Engineering Applications of Artificial Intelligence 124 (2023) 106654

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

FedraTrees: A novel computation-communication efficient federated learning framework investigated in smart grids

Mohammad Al-Quraan *, Ahsan Khan, Anthony Centeno, Ahmed Zoha, Muhammad Ali Imran, Lina Mohjazi

James Watt School of Engineering, University of Glasgow, Glasgow, G12 8QQ, UK

A R T I C L E I N F O

Keywords:
Federated learning
Ensemble learning
Energy forecasting
Federated averaging
Smart city
Smart grids

A B S T R A C T

Smart energy performance monitoring and optimisation at the supplier and consumer levels is essential to realising smart cities. In order to implement a more sustainable energy management plan, it is crucial to conduct a better energy forecast. The next-generation smart meters can also be used to measure, record, and report energy consumption data, which can be used to train machine learning (ML) models for predicting energy needs. However, sharing energy consumption information to perform centralised learning may compromise data privacy and make it vulnerable to misuse, in addition to incurring high transmission overhead on communication resources. This study addresses these issues by utilising federated learning (FL), an emerging technique that performs ML model training at the user/substation level, where data resides. We introduce FedraTrees, a new, lightweight FL framework that benefits from the outstanding features of ensemble learning. Furthermore, we developed a delta-based FL stopping algorithm to monitor FL training and stop it when it does not need to continue. The simulation results demonstrate that FedraTrees outperforms the most popular federated averaging (FedAvg) framework and the baseline Persistence model for providing accurate energy forecasting patterns while taking only 2% of the computation time and 13% of the communication rounds compared to FedAvg, saving considerable amounts of computation and communication resources.

1. Introduction

Fifth-generation (5G) and beyond wireless technologies are expected to unlock the full potential of the Internet of Things (IoT), the key enabler of the smart city model (Gohar and Nencioni, 2021). The concept of smart city combines several elements, such as a smart environment, mobility, living, and energy to improve citizens' quality of life. Fulfilling smart energy, in the form of smart grids and smart buildings, has been the focus of many bodies in industry and academia through monitoring and predicting energy consumption patterns (Abdel-Basset et al., 2021). Accurate short-, mid-, or long-term energy forecasting is the ultimate goal that helps managers or consumers to prepare better future plans, thus improving energy performance.

Energy suppliers need to maintain an equilibrium point between supply and demand, since producing excessive amounts of energy will result in energy wastage. On the contrary, failure to meet consumers' demands may lead to the need to purchase energy at higher rates; otherwise, frequent blackouts will happen. Therefore, various load forecasting techniques have been considered for the efficient and reliable operation of electricity networks. Statistical forecasting methods, e.g., multiple linear regression (MLR), autoregressive (AR), and moving average (MA), were used to project past and present load profiles into future predictions. Later, the introduction of smart metering and the evolution of artificial intelligence (AI) technology paved the way for replacing traditional prediction techniques with various machine learning (ML) algorithms, due to their ability in analysing large amounts of datasets in short periods of time while providing impressive accuracy levels (Chui et al., 2018). Advanced metering infrastructure (AMI), a system of smart meters connected to a communication network for two-way communications between customers and utility companies, is the first step towards smart energy, which helps collect and analyse smart-meter data. However, collecting load profiles into a central entity to conduct energy forecasting raises privacy and security concerns (Waqas et al., 2022). Power consumption information could be misused by revealing consumers' habits, household occupancy, and energy suppliers' market strength, among others (Himeur et al., 2020). Moreover, sending massive amounts of constantly generated electrical load data to a central location, i.e. cloud server, burdens communications resources and is costly.

To address these issues, the ML community recently introduced a new learning paradigm termed as federated learning (FL) (Al-Quraan

* Corresponding author.
E-mail addresses: m.alquraan.1@research.gla.ac.uk (M. Al-Quraan), a.khan.9@research.gla.ac.uk (A. Khan), Anthony.Centeno@glasgow.ac.uk (A. Centeno), Ahmed.Zoha@glasgow.ac.uk (A. Zoha), Muhammad.Imran@glasgow.ac.uk (M.A. Imran), Lina.Mohjazi@glasgow.ac.uk (L. Mohjazi).

https://doi.org/10.1016/j.engappai.2023.106654
Received 30 September 2022; Received in revised form 26 April 2023; Accepted 13 June 2023
Available online xxxx

et al., 2021). The FL is analogous to the concept of distributed learning in terms of handling enormous datasets and developing efficient and scalable systems. However, maintaining data privacy is the goal of FL as it does not involve collecting data in a central location; instead, it sends the model to the clients where the data is generated. The FL framework is orchestrated by a server placed in a central entity, i.e., an energy supplier, to train and improve a shared model with many clients collaboratively. Two typical FL architectures exist based on the scale of the federation. The first is cross-device, where the number of clients may be massive, for example, consumers' smart meters. The second is cross-silo, which considers relatively limited and reliable clients, for example, substations. The FL process starts by initialising a global model in the server and then sending it to the clients to conduct model training. Once completed, the clients send the model updates back to the server, which will aggregate them, resulting in an updated model. Then, the updated model is sent to the clients for another training round. This process is repeated until the limit of communication rounds is reached, or the model achieves the desired accuracy.

1.1. Motivation

The use of FL in energy forecasting is still in very early stages, and few studies have considered this approach (Taïk and Cherkaoui, 2020; Savi and Olivadese, 2021; Briggs et al., 2021; Gholizadeh and Musilek, 2022; Fekri et al., 2022). In these studies, the authors focus on utilising long short-term memory (LSTM) architectures, a type of recurrent neural network (RNN) used in the field of deep learning (DL), due to their remarkable performance in predicting time-series data sequences. However, the mentioned works overlook a critical issue: DL models are extremely resource-consuming (energy, memory, processor, etc.), and the lengthy and extensive underlying mathematical operations demand resource-rich hardware. Considering such schemes of combining FL with LSTM models requires extended computation time to reach the desired precision and impedes their scalability. Furthermore, individual households are the focal point of the abovementioned studies to be used as FL clients. Conversely, our study applies FL at the substation level, i.e., a part of the power system dedicated to servicing local dwellings in a specific area, allowing for the use of FL algorithm within one or more energy suppliers, hence achieving more generalised ML models.

