity Theft and Non-Technical Losses: Global Markets, Solutions, and Vendors*, Northeast Group LLC, Washington, DC, USA, 2017.

[12] N. Elmrabit, F. Zhou, F. Li, and H. Zhou, "Evaluation of machine learning algorithms for anomaly detection," in *Proc. Int. Conf. Cyber Secur. Protection Digit. Services (Cyber Security)*, Jun. 2020, pp. 1–8.

[13] M. Panthi, "Anomaly detection in smart grids using machine learning techniques," in *Proc. 1st Int. Conf. Power, Control Comput. Technol. (ICPC2T)*, Jan. 2020, pp. 220–222.

[14] J. Martinez, A. Ruiz, J. Puelles, I. Arechalde, and Y. Miadzvetskaya, "Smart grid challenges through the lens of the European general data protection regulation," in *Proc. Int. Conf. Inf. Syst. Develop.*, 2019, pp. 113–130.

[15] J. Konecný, H. B. McMahan, X. Felix Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," 2016, *arXiv:1610.05492*.

[16] The TensorFlow Federated Authors. (2018). *TensorFlow Federated, Released*. [Online]. Available: https://github.com/tensorflow/federated,version1.6.0.

[17] A. Hard, K. Rao, R. Mathews, S. Ramaswamy, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage, "Federated learning for mobile keyboard prediction," 2018, *arXiv:1811.03604*.

[18] D. Leroy, A. Coucke, T. Lavril, T. Gisselbrecht, and J. Dureau, "Federated learning for keyword spotting," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 6341–6345.

[19] S. Ramaswamy, R. Mathews, K. Rao, and F. Beaufays, "Federated learning for emoji prediction in a mobile keyboard," 2019, *arXiv:1906.04329*.

[20] X. Han, H. Yu, and H. Gu, "Visual inspection with federated learning," in *Proc. ICIAR*, 2019, pp. 52–64.

[21] S. Silva, B. A. Gutman, E. Romero, P. M. Thompson, A. Altmann, and M. Lorenzi, "Federated learning in distributed medical databases: Meta-analysis of large-scale subcortical brain data," in *Proc. IEEE 16th Int. Symp. Biomed. Imag. (ISBI)*, Apr. 2019, pp. 270–274.

[22] I. Feki, S. Ammar, Y. Kessentini, and K. Muhammad, "Federated learning for COVID-19 screening from chest X-ray images," *Appl. Soft Comput.*, vol. 106, Jul. 2021, Art. no. 107330.

[23] H. Lee, Y. J. Chai, H. Joo, K. Lee, J. Y. Hwang, S.-M. Kim, K. Kim, I.-C. Nam, J. Y. Choi, H. W. Yu, M.-C. Lee, H. Masuoka, A. Miyauchi, K. E. Lee, S. Kim, and H.-J. Kong, "Federated learning for thyroid ultrasound image analysis to protect personal information: Validation study in a real health care environment," *JMIR Med. Informat.*, vol. 9, no. 5, May 2021, Art. no. e25869.

[24] S. R. Pfohl, A. M. Dai, and K. Heller, "Federated and differentially private learning for electronic health records," 2019, *arXiv:1911.05861*.

[25] T. S. Brisimi, R. Chen, T. Mela, A. Olshevsky, I. C. Paschalidis, and W. Shi, "Federated learning of predictive models from federated electronic health records," *Int. J. Med. Inform.*, vol. 112, pp. 59–67, 2018.

[26] A. Jiménez-Sánchez, M. Tardy, M. A. González Ballester, D. Mateus, and G. Piella, "Memory-aware curriculum federated learning for breast cancer classification," 2021, *arXiv:2107.02504*.

[27] B. Camajori Tedeschini, S. Savazzi, R. Stoklasa, L. Barbieri, I. Stathopoulos, M. Nicoli, and L. Serio, "Decentralized federated learning for healthcare networks: A case study on tumor segmentation," *IEEE Access*, vol. 10, pp. 8693–8708, 2022.

