TYPE Original Research
PUBLISHED 09 January 2024
DOI 10.3389/fenrg.2023.1340639

Decentralized asynchronous adaptive federated learning algorithm for securely prediction of distributed power data

Qiang Li[1], Di Liu[1], Hui Cao[2]*, Xiao Liao[1], Xuanda Lai[2] and Wei Cui[1]

[1]State Grid Information & Telecommunication Group Co., Ltd., Beijing, China, [2]School of Electrical Engineering, Xi'an Jiaotong University, Xi'an, Shaanxi, China

Check for updates

OPEN ACCESS

EDITED BY
Fuqi Ma,
Xi'an University of Technology, China

REVIEWED BY
Keyu Wu,
Institute for Infocomm Research (A*STAR), Singapore
Liao Wu,
University of New South Wales, Australia

*CORRESPONDENCE
Hui Cao,
huicao@mail.xjtu.edu.cn

RECEIVED 18 November 2023
ACCEPTED 21 December 2023
PUBLISHED 09 January 2024

CITATION
Li Q, Liu D, Cao H, Liao X, Lai X and Cui W (2024), Decentralized asynchronous adaptive federated learning algorithm for securely prediction of distributed power data.
Front. Energy Res. 11:1340639.
doi: 10.3389/fenrg.2023.1340639

Introduction: Improving the precision and real-time speed of electricity data prediction while safeguarding data privacy and security holds immense significance for all power system participants' decision-making. To surmount the issues of exorbitant computational expenses and privacy breaches of traditional centralized prediction methods, this paper proposes a decentralized asynchronous adaptive federated learning algorithm for securely prediction of distributed power data, which makes predictions from distributed data more flexible and secure.

Methods: First, each regional node trains its own deep neural network model locally. After that, the node model parameters are uploaded to the decentralized federated learning chain for ensuring local data protection. Asynchronous aggregated update of the global prediction model is then achieved via block mining and shared maintenance. The algorithm has been enhanced based on the traditional federated learning algorithm, which introduces an asynchronous mechanism while adaptively adjusting the regional node model weights and local update step size to overcomes the inefficiency of traditional methods.

Results and Discussion: The experimental analysis of actual electricity price data is conducted to compare and analyze with the centralized prediction model, study the impact of model adoption and parameter settings on the results, and compare with the prediction performance of other federated learning algorithms. The experimental results show that the method proposed in this paper is highly accurate, efficient, and safe.

KEYWORDS

time-series data prediction, distributed learning, federated learning, decentralization, privacy preservation

1 Introduction

Power time-series data has a significant impact on power system operation, planning and decision making. It can support decision making and optimization in load forecasting and dispatching, fault detection and handling, energy planning and market trading. However, with the development of distributed energy resources, the rise of microgrids and the Energy Internet, and the application of intelligent and digital technologies, several factors have contributed to making power systems become distributed (Liu et al., 2021). Meanwhile, power industries, which are a vital part of national energy security, have data on energy supply and demand, grid stability, etc. Therefore, it is necessary to develop an efficient method applicable to the analysis and prediction of distributed power data, which can reduce the cost of data transmission and processing, and also ensure the security of private data in the power industry (Ali et al., 2023).

Currently, researchers have conducted relevant studies on the analysis and prediction of time series data in distributed power systems. The primary methods employed encompass traditional time series analysis and intelligent data mining techniques. In the time series approach, statistical analysis is conducted on power time series data to identify its characteristics, patterns, and trends, which are then utilized to derive predictive values (Badhiye et al., 2022; Frizzo et al., 2023). Some of the more classical time series models include Autoregressive (AR), Moving Average (MA), and Autoregressive Moving Average (ARMA). Furthermore, data mining methods are employed to capture potential non-linear relationships in the formation process of power time series data, thus enhancing prediction accuracy (Wang et al., 2023; Zhang et al., 2023). Commonly utilized intelligent methods include Random Forest (RF), Support Vector Machine (SVM), Deep Neural Network (DNN), Recurrent Neural Network (RNN), Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU), among others.

The above methods for predicting electricity time series data usually adopt a centralized computing model, collecting all the data and concentrating them in a central node for model training and prediction analysis. During the training of the centralized predictive model, all data is transmitted and stored on the central server, the model is trained on the central server, and the update of the model is applied directly on the central server (Ahmed et al., 2022). However, the transmission of large amounts of data during centralised learning poses a risk to user privacy (Mcmahan et al., 2017). Meanwhile, the centralized mode has some problems, such as high data acquisition cost, low real-time performance, large resource consumption, and insufficient adaptability to the rapid change of the distributed power system (Li et al., 2020a).