1.2. Contributions

In this paper, we propose FedraTrees; a novel light aggregation algorithm developed to utilise decision trees (DTs) under the FL setting. Specifically, we use the light gradient boosting model (LGBM) (Ke et al., 2017), one of the boosting techniques in ensemble learning, to be sent and trained across the clients of the FL framework. The main reasons for considering the LGBM models are their rapid training speed, lower memory consumption, higher efficiency, and accurate predictions. FedraTrees attempts to minimise computations and the number of communication rounds while guaranteeing high training performance; hence, it is envisioned to play a crucial role in a wide range of FL-based applications in several fields, such as smart energy. Moreover, this work considers the common challenge of optimising the number of communication rounds that may lead to suboptimal performance or excessive rounds of unnecessary training, thus consuming computation and communication resources. Therefore, we developed a delta-based FL stopping technique, a dynamic algorithm that monitors the FL training process and stops it when no further enhancement is possible. Furthermore, this article examines the importance of each feature used in the training process and offers a study of the effect of using a different number of features on the final training performance. Additionally, the performance of the FedraTrees algorithm is benchmarked against the popular LSTM-based federated averaging (FedAvg) and the naïve Persistence model. Finally, the proposed framework is evaluated based on state-of-the-art metrics and by conducting extensive simulations. The main contributions of the paper are summarised as follows:

- First, we introduce FedraTrees, a novel, light algorithm that employs DT-based models within the FL setup. FedraTrees using LGBM models shows improved performance in terms of the required number of communication rounds and computation time compared to LSTM-based FedAvg aggregation scheme when employed for energy forecasting.
- Unlike other FL-based energy forecasting frameworks that rely on fixing the number of communication rounds, this study develops a delta-based FL stopping technique to reach the best possible accuracy while reducing the computation and communications costs.
- A feature importance evaluation study is conducted against individual load profiles for optimal forecasting performance.
- Finally, we compare FedraTrees performance with LSTM-based FedAvg and a naïve Persistence model as a benchmark using state-of-the-art metrics. The results reveal a significant improvement in overall performance using the FedraTrees framework.

1.3. Organisation

The remainder of this paper is structured as follows. Section 2 recalls the related work, while Section 3 presents some background notions. Section 4 describes the proposed FedraTrees algorithm versus FedAvg and represents the delta-based FL stopping technique. The simulation setup, the dataset used, and the numerical results are presented in Section 5. Finally, the concluding remarks are drawn in Section 6.

2. Related work

This section first reviews the works that use different load forecasting methodologies performed locally or centrally; then it presents the state-of-the-art research that takes into account the FL framework for load forecasting.

2.1. Local and centralised approaches

Load profiles store energy/power consumption information in time-series data, which can be predicted using several approaches, such as statistical and computational intelligence. Statistical methods have been used in the literature for the short and medium-temporal forecasting ranges and show good performance. For instance, the study in Ciulla and D'Amico (2019) uses the MLR technique to verify its reliability in forecasting energy demand. Whereas the traditional AR methodology is used to predict electrical energy in Saab et al. (2001). Moreover, the studies in Singh and Pozo (2019), Ozturk and Ozturk (2018), Darbellay and Slama (2000) and Zancanaro et al. (2019) benefit from the combination of AR and MA to improve the forecasting process. Later, the focus on ML methods became dominant owing to the advantages of AI in analysing large amounts of data, which was made available by the introduction of smart metering. The contribution in Ju and Hong (2013) highlights the use of the support vector machine (SVM) to forecast the monthly electrical load of Taiwan. On the other hand, many studies use NNs in the energy forecasting domain by virtue of their impressive performance in various areas (Koprinska et al., 2018; Kuo and Huang, 2018). RNN is also widely used in smart energy, particularly LSTM and its variants (Wang et al., 2020; Yan et al., 2019).

Apart from NNs, DTs have also been used in the energy forecasting task. Although the DT approach is simple, it shows the desired performance when predicting future energy consumption (Tso and Yau, 2007; Yu et al., 2010). However, DT alone has not been widely used because it suffers from several drawbacks, such as instability, easily losing generalisation, and performing poorly with noisy and nonlinear data. Later, ensemble-based algorithms gained popularity and were explored in the energy research domain. In general, ensemble techniques possess attractive features that draw the research community's attention, such as simplicity, ease of use, interpretability, and computational efficiency.

For instance, random forest forecasting performance is compared with NN in Ahmad et al. (2017), demonstrating that both are feasible and effective in building energy applications. Gradient-boosted decision tree (GBDT) and extreme gradient boosting (XGBoost) algorithms are also utilised in predicting future electricity loads and have shown to be effective in Liu et al. (2018) and Abbasi et al. (2019), respectively.

The proposed studies and frameworks mentioned above are generally based on a centralised model training where the energy consumption information is transmitted across the network and combined in a central location. However, this training scheme raises privacy concerns. Consumers' load profiles hold sensitive information that can be used in various dimensions like inferring occupancy and usage patterns of households, government surveillance, data selling, and illegal data use among others.

2.2. FL-based load forecasting

Recently, the ML community has investigated a new research direction to address the privacy concerns associated with traditional training methods. As a distributed learning algorithm, FL can perform model training at the edge of the network without the need to share sensitive load information. Few studies have begun to consider employing FL in the context of smart energy; for example, Taïk and Cherkaoui (2020) demonstrates the potential of using federated settings in predicting electrical load consumption patterns and assisting load monitoring and energy demand response. The FedAvg technique is applied to aggregate the parameters of the LSTM models and produce a generalised global model. Similarly, FedAvg and LSTM are adopted in Savi and Olivadese (2021) to provide a generalised electrical load forecasting model. The authors use complementary features related to calendar and weather conditions along with the sequences of previous electrical loads to improve the forecast model.

Briggs et al. (2021) present a study on the importance of using smart meters in residential areas for short-term forecasting tasks using LSTM-based FL. Similarly, the contribution in Gholizadeh and Musilek (2022) evaluates the performance of FL versus centralised and local training methods when using LSTM models in electrical load forecasting. The study concluded that local learning is better suited to predict individual energy consumption than FL. However, FL is needed when a generalised forecasting model is required and access to aggregated data is impossible. Very recently, the research in Fekri et al. (2022) adopts the LSTM algorithm under the FL setting to forecast energy profiles. Two strategies are considered: federated stochastic gradient descent (FedSGD) and FedAvg to perform parameters aggregation. Experimental results demonstrated that FedAvg achieves better accuracy and requires fewer communication rounds.

The aforementioned FL-based energy forecasting studies rely on DL algorithms, specifically LSTM networks. Although LSTMs have been shown to achieve excellent prediction accuracy, they require intensive processing duties that yield a heavy computational burden. The following section provides a background to the main elements for conducting this study.

3. Preliminaries and background

In this section, we review the fundamentals of LSTM, LGBM, and FL needed to understand better the study conducted in this paper.

3.1. Long Short-Term Memory (LSTM) networks

An RNN is a class of artificial neural networks designed to process sequential data through which it recognises patterns, understands temporal dynamic behaviours, and provides predictions based on previous states (Sherstinsky, 2020). As in NNs, RNNs are gradient-based learning algorithms that rely on the backpropagation mechanism to update the weights of neurons. However, RNNs suffer from two main challenges associated with performing partial derivatives across the network to find the model weights: vanishing and exploding gradients. These problems may prevent the network from further training, causing the network to struggle to learn long-term dependencies. Therefore, LSTM networks were introduced as more robust models without being affected by the unstable gradients problem by improving the gradient flow (Sherstinsky, 2020). This was done by introducing artificial memory (cell state) and three gates: a forget gate, an input gate, and an output gate. The three gates can be thought of as filters that regulate the flow of information into and out of the cell to help predict the output sequence. A pictorial representation of an LSTM unit is given in Fig. 1.