[28] N. Mowla, N. H. Tran, I. Doh, and K. Chae, "Federated learning-based cognitive detection of jamming attack in flying ad-hoc network," *IEEE Access*, vol. 8, pp. 4338–4350, 2020.

[29] Y. M. Saputra, D. T. Hoang, D. N. Nguyen, E. Dutkiewicz, M. D. Mueck, and S. Srikanteswara, "Energy demand prediction with federated learning for electric vehicle networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2019, pp. 1–6.

[30] X. Yu, J. Peña Queralta, and T. Westerlund, "Federated learning for vision-based obstacle avoidance in the Internet of robotic things," 2022, *arXiv:2204.06949*.

[31] H. T. Truong, B. P. Ta, Q. A. Le, D. M. Nguyen, C. T. Le, H. X. Nguyen, H. T. Do, H. T. Nguyen, and K. P. Tran, "Light-weight federated learning-based anomaly detection for time-series data in industrial control systems," *Comput. Ind.*, vol. 140, Sep. 2022, Art. no. 103692.

[32] L. Cui et al., "Security and privacy-enhanced federated learning for anomaly detection in IoT infrastructures," *IEEE Trans. Ind. Informat.*, vol. 18, no. 5, pp. 3492–3500, May 2021.

[33] X. Wang, S. Garg, H. Lin, J. Hu, G. Kaddoum, M. J. Piran, and M. S. Hossain, "Toward accurate anomaly detection in industrial Internet of Things using hierarchical federated learning," *IEEE Internet Things J.*, vol. 9, no. 10, pp. 7110–7119, May 2022.

[34] T. T. Huong, T. P. Bac, D. M. Long, T. D. Luong, N. M. Dan, L. A. Quang, L. T. Cong, B. D. Thang, and K. P. Tran, "Detecting cyberattacks using anomaly detection in industrial control systems: A federated learning approach," *Comput. Ind.*, vol. 132, Nov. 2021, Art. no. 103509.

[35] Y. Liu, S. Garg, J. Nie, Y. Zhang, Z. Xiong, J. Kang, and M. S. Hossain, "Deep anomaly detection for time-series data in industrial IoT: A communication-efficient on-device federated learning approach," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6348–6358, Apr. 2020.

[36] V. Mothukuri, P. Khare, R. M. Parizi, S. Pouriyeh, A. Dehghantanha, and G. Srivastava, "Federated-learning-based anomaly detection for IoT security attacks," *IEEE Internet Things J.*, vol. 9, no. 4, pp. 2545–2554, Feb. 2022.

[37] R. A. Sater and A. B. Hamza, "A federated learning approach to anomaly detection in smart buildings," *ACM Trans. Internet Things*, vol. 2, no. 4, pp. 1–23, Nov. 2021.

[38] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A.-R. Sadeghi, "DÏoT: A federated self-learning anomaly detection system for IoT," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2019, pp. 756–767.

[39] J. Yeckle and B. Tang, "Detection of electricity theft in customer consumption using outlier detection algorithms," in *Proc. 1st Int. Conf. Data Intell. Secur. (ICDIS)*, Apr. 2018, pp. 135–140.

[40] B. Rossi, S. Chren, B. Buhnova, and T. Pitner, "Anomaly detection in smart grid data: An experience report," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2016, pp. 002313–002318.

[41] P. Arjunan, H. D. Khadilkar, T. Ganu, Z. M. Charbiwala, A. Singh, and P. Singh, "Multi-user energy consumption monitoring and anomaly detection with partial context information," in *Proc. 2nd ACM Int. Conf. Embedded Syst. Energy-Efficient Built Environments*, Nov. 2015, pp. 35–44.

[42] M. Toshpulatov and N. Zincir-Heywood, "Anomaly detection on smart meters using hierarchical self organizing maps," in *Proc. IEEE Can. Conf. Electr. Comput. Eng. (CCECE)*, Sep. 2021, pp. 1–6.

[43] Y. Himeur, A. Alsalemi, F. Bensaali, and A. Amira, "Smart power consumption abnormality detection in buildings using micromoments and improved K-nearest neighbors," *Int. J. Intell. Syst.*, vol. 36, no. 6, pp. 2865–2894, Mar. 2021.