To overcome the shortcomings of centralized tariff prediction models, some studies have applied techniques such as distributed computing and federated learning to power systems, which delegate data processing and analysis tasks to multiple edge nodes to improve efficiency, real-time performance and security. Federated learning methods can perform model training and parameter updating without exposing data and achieve better privacy results. For example, in FedAvg (Mcmahan et al., 2017), participating nodes perform local training and upload model parameters to the server, which performs parameter aggregation and model updating and distributes the global model to nodes, iteratively performing the above steps until convergence.

However, traditional federated learning methods do not perform well in the presence of heterogeneity in the system (Li et al., 2020b). In a fluctuating and undulating distributed power system, the conditions in each region are heterogeneous, and there may be problems such as inconsistent data distribution in each region, different data volumes, different data transmission efficiencies, and different computing power (Yu et al., 2019; Zeng et al., 2023). If an inappropriate data analysis and prediction method is used, the results obtained will have a negative impact on strategy analysis and decision making.

To address the limitations of the above methods, this paper proposes a decentralized asynchronous adaptive federated learning algorithm (DAAFed) for securely prediction of distributed power data. The main contributions of this study are summarized as follows.

1) A decentralized distributed framework has been developed for predicting regional time series data. Each regional node trains its own local data prediction model using its unique data. Next, the parameters of the deep network model are uploaded. With the use of blockchain technology, the aggregation update of the global prediction model is realized through the generation of blocks and the common maintenance of block sets.
2) The conventional federated learning algorithm is improved by the integration of a time-synchronous mechanism, which adopts the adaptive adjustment of the regional node model weight and the local update step size, thereby improving the accuracy and efficiency of distributed data prediction.
3) The proposed method will be validated using measured data from different regions in Spain. By comparing and analyzing the results with a centralized prediction model, investigating the impact of model adoption and parameter settings on the results, and comparing the prediction performance with other federated learning algorithms, the experimental results validate that the proposed method is able to balance safety, accuracy and efficiency.

The rest of this article is organized as follows. Sections 2, 3 present a review of related work and the basic model for power time series data prediction, respectively. The decentralized asynchronous adaptive federated learning method proposed in this article is explained in Section 4. Then, in Section 5, we perform simulation experiments to verify the effectiveness of the method proposed in this paper. Finally, Section 6 concludes this article.

2 Related work

In distributed machine learning, federated learning, as an emerging artificial intelligence technology, can ensure data privacy while performing efficient machine learning, providing a new way to solve the “data silo” problem. Federated learning algorithms have been gradually applied to various problems, such as healthcare (Chen et al., 2023), communication (Qu et al., 2023), language modelling (Wu et al., 2020), transportation (Qi et al., 2023), etc. Federated learning was proposed by Google in 2017, where global models are trained through the cooperation of edge devices without sharing training data. In this approach, training is performed by edge devices, and the weights of the training results are shared with a central server to perform weight updates. And the updated weights are then sent back to the edge devices for a new round of training.

The basic federated learning methods are only suitable for certain environments, so there are also some improved federated learning methods that can be useful in different environments. A federated learning approach in conjunction with deep autoencoder networks based on representation learning was proposed by Husnoo et al. (2023) to enable monitoring and data collection subsystems in distributed grid regions to collaboratively train attack detection models for accurate detection of power system and network security anomalies without sharing sensitive power-related data. The K-means was used to cluster power data locally at the utility, followed by federated learning to build accurate power prediction models for each class of power data in conjunction with other local clients (He and Zhao, 2022). A combined federated learning and deep reinforcement learning scheme for ultra-short term wind power prediction has been proposed by Li et al. (2023a), which uses a deep deterministic policy gradient algorithm as the basic prediction model to improve prediction accuracy, and then integrates the prediction model into the federated learning framework.