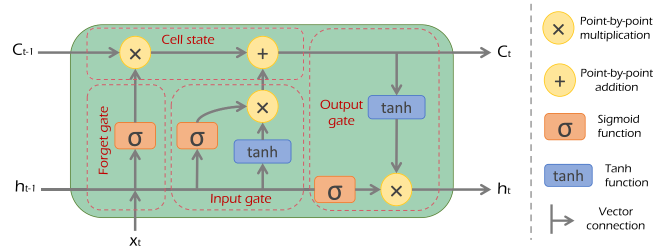

Fig. 1. LSTM memory cell with gating units.

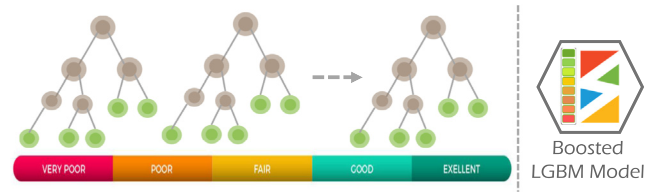

Fig. 2. Ensemble of DTs are combined to boost and form the LGBM model.

3.2. Light Gradient Boosting Model (LGBM)

DTs are supervised ML algorithms that can perform regression and classification by continually splitting data according to specific rules. The simplicity of DTs has made them popular, and they have been applied in many applications. However, DTs suffer from several challenges that limit their use in more complex situations, such as overfitting, instability, and bias. Therefore, the concept of ensemble learning was adopted with the aim of combining many weaker learners (i.e., DTs) to produce a more robust ensemble. Ensemble learning includes three main classes, bagging, stacking, and boosting. For more details on ensemble learning, we refer the reader to Ribeiro and dos Santos Coelho (2020).

In 2017, Microsoft introduced one of the most powerful boosting models named LGBM (Ke et al., 2017), depicted in Fig. 2. What distinguishes LGBM from other boosting models is its efficiency, fast processing, and scalability. These desirable features are gained by introducing two techniques: gradient-based one-side sampling (GOSS) and exclusive feature bundling (EFB). GOSS downsamples data instances with small absolute gradients while keeping the samples with higher absolute gradients because they contribute more to the training process. At the same time, EFB reduces the number of data features by bundling the mutually exclusive ones. Lower computations, less memory consumption, fast training, handling large-scale data, and scalability are the main characteristics that will help the prevalence of LGBM algorithms in solving many real-world problems.

3.3. Federated averaging

ML is a data-driven approach that relies on several quantities of datasets to train a model for a specific task. Conventionally, model

Algorithm 1 Federated Averaging (McMahan et al., 2016)

Require: Communications limit (T), Clients count (C), Number of mini-batches (B), Number of epochs (E)
Server executes:
  $w_0 \leftarrow$ initialise weights
  for $t = 1, 2, ..., T$ **do**
    S_t = Random set of C
    for $k = 1, 2, ..., S_t$ **in parallel do**
      $w_t^k \leftarrow$ ClientUpdate(k, w_{t-1})
    $w_t \leftarrow \sum_{k=1}^{S_t} \frac{n_k}{n} w_t^k$

ClientUpdate (k, w):
  for $e = 1, 2, ..., E$ **do**
    $batches \leftarrow$ Split dataset into B batches
    for $b = 1, 2, ..., batches$ **do**
      $w \leftarrow w - \eta \nabla l(w; b)$ *//Compute gradients and update weights*
  Return w to server

Algorithm 2 FedraTrees

Require: Communications rounds (R), Clients count (C), Number of trees per batch (T)
Server executes:
  for $r = 1, 2, ..., R$ **do**
    if $r == 1$ **then**
      Broadcast LGBM parameters values to clients
      $Batch^0 = \{\}$
    Broadcast $Batch^{r-1}$
    for $k = 1, 2, ..., C$ **in parallel do**
      Store $Batch^{r-1}$ in memory
      $Batch_k^r \leftarrow$ ClientUpdate($k, \sum_{i=0}^{r-1} Batch^i, r$)
    //Find the best Batch
    $Batch^r \leftarrow$ Server validation $\{\sum_{i=0}^{r-1} Batch^i + Batch_k^r\}_{k=1}^{C}$

ClientUpdate ($k, \sum_{i=0}^{r-1} Batch^i, r$):
  $Model_k^r \leftarrow \sum_{i=0}^{r-1} Batch^i + \sum_{t=1}^{T} DT_t$
  for $t = 1, 2, ..., T$ **do**
    Find the optimal split for each split node in the new batch trees and update $Model_k^r$
  Return $\sum_{t=1}^{T} DT_t$

training is performed centrally by collecting the required datasets at a central location. However, centralised training methods threaten data privacy and security and contradict the legislation in data protection laws. Therefore, FL has emerged as a promising solution that addresses privacy concerns by pushing the model to the locations where the data is generated and resides (Al-Quraan et al., 2021). Several aggregation algorithms have been introduced, such as FedAvg (McMahan et al., 2016) and FedDist (Sannara et al., 2021). However, these algorithms are designed to aggregate the parameters of NNs models and cannot be used with other ML algorithms, like DTs. Furthermore, the FL framework is highly dependent on the resources of clients and networks, and using NN models under the FL framework will lead to high consumption of these resources. Therefore, building new efficient aggregation schemes is crucial. To tackle this problem, this study proposes a novel tree-based FL framework termed FedraTrees, which will be discussed in detail in the next section. FedraTrees performance is compared with FedAvg, the most widely used aggregation algorithm in the literature.

FedAvg algorithm best fits neural networks (NNs), where network parameters (weights and biases) can be extracted and aggregated to form newly updated parameters. Three main parameters characterise FedAvg; a subset of total client count (C), breaking down the dataset into small-sized mini-batches (B), and the number of epochs (E) that the client passes over its dataset per round. FedAvg allows each client to perform multiple stochastic gradient descent rounds locally across different local data subsets and then find the optimal model parameters by averaging the locally evaluated gradients at the FL server. The complete pseudo-code of FedAvg is given in Algorithm 1. In addition, LSTM networks have been widely used in predicting time-series datasets, and they fit the federated optimisation problem. In previous studies that considered energy forecasting tasks under the FL environment, their primary model choice was certainly the LSTM networks. Although the prediction performance of LSTM models is accurate, these studies have overlooked crucial problems accompanied by FL distributed learning: the computation and communication costs.