[44] F. Harrou, A. Dairi, B. Taghezouit, and Y. Sun, "An unsupervised monitoring procedure for detecting anomalies in photovoltaic systems using a one-class support vector machine," *Sol. Energy*, vol. 179, pp. 48–58, Feb. 2019.

[45] A. Sial, A. Singh, and A. Mahanti, "Detecting anomalous energy consumption using contextual analysis of smart meter data," *Wireless Netw.*, vol. 27, no. 6, pp. 4275–4292, Aug. 2021.

[46] L. Zhang, L. Wan, Y. Xiao, S. Li, and C. Zhu, "Anomaly detection method of smart meters data based on GMM-LDA clustering feature learning and PSO support vector machine," in *Proc. IEEE Sustain. Power Energy Conf. (iSPEC)*, Nov. 2019, pp. 2407–2412.

[47] A. Santolamazza, V. Cesarotti, and V. Introna, "Anomaly detection in energy consumption for condition-based maintenance of compressed air generation systems: An approach based on artificial neural networks," *IFAC-PapersOnLine*, vol. 51, no. 11, pp. 1131–1136, 2018.

[48] Z. Zheng, Y. Yang, X. Niu, H.-N. Dai, and Y. Zhou, "Wide and deep convolutional neural networks for electricity-theft detection to secure smart grids," *IEEE Trans. Ind. Informat.*, vol. 14, no. 4, pp. 1606–1615, Apr. 2018.

[49] M. Zhou and P. Musílek, "Real-time anomaly detection in distribution grids using long short term memory network," in *Proc. IEEE Electr. Power Energy Conf. (EPEC)*, Oct. 2021, pp. 208–213.

[50] X. Wang, T. Zhao, H. Liu, and R. He, "Power consumption predicting and anomaly detection based on long short-term memory neural network," in *Proc. IEEE 4th Int. Conf. Cloud Comput. Big Data Anal. (ICCCBDA)*, Apr. 2019, pp. 487–491.

[51] J. Pereira and M. Silveira, "Unsupervised anomaly detection in energy time series data using variational recurrent autoencoders with attention," in *Proc. 17th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2018, pp. 1275–1282.

[52] G. Fenza, M. Gallo, and V. Loia, "Drift-aware methodology for anomaly detection in smart grid," *IEEE Access*, vol. 7, pp. 9645–9657, 2019.

[53] M. Fahim and A. Sillitti, "An anomaly detection model for enhancing energy management in smart buildings," in *Proc. IEEE Int. Conf. Commun., Control, Comput. Technol. Smart Grids (SmartGridComm)*, Oct. 2018, pp. 1–6.

[54] S. Saqaeeyan, H. H. S. Javadi, and H. Amirkhani, "Anomaly detection in smart Homes using Bayesian networks," *KSII Trans. Internet Inf. Syst.*, vol. 14, no. 4, pp. 1796–1816, 2020.

[55] B. Chen, M. Sinn, J. Ploennigs, and A. Schumann, "Statistical anomaly detection in mean and variation of energy consumption," in *Proc. 22nd Int. Conf. Pattern Recognit.*, Aug. 2014, pp. 3570–3575.

[56] A. A. Korba and N. El I. Karabadji, "Smart grid energy fraud detection using SVM," in *Proc. Int. Conf. Netw. Adv. Syst. (ICNAS)*, Annaba, Algeria, 2019, pp. 1–6, doi: 10.1109/ICNAS.2019.8807832.

[57] C. Cody, V. Ford, and A. Siraj, "Decision tree learning for fraud detection in consumer energy consumption," in *Proc. IEEE 14th Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2015, pp. 1175–1179.

[58] S. Touzani, J. Granderson, and S. Fernandes, "Gradient boosting machine for modeling the energy consumption of commercial buildings," *Energy Buildings*, vol. 158, pp. 1533–1543, Jan. 2018.