Recently, blockchain-based federated learning approaches have become increasingly popular. Blockchain provides tamper-proof and non-repudiation guarantees for the data stored in its ledger. Therefore, storing local and global models in the blockchain helps to improve the overall integrity of the system. Blockchain nodes can also collaborate to maintain the stored data and detect any malicious intentions of the participants. In addition, the execution of smart contracts is deterministic, which allows the federated learning process to be executed correctly and fairly. Intelligent encryption was redesigned to take advantage of the decentralized nature of blockchain technology to design collaborative intelligent federated learning frameworks for automated diagnosis without violating the trustworthiness metrics of privacy, security and data sharing encountered in smart city healthcare systems (Mohamed et al., 2023). Privacy-sensitive energy data can be stored locally at edge producer-consumer nodes without disclosure to third parties, and only the learned local model weights are shared using a blockchain network. Smart contracts are also used to handle the integration of local machine learning predictive models with the blockchain, enabling model parameter scaling and reducing blockchain overhead (Antal et al., 2022). A blockchain federated learning based object detection scheme was proposed by Li et al. (2023b), which eliminates central authority by using a distributed InterPlanetary File System (IPFS). The global model is periodically aggregated when multiple local model parameters are uploaded to the IPFS. Nodes can retrieve the global model from the IPFS. This method has been used in face detection, animal detection, unsafe content detection, vehicle detection, etc.

3 Preliminary

The characteristics and patterns of electricity time series data are intricately linked to the geographical location and regional environment of each distributed area. Factors such as electricity demand, load peaks and troughs, weather conditions, power generation structure, and network configuration in different regions all contribute to the distinct characteristics of electricity data. To address the challenge of predicting electricity time series data in diverse regions and to more effectively capture the non-linear relationships and data features of electricity prices, this study adopts a deep neural network model for accurate predictions at the edge nodes. Specifically, we focus on investigating and analyzing the commonly used CNN and LSTM, as well as their combination, to achieve accurate forecasting.

Let Ω_k represent the model of the kth node, and the sample data used for training the prediction includes various feature data such as load information L_k, generation information G_k, weather information W_k, date information D_k, along with the corresponding labeled data for prediction target P_k. The representation can be expressed as Eqs 1, 2:

$$\boldsymbol{x_k} = [\boldsymbol{L_k}, \boldsymbol{G_k}, \boldsymbol{W_k}, \boldsymbol{D_k}] \tag{1}$$

$$\boldsymbol{y_k} = [\boldsymbol{P_k}] \tag{2}$$

Training of each node is based on both the model as well as the sample data. The prediction accuracy is higher when the value of the loss function is small. The commonly used loss function, Root Mean Square Error (RMSE), can be expressed as in Eq. 3:

$$\boldsymbol{l}(\boldsymbol{\Omega}(\boldsymbol{x_k}), \boldsymbol{y_k}) = \sqrt{\frac{\boldsymbol{1}}{\boldsymbol{n}}\sum_{i=1}^{n} (\boldsymbol{\Omega}(\boldsymbol{x_i}) - \boldsymbol{y_i})^2} \tag{3}$$

where n is the total number of samples at the kth node, $x_i \in x_k$, $y_i \in y_k$, and $\Omega(x_i)$ is the prediction output of the model.

Considering the efficient utilization of computing resources in each region and the paramount importance of data security, this study adopts the federated learning framework for model training and parameter updating in the region-specific prediction. Each edge node possesses data transmission and computation capabilities, making it advantageous to delegate computation tasks to the edge nodes, thereby ensuring accurate and efficient time series data prediction in a trustworthy environment.

In contrast to the traditional centralized learning model that uploads raw data to a central server for model training, the federated learning framework assigns the model training task to distributed local devices. Subsequently, the global model is updated on the server side by exchanging certain parameters. The process of prediction for each region using the federated learning framework comprises the following steps:

1) The server defines the general task of prediction, including the determination of global variables such as the choice of prediction model, training rounds, and aggregation rounds.
2) As in Eq. 4, each node determines the initial local model based on the global model ω. The node then trains the local model Ω_k using local data, encompassing generation, load, climate, and date information. The node then uploads the parameters v_k, such as local model weights, to the server.

$$\boldsymbol{v_k} \leftarrow \boldsymbol{LocalUpdate}(\boldsymbol{\omega}, \boldsymbol{x_k}, \boldsymbol{y_k}), \boldsymbol{k} = \boldsymbol{1}, \boldsymbol{2}, \ldots, \boldsymbol{K} \tag{4}$$

3) As in Eq. 5, the server globally aggregates the model parameters from each node according to the specified mechanism to construct a new global model ω.

$$\boldsymbol{\omega} \leftarrow \boldsymbol{GlobalUpdate}(\boldsymbol{v_1}, \boldsymbol{v_2}, \ldots, \boldsymbol{v_K}) \tag{5}$$

4) Each node receives the latest global model as the initial model for the next round of training and iteratively performs steps 2) and 3) until the model prediction reaches the desired accuracy.

4 Methodology

In this section, a decentralized asynchronous adaptive federated learning algorithm is proposed, which is oriented towards the secure prediction of distributed power data. As shown in Figure 1, the