4. Proposed FedraTrees algorithm

4.1. FedraTrees in LGBM-based FL

As mentioned earlier, in the federal environment, the use of NNs is becoming more and more popular. However, many other ML algorithms have not had the opportunity to be explored in such an environment, even though they conceal many wanted merits like simple but efficient approaches. Recently, the research community has begun to investigate the use of other ML techniques, such as DTs (Zhao et al., 2018; Liu et al., 2021). This study follows the same concept of employing DTs under the FL setting; however, it is the first research to explore the use of LGBM models with FL. This study is also the leading one in exploring LGBM-based FL in the context of smart grids. Inspired by the FedVoting algorithm presented in Liu et al. (2021), where the authors construct a federated GBDT model trained for human mobility prediction, this study proposes FedraTrees. FedraTrees framework harnesses the power of the LGBM algorithm within the FL environment. What distinguishes FedraTrees from FedVoting is that it is less complex and scalable. To be precise, FedVoting is developed for the cross-silo setup and relies on cross-validations. To determine the optimal model for each training round, each client must validate other clients' trained local models, incurring additional computation cost. Moreover, scalability is not by design of FedVoting since it is designed to be performed on a limited number of clients. By contrast, FedraTrees is developed to fulfil simplicity, efficiency, and scalability, while fitting both cross-device and cross-silo settings. In FedraTrees, the complexity is alleviated as there is no need for any client to validate others' models; the central server validates the received models and selects the best one to build upon in upcoming training rounds.

Fig. 3 demonstrates the structure of FedraTrees and the detailed training process. FedraTrees is a batch-based aggregation algorithm that determines the number of DTs per batch to be trained in each round and builts on it. Thus, besides deciding the number of trees per batch, constructing FedraTrees begins with choosing the best hyper-parameters of the LGBM model, like the number of leaves for each estimator/tree, boosting type, and max depth. Once determined, the central server broadcasts these parameters to each participating client. When every client receives the model parameters, they create the LGBM model accordingly and start the iterative model training. At the end of each training round, the clients only send the last trained batch of trees to the central server for evaluation. The server updates the global model by affixing a single batch of trees at a time and evaluating the model, and this process is repeated until all clients' batches have been tested. The batch of trees that achieves the best evaluation performance will be kept to build upon in the subsequent communication rounds, while the batches from other clients will be discarded. The server broadcasts the newly elected batch of trees to the clients to be appended to the previous version of the model and then starts a new training round. This process is repeated until the global LGBM model reaches the desired accuracy or no further improvement can be achieved. The complete FedraTrees pseudo-code is presented in Algorithm 2.

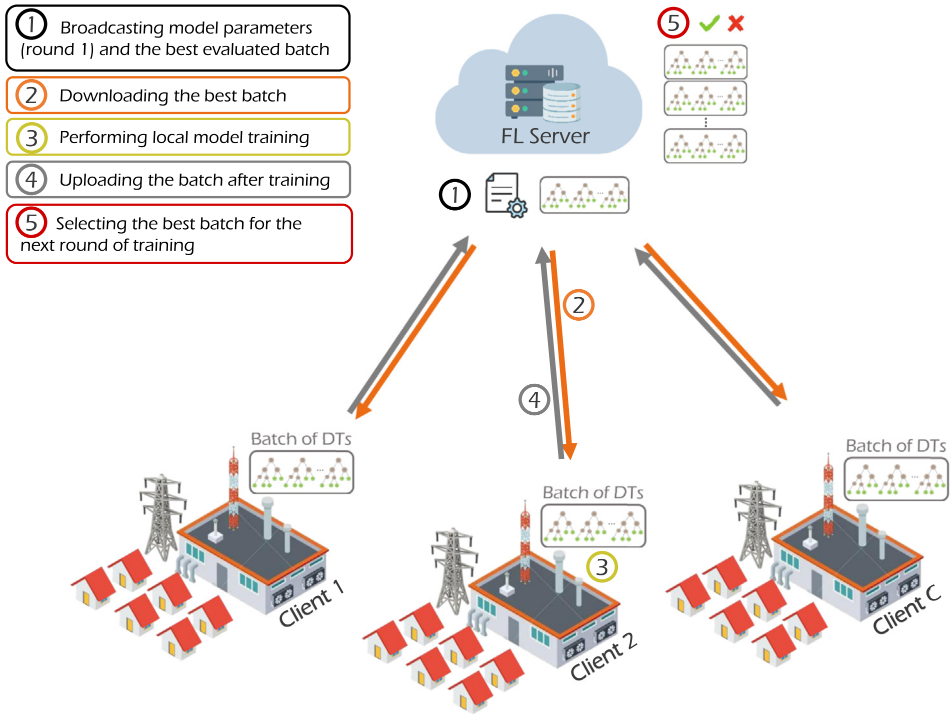

Fig. 3. FedraTrees sequential operation steps for the energy forecasting task considering C substations as clients.

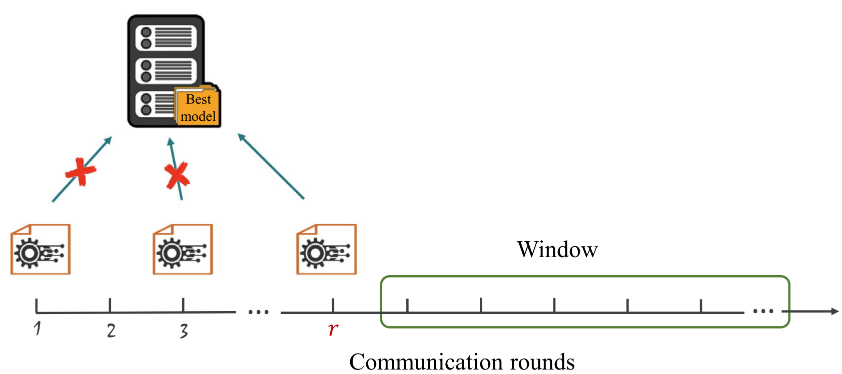

Fig. 4. Illustration of the delta-based FL stopping algorithm; the current r communication round has a better model that replaces the previous one, emptying the window.

4.2. Delta-based FL stopping

The challenge of training an ML model in the federated setting is choosing the number of communication rounds. A large number of rounds incur unnecessary computations and waste of resources, leading to overfitting. Whereas a small number of rounds would result in a suboptimal model suffering underfitting. In this study, we also consider this challenge by developing a stopping algorithm that monitors the evaluation performance of the trained model at every iteration and halts the FL training process when no further improvement is expected. This algorithm is inspired by the Scikit-Learn stopping criterion, which is developed to stop the training process within the model (Pedregosa et al., 2011). Instead of fixing the number of communication rounds, the delta-based FL stopping algorithm sets a threshold (delta) for comparing the best model of the previous rounds with the currently trained model based on the validation dataset, as depicted in Fig. 4. If the current model performs better than the preceding best-stored model, it will replace it; otherwise, the best model will remain. This comparison runs for several rounds defined as a window size; if the window is filled in without a better model being found, the training will stop and return the best model. We have implemented this algorithm on the FedAvg and FedraTrees frameworks after an extensive study to determine the best delta and window size values. More details will be provided in the following section.