[59] J. D. de Guia, R. S. Concepcion, H. A. Calinao, S. C. Lauguico, E. P. Dadios, and R. R. P. Vicerra, "Application of ensemble learning with mean shift clustering for output profile classification and anomaly detection in energy production of grid-tied photovoltaic system," in *Proc. 12th Int. Conf. Inf. Technol. Electr. Eng. (ICITEE)*, Oct. 2020, pp. 286–291.

[60] X. Liang, B. Zhao, Q. Ma, B. Sun, and B. Cui, "Terminal access data anomaly detection based on random forest for power user electric energy data acquisition system," in *Advanced Information Networking and Applications*. Cham, Switzerland: Springer, 2019.

[61] X. Wang and S.-H. Ahn, "Real-time prediction and anomaly detection of electrical load in a residential community," *Appl. Energy*, vol. 259, Feb. 2020, Art. no. 114145.

[62] C. Chahla, H. Snoussi, L. Merghem, and M. Esseghir, "A novel approach for anomaly detection in power consumption data," in *Proc. 8th Int. Conf. Pattern Recognit. Appl. Methods (ICPRAM)*. SciTePress, 2019, pp. 483–490, doi: 10.5220/0007361704830490.

[63] A. Takiddin, M. Ismail, U. Zafar, and E. Serpedin, "Deep autoencoder-based anomaly detection of electricity theft cyberattacks in smart grids," *IEEE Syst. J.*, vol. 16, no. 3, pp. 4106–4117, Sep. 2022.

[64] J. Pei, K. Zhong, M. A. Jan, and J. Li, "Personalized federated learning framework for network traffic anomaly detection," *Comput. Netw.*, vol. 209, May 2022, Art. no. 108906.

[65] Y. Guo, Y. Wu, Y. Zhu, B. Yang, and C. Han, "Anomaly detection using distributed log data: A lightweight federated learning approach," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2021, pp. 1–8.

[66] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2019, pp. 739–753.

[67] S. Truex, L. Liu, K.-H. Chow, M. E. Gursoy, and W. Wei, "LDP-fed: Federated learning with local differential privacy," in *Proc. 3rd ACM Int. Workshop Edge Syst., Analytics Netw.*, 2020, pp. 61–66.

[68] S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, and Y. Zhou, "A hybrid approach to privacy-preserving federated learning," in *Proc. 12th ACM Workshop Artif. Intell. Secur.*, Nov. 2019, pp. 1–11.

[69] R. Xu, N. Baracaldo, Y. Zhou, A. Anwar, and H. Ludwig, "HybridAlpha: An efficient approach for privacy-preserving federated learning," in *Proc. 12th ACM Workshop Artif. Intell. Secur.*, Nov. 2019, pp. 13–23.

[70] Y. Zhao, J. Zhao, L. Jiang, R. Tan, D. Niyato, Z. Li, L. Lyu, and Y. Liu, "Privacy-preserving blockchain-based federated learning for IoT devices," *IEEE Internet Things J.*, vol. 8, no. 3, pp. 1817–1829, Feb. 2021.

[71] (1999). *Kdd99 Dataset*. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99

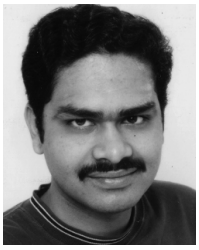
[72] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Proc. IEEE Symp. Comput. Intell. for Secur. Defense Appl.*, Jul. 2009, pp. 1–6.

[73] M. Ring, S. Wunderlich, D. Grüdl, D. Landes, and A. Hotho, "Creation of flow-based data sets for intrusion detection," *J. Inf. Warfare*, vol. 16, no. 4, pp. 40–53, 2017.