5. Performance evaluation and results

5.1. Dataset pre-processing and evaluation methods

Since this study is intended to provide a practical framework capable of forecasting electricity load patterns in smart cities, it is essential to find an excellent dataset to evaluate the performance of FedraTrees versus FedAvg. In this regard, the Tetouan power consumption dataset is selected for this purpose (Salam and El Hibaoui, 2018). This dataset was collected in 2017 at three different distribution substations from the zones: Quads, Boussafou, and Smir in Tetouan, a city located in north Morocco. In addition to providing information on power consumption every 10 min, the Tetouan dataset offers complementary data on the calendar and weather conditions. To prepare this dataset for our study, we initially converted the time scale from 10 min to 60 min because we are interested in predicting short-term loads for the next hour. Furthermore, we created two new dataset features; the

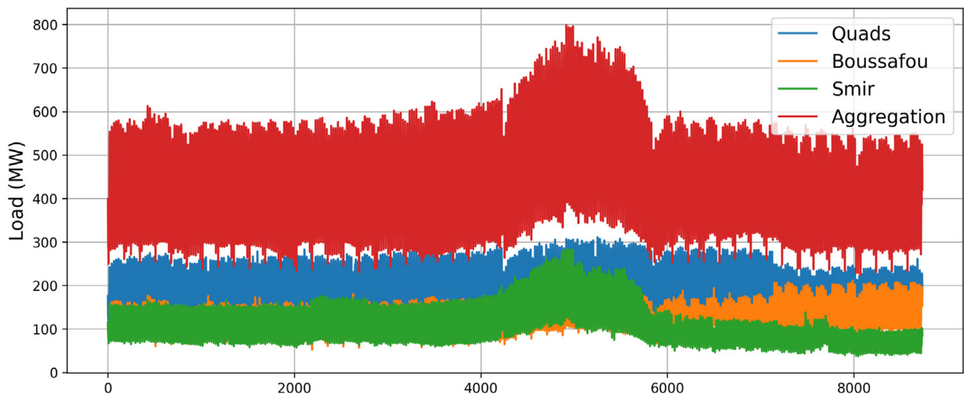

Fig. 5. Tetouan dataset preparation generated hourly power consumption of the three zones in addition to the aggregated power.

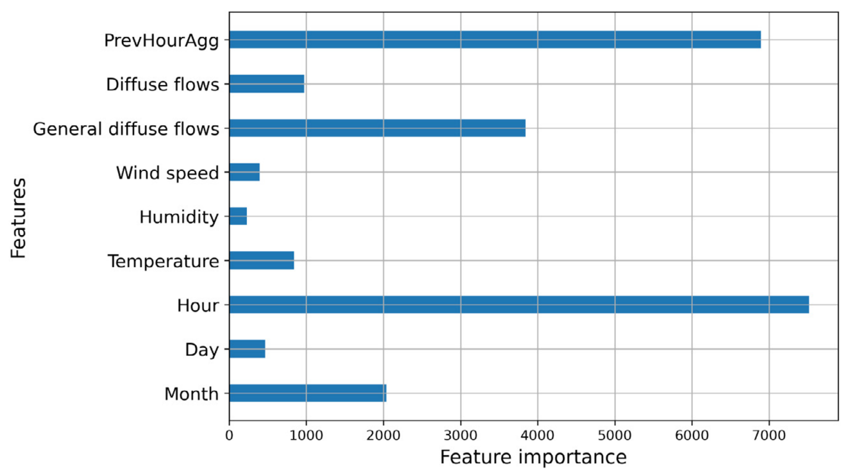

Fig. 6. The importance of each feature in forecasting power consumption.

Table 1
Tetouan dataset features used for load forecasting.

Context	Features
Calendar	① Month ② Day ③ Hour
Weather	④ Temperature ⑤ Humidity ⑥ Wind speed ⑦ Diffuse flow ⑧ General diffuse flow
Power	⑨ PrevHourAgg

aggregation feature that aggregates the power consumption of the three zones for use while performing centralised and distributed learning, and the previous hour aggregation (PrevHourAgg) feature that gives the aggregated feature reading of the previous hour. Fig. 5 demonstrates the hourly aggregated load as well as zones' load information. Moreover, Table 1 shows the features considered to perform load forecasting.

Before commencing the training process, the default scale of the features is normalised using MinMax scaler into the range [0,1]. Accordingly, all features have the chance of contributing equally to model fitting and avoiding biasing. For evaluation purposes, we use the most popular metrics of mean absolute error (MAE) and mean absolute percentage error (MAPE), which are defined as (Ahmad and Chen, 2018):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{y}_i\right| \tag{1}$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{2}$$

where y_i is the actual value, $\hat{y}_i$ is the predicted value, and n represents the number of data samples.

5.2. Simulation setup

To evaluate the effectiveness of our proposed FedraTrees framework in solving energy forecasting problems, we have conducted a comparative analysis against the widely adopted LSTM-based FedAvg algorithm. Furthermore, we have compared both algorithms with the Persistence approach, which predicts the current value to be identical to the preceding actual value. The forecasting problem is converted to a multivariate regression problem in which we exploit various calendar, weather, and power features to predict the power consumption for the next hour. For this purpose, we built LSTM and LGBM models based on the Random search strategy to discover the best combination of hyperparameters values. The LSTM model consists of a single hidden layer with 64 LSTM units that use the ReLU activation function and a dense output layer with one neuron. Also, it uses Adam optimiser for compilation, the dataset is divided into 80/20% train/test split ratio, batch size equals 30, and the number of epochs equals 300. Whereas the LGBM's boosting type, the number of trees,[1] max_depth, learning rate, num_leaves, and train/test split ratio are set to DART, 800, 12, 0.078, 30, and 80/20%, respectively. Our simulation experiments are based on Python programmes installed on a Windows operating system with Intel Xeon CPU E5-2620 @ 2 GHz and 16 GB RAM.

[1] This hyperparameter is replaced by batch in FedraTrees.

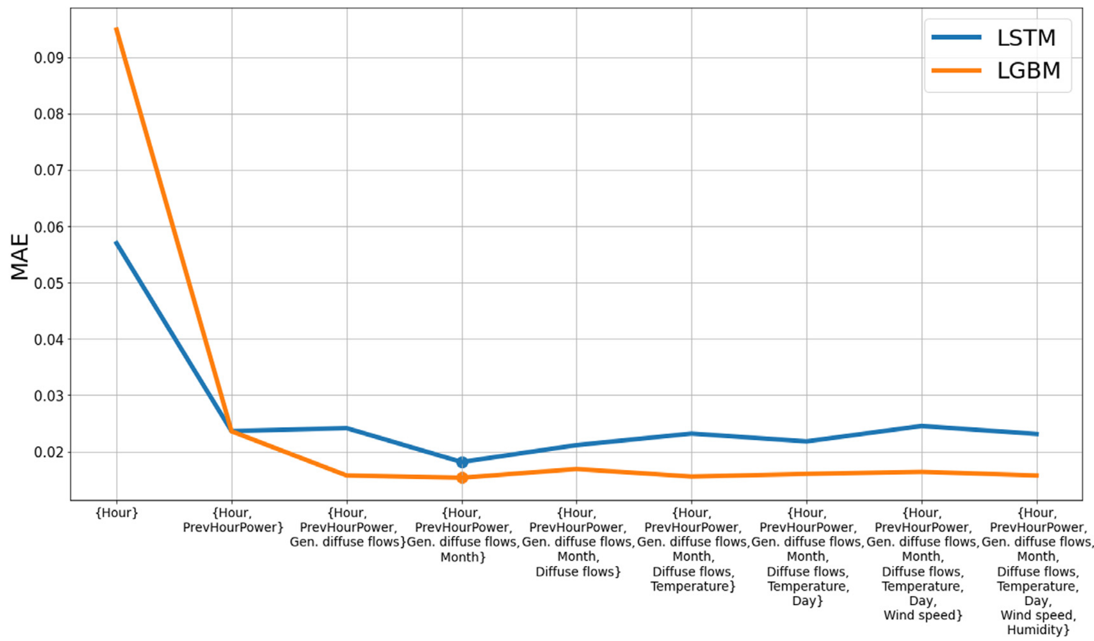

Fig. 7. MAE as a function of different features that contribute to the prediction process.

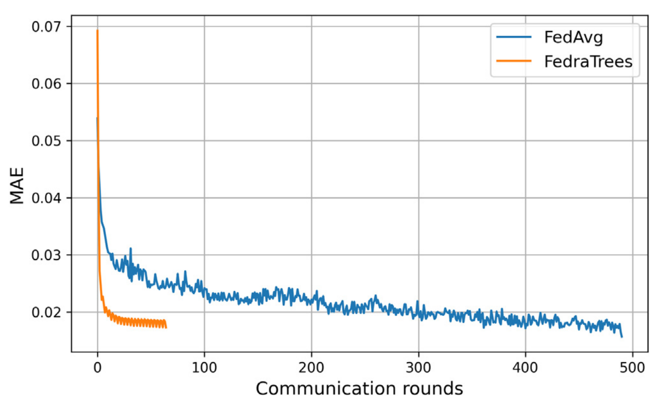

Fig. 8. MAE as a function of the communication rounds needed to train the global model of FedAvg and FedraTrees.

Table 2
Performance comparison between LSTM and LGBM models when performing centralised model training.

ML model	MAE	MAPE	Computation time
LSTM	0.019	3.04%	77 s
LGBM	0.017	2.69%	2 s

5.3. Numerical results

We first investigate the performance of the selected models when performing the conventional approach of centralised training. Table 2 shows the results of the evaluation metrics, as well as the computation time required for each model. The MAE and MAPE for the LSTM model are 0.02 and 3.04%, respectively, while they are improved when using the LGBM model and reached 0.017 and 2.69%, respectively. The fast computing merit of the LGBM model is confirmed in this table, which shows that it needs only two seconds to converge, while the LSTM model requires more than 97% computations compared to the LGBM model.

Furthermore, we conducted a study to explore the impact of every feature in predicting the target value. The LGBM model is equipped with a feature importance tool that can be used to find out what features contribute the most to the prediction of power values. Fig. 6 demonstrates the results of the feature importance study from which we can conclude that Hour, PrevHourAgg, General diffuse flows, and Month have the most impact in predicting the power consumption values. Other features also contribute to the prediction but their contribution is less noticeable. Furthermore, we conducted a study to examine the prediction performance when using several numbers of features for the LSTM and LGBM models. Fig. 7 shows the best achieved MAE versus using different numbers of features which are ordered based on their rank in the feature importance study. This figure indicates that, in general, multivariate prediction gives improved performance over univariate; however, the best performance for both models is obtained when the four most important features are used.

Moving on to the FL setup, it is worth recalling that FedraTrees is designed to accommodate cross-device and cross-silo settings. To the best of the authors' knowledge, FL has not been previously applied at the substation level in energy/power forecasting studies. Therefore, in this study, we focus on applying FedraTrees and FedAvg at the substation level. The FL setup for both algorithms comprises a central orchestrating server and three clients representing the three zones of the Tetouan dataset. The number of communication rounds is not fixed since we use the delta-based FL stopping technique, discussed in Section 4.2, to find the best round that produces the optimal trained model while reducing the computation and communication costs. An extensive study is carried out to determine the best delta and window size values for the stopping algorithm; the findings of this study are given in Tables 3 and 4. From Table 3, the most computationally efficient and the best MAE/MAPE for the LSTM model are obtained when the delta and window size values are 0.00001 and 55, respectively. Regarding LGBM, Table 4 demonstrates that the best values for the delta and window size are 0.00001 and 10, respectively. Table 5 summarises the best results obtained for each of the Persistence, FedAvg, and FedraTrees algorithms. The performance of the Persistence model is poor compared to other algorithms, and the FedAvg has the best performance, which is slightly better than that of FedraTrees; however, FedraTrees outperforms the FedAvg algorithm in terms of the communication rounds and the required computations. FedraTrees only requires 65 rounds of communications that result in approximately 26 s of computations, while FedAvg requires a much higher number of

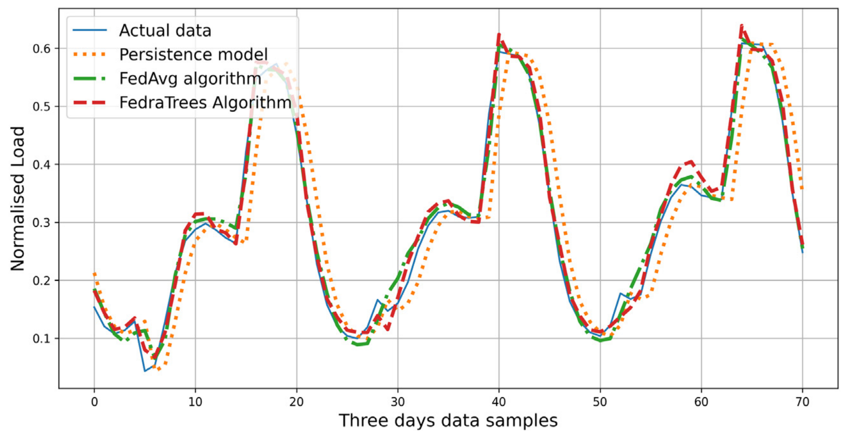

Fig. 9. Forecasting power consumption for three days.

Table 3
A study to determine the best values of delta and window size for LSTM-based FedAvg.