[74] M. Safa Ozdayi, M. Kantarcioglu, and R. Iyer, "Improving accuracy of federated learning in non-IID settings," 2020, *arXiv:2010.15582*.
[75] K. Zhou and S. Yang, "Understanding household energy consumption behavior: The contribution of energy big data analytics," *Renew. Sustain. Energy Rev.*, vol. 56, pp. 810–819, Apr. 2016.
[76] *Solar Home Electricity Data*. Accessed: Jun. 7, 2022. [Online]. Available: https://www.ausgrid.com.au/Industry/Our-Research/Data-to-share/Solar-home-electricity-data
[77] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, J. Fernandez-Marques, Y. Gao, L. Sani, K. Hei Li, T. Parcollet, P. Porto Buarque de Gusmão, and N. D. Lane, "Flower: A friendly federated learning research framework," 2020, *arXiv:2007.14390*.
[78] *Light-Weight System Monitor for X*. Accessed: Jul. 27, 2022. [Online]. Available: https://github.com/brndnmtthws/conky
[79] *Makerhawk UM25C USB Tester*. Accessed: Jun. 5, 2022. [Online]. Available: https://www.makerhawk.com/
[80] *IFTOP: Display Bandwidth Usage on an Interface*. Accessed: Aug. 10, 2022. [Online]. Available: http://www.ex-parrot.com/pdw/iftop/

J. JITHISH (Member, IEEE) received the B.Tech. degree in electronics and communication engineering from the University of Kerala, India, in 2012, the M.Tech. degree in electronics from the Cochin University of Science and Technology, Kerala, in 2015, and the Ph.D. degree in computer science and engineering from Amrita Vishwa Vidyapeetham, India, in 2021. He is currently a Research Fellow with the Singapore University of Technology and Design (SUTD), Singapore. His research interests include machine learning, MLOps, deep learning, data science, data engineering, statistics, cyber-physical systems, embedded systems, cybersecurity, and the Internet of Things.

BITHIN ALANGOT received the bachelor's and masters's degrees in computer science from Amrita University, in 2010 and 2012, respectively, and the Ph.D. degree in computer science and engineering from Amrita Vishwa Vidyapeetham, Amritapuri Campus, in 2021. He was a Research Fellow at the Singapore University of Technology and Design, until November 2021. His research interests include applied cryptography, blockchain security, automotive security, and distributed systems.



NAGARAJAN MAHALINGAM (Senior Member, IEEE) received the B.E. degree in electronics and communication engineering from Bharathidasan University, India, in 2001, the M.S. degree in electrical engineering from The University of Texas at Arlington, Arlington, USA, in 2005, and the Ph.D. degree from Nanyang Technological University (NTU), Singapore, in 2016. In 2006, he joined Advanced RFIC (S) Private Limited as an IC Design Engineer and worked on frequency synthesizers for portable wireless and data converter applications. He joined as a Research Associate at the Circuits and Systems Division, School of Electrical and Electronic Engineering, NTU, in 2008, and then focused on low power designs for wireless and biomedical applications, where he was an Integral Member of the 60 GHz Team which has developed the VIRTUS chipset. He is currently working as a Research Fellow II with the Singapore University of Technology and Design (SUTD). His current research interests include radio and millimeter-wave integrated circuit design with focus on oscillators and frequency synthesizers for IoT applications.

KIAT SENG YEO (Fellow, IEEE) received the B.Eng. and Ph.D. degrees in EE from Nanyang Technological University (NTU), Singapore, in 1993 and 1996, respectively. He is a Chairperson of the University Research Board and an Associate Provost with the Singapore University of Technology and Design, he is a widely known authority in low-power RF/mm-wave IC design and a Recognized Expert in CMOS technology. He was an Associate Chair (Research), the Head of Circuits and Systems, and the Founding Director of Virtus (Centre for Integrated Circuits and Systems), School of EEE, NTU. He has published ten books, seven book chapters, over 600 journals and conference papers and holds 38 patents. He was a Member of the Board of Advisors of the Singapore Semiconductor Industry Association and holds/held key positions in several international conferences as an Advisor, the General Chair, and the Technical Chair. He is a fellow of the Singapore Academy of Engineering, the Singapore National Academy of Science, and the Asia-Pacific Artificial Intelligence Association. He was awarded the Public Administration Medal (Bronze) on National Day by the President of the Republic of Singapore and the Nanyang Alumni Achievement Award both in 2009. He is the Principal Author of World University Research Rankings (WURR) 2020 and ranked among the World's Top 2% Scientists by Stanford University in 2020 and 2021.