Delta	Window size	MAE	MAPE %	No. of rounds	Computation time (sec)
0.001	45	0.0217	4.62	106	214.2
0.0001	45	0.0214	4.55	133	277.5
0.00001	45	0.0171	3.72	401	1284
0.000001	45	0.0171	3.72	401	1284
0.001	55	0.0217	4.62	106	214.2
0.0001	55	0.0195	4.2	235	787
0.00001	**55**	**0.0157**	**3.43**	**491**	**1356**
0.000001	55	0.0157	3.43	491	1356
0.001	65	0.0217	4.62	106	214.2
0.0001	65	0.0157	3.43	491	1383.6
0.00001	65	0.0157	3.43	491	1383.6
0.000001	65	0.0157	3.43	491	1383.6

Table 4
A study to determine the best values of delta and window size for LGBM-based FedraTrees.

Delta	Window size	MAE	MAPE%	No. of rounds	Computation time (sec)
0.001	5	0.0199	4.25	8	2.0
0.0001	5	0.0177	3.79	26	7.6
0.00001	5	0.0175	3.75	35	10.3
0.000001	5	0.0175	3.75	35	10.3
0.001	10	0.0186	3.99	14	4.5
0.0001	10	0.0174	3.7	50	18.8
0.00001	**10**	**0.0173**	**3.69**	**65**	**26.2**
0.000001	10	0.0173	3.69	65	26.2
0.001	15	0.0186	3.99	14	4.5
0.0001	15	0.0174	3.7	50	18.8
0.00001	15	0.0173	3.69	65	27.2
0.000001	15	0.0173	3.69	65	27.2

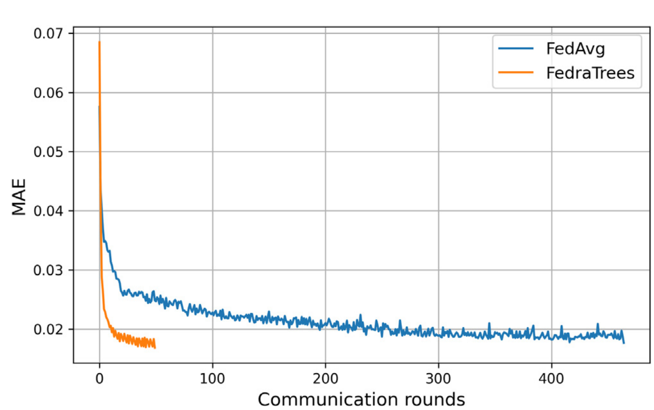

Fig. 10. MAE as a function of the communication rounds needed to train the global model of FedAvg and FedraTrees when considering only the top four features.

communication rounds and computation time by a factor of 7.6 and 52.2, respectively, to achieve the same level of performance.

Furthermore, Fig. 8 shows the convergence curve of MAE for both algorithms during the training of the global model. In addition, Fig. 9 shows the actual and the predicted power consumption of both algorithms and the baseline Persistence model. These figures indicate that FedraTrees converges faster and achieves an outstanding performance compared to FedAvg.

Another study was conducted that looked only at the four most important features to see the impact of using fewer features on the required number of communication rounds, the computation time, and the model performance. Table 6 gives the outcomes of this study and indicates that the performance of FedraTrees is improved when removing the less important features. However, this is not the case with FedAvg, as this table shows that its performance is slightly degraded compared to using all the features. The number of communication rounds is slightly less than that in the full feature study for both algorithms; however, this study also shows the outstanding performance of FedraTrees

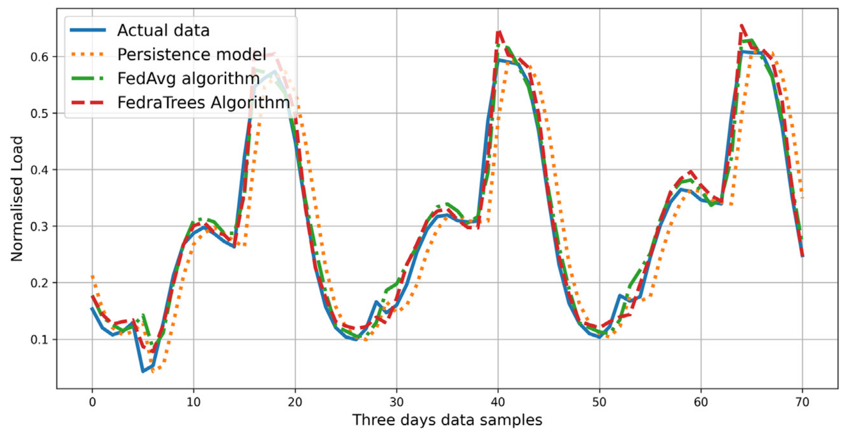

Fig. 11. Forecasting power consumption for three days when considering only the top four features.

Table 5
Performance results of the FedraTrees compared to the FedAvg and the Persistence model.

Algorithm	MAE	MAPE%	No. of rounds	Computation time
Persistence	0.08	6.64	N/A	N/A
FedAvg	0.0157	3.43	491	1356 s
FedraTrees	0.0173	3.69	65	26.2 s

Table 6
Performance results of the FedraTrees compared to the FedAvg and Persistence models when considering only the top four features.

Algorithm	MAE	MAPE%	No. of rounds	Computation time
Persistence	0.08	6.64	N/A	N/A
FedAvg	0.0177	3.93	465	1293 s
FedraTrees	0.0168	3.54	50	8.8 s

as it requires far fewer rounds of communications and computation costs. Similarly, Figs. 10 and 11 give the MAE convergence curve and the actual and forecasted power consumption for both algorithms, respectively. Also, these figures ensure the superb overall performance of FedraTrees.

6. Conclusions

This paper aimed at developing FedraTrees, a novel framework that incorporates ensemble learning, specifically the LGBM model, within the FL settings and coined to accommodate smart cities. Utilising the LGBM model transforms the FL into a highly efficient, fast processing, and scalable framework. Furthermore, instead of following the conventional fixed number of communication rounds method in FL, we developed a delta-based FL stopping algorithm that monitors and stops the FL training process when no further enhancement is possible, thus ensuring achieving the desired training accuracy with the minimal use of computation and communication resources. FedraTrees is employed for the energy demand forecasting problem and benchmarked against LSTM-based FedAvg and Persistence model. The simulation results demonstrated that FedraTrees has a remarkable performance in predicting short-term energy patterns and requires much less computation and communication than FedAvg with just 2% and 13%, respectively.

CRediT authorship contribution statement

Mohammad Al-Quraan: Methodology, Data curation, Simulation, Validation, Analysis, Writing. **Ahsan Khan:** Reviewing. **Anthony Centeno:** Reviewing. **Ahmed Zoha:** Methodology, Conceptualization, Review & editing. **Muhammad Ali Imran:** Reviewing. **Lina Mohjazi:** Methodology, Conceptualization, Review & editing.

Declaration of competing interest

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

Data availability

The data used in this research is the publicly available Tetouan Power Consumption dataset.

Acknowledgement

All authors approved the version of the manuscript to be published.

References

Abbasi, R.A., et al., 2019. Short term load forecasting using XGBoost. In: Proc. Int. Conf. Adv. Inf. Netw. Appl.. Springer, pp. 1120–1131.

Abdel-Basset, M., et al., 2021. Energy-net: a deep learning approach for smart energy management in IOT-based smart cities. IEEE Internet Things J. 8 (15), 12422–12435.

Ahmad, T., Chen, H., 2018. Short and medium-term forecasting of cooling and heating load demand in building environment with data-mining based approaches. Energy Build. 166, 460–476.

Ahmad, M.W., Mourshed, M., Rezgui, Y., 2017. Trees vs neurons: Comparison between random forest and ANN for high-resolution prediction of building energy consumption. Energy Build. 147, 77–89.

Al-Quraan, M., et al., 2021. Edge-native intelligence for 6G communications driven by federated learning: A survey of trends and challenges. arXiv preprint arXiv:2111.07392, URL http://arxiv.org/abs/2111.07392.

Briggs, C., Fan, Z., Andras, P., 2021. Federated learning for short-term residential energy demand forecasting. arXiv preprint arXiv:2105.13325 URL http://arxiv.org/abs/2105.13325.

Chui, K.T., Lytras, M.D., Visvizi, A., 2018. Energy sustainability in smart cities: Artificial intelligence, smart monitoring, and optimization of energy consumption. Energies 11 (11), 2869.

Ciulla, G., D'Amico, A., 2019. Building energy performance forecasting: A multiple linear regression approach. Appl. Energy 253, 113500.

Darbellay, G.A., Slama, M., 2000. Forecasting the short-term demand for electricity: Do neural networks stand a better chance? Int.L J. Forecast. 16 (1), 71–83.

Fekri, M.N., Grolinger, K., Mir, S., 2022. Distributed load forecasting using smart meter data: Federated learning with Recurrent Neural Networks. Int. J. Elect. Power Energy Syst. 137, 107669.

Gholizadeh, N., Musilek, P., 2022. Federated learning with hyperparameter-based clustering for electrical load forecasting. Internet Things 17, 100470.

Gohar, A., Nencioni, G., 2021. The role of 5G technologies in a smart city: The case for intelligent transportation system. Sustain. 13 (9), 5188.

Himeur, Y., et al., 2020. Building power consumption datasets: Survey, taxonomy and future directions. Energy Build. 227, 110404.

Ju, F.-Y., Hong, W.-C., 2013. Application of seasonal SVR with chaotic gravitational search algorithm in electricity forecasting. Appl. Math. Model. 37 (23), 9643–9651.

Ke, G., et al., 2017. LightGBM: A highly efficient gradient boosting decision tree. Adv. Neural Inf. Process. Syst. 30.

Koprinska, I., Wu, D., Wang, Z., 2018. Convolutional neural networks for energy time series forecasting. In: Proc. Int. Joint Conf. Neural Netw. (IJCNN), Rio de Janeiro, Brazil. pp. 1–8.
Kuo, P.-H., Huang, C.-J., 2018. A high precision artificial neural networks model for short-term energy load forecasting. Energies 11 (1), 213.
Liu, S., et al., 2018. Short-term load forecasting based on GBDT combinatorial optimization. In: Proc. IEEE Conf. Energy Internet and Energy Syst. Integr. (EI2), Beijing, China. pp. 1–5.
Liu, Y., et al., 2021. FedVoting: A cross-silo boosting tree construction method for privacy-preserving long-term human mobility prediction. Sensors 21 (24), 8282.
McMahan, H.B., et al., 2016. Federated learning of deep networks using model averaging. arXiv preprint arXiv:1602.05629 URL http://arxiv.org/abs/1602.05629.
Ozturk, S., Ozturk, F., 2018. Forecasting energy consumption of Turkey by ARIMA model. J. Asian Sci. Res. 8 (2), 52–60.
Pedregosa, F., et al., 2011. Scikit-learn: Machine learning in Python. J. Mach. Learn. Res. 12, 2825–2830.
Ribeiro, M.H.D.M., dos Santos Coelho, L., 2020. Ensemble approach based on bagging, boosting and stacking for short-term prediction in agribusiness time series. Appl. Soft Comput. 86, 105837.
Saab, S., Badr, E., Nasr, G., 2001. Univariate modeling and forecasting of energy consumption: the case of electricity in Lebanon. Energy 26 (1), 1–14.
Salam, A., El Hibaoui, A., 2018. Comparison of machine learning algorithms for the power consumption prediction:Case study of Tetouan city. In: Proc. 2018 6th Int. Renew. Sustain. Energy Conf. (IRSEC), Rabat, Morocco. IEEE, pp. 1–5.
Sannara, E., et al., 2021. A federated learning aggregation algorithm for pervasive computing: Evaluation and comparison. In: Proc. IEEE Int. Conf. Pervasive Comput. Commun. (PerCom), Kassel, Germany. pp. 1–10.
Savi, M., Olivadese, F., 2021. Short-term energy consumption forecasting at the edge: A federated learning approach. IEEE Access 9, 95949–95969.
Sherstinsky, A., 2020. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. Physica D 404, 132306.
Singh, B., Pozo, D., 2019. A guide to solar power forecasting using ARMA models. In: Proc. IEEE PES Innov. Smart Grid Technol. Eur. (ISGT-Europe), Bucharest, Romania. pp. 1–4.
Taïk, A., Cherkaoui, S., 2020. Electrical load forecasting using edge computing and federated learning. In: Proc. IEEE Int. Conf. Commun. (ICC), Dublin, Ireland. pp. 1–6.
Tso, G.K., Yau, K.K., 2007. Predicting electricity energy consumption: A comparison of regression analysis, decision tree and neural networks. Energy 32 (9), 1761–1768.
Wang, J.Q., Du, Y., Wang, J., 2020. LSTM based long-term energy consumption prediction with periodicity. Energy 197, 117197.
Waqas, M., et al., 2022. The role of artificial intelligence and machine learning in wireless networks security: principle, practice and challenges. Artif. Intell. Rev. 1–47.
Yan, K., et al., 2019. A hybrid LSTM neural network for energy consumption forecasting of individual households. IEEE Access 7, 157633–157642.
Yu, Z., et al., 2010. A decision tree method for building energy demand modeling. Energy Build. 42 (10), 1637–1646.
Zancanaro, J.A., et al., 2019. Energy consumption forecasting using SARIMA and NARNET: An actual case study at university campus. In: Proc. IEEE PES Innov. Smart Grid Technol. Conf.-Latin America (ISGT Latin America), Gramado, Brazil. IEEE, pp. 1–6.
Zhao, L., et al., 2018. Inprivate digging: Enabling tree-based distributed data mining with differential privacy. In: Proc. IEEE INFOCOM Conf. Comput. Commun.. pp. 2087–2095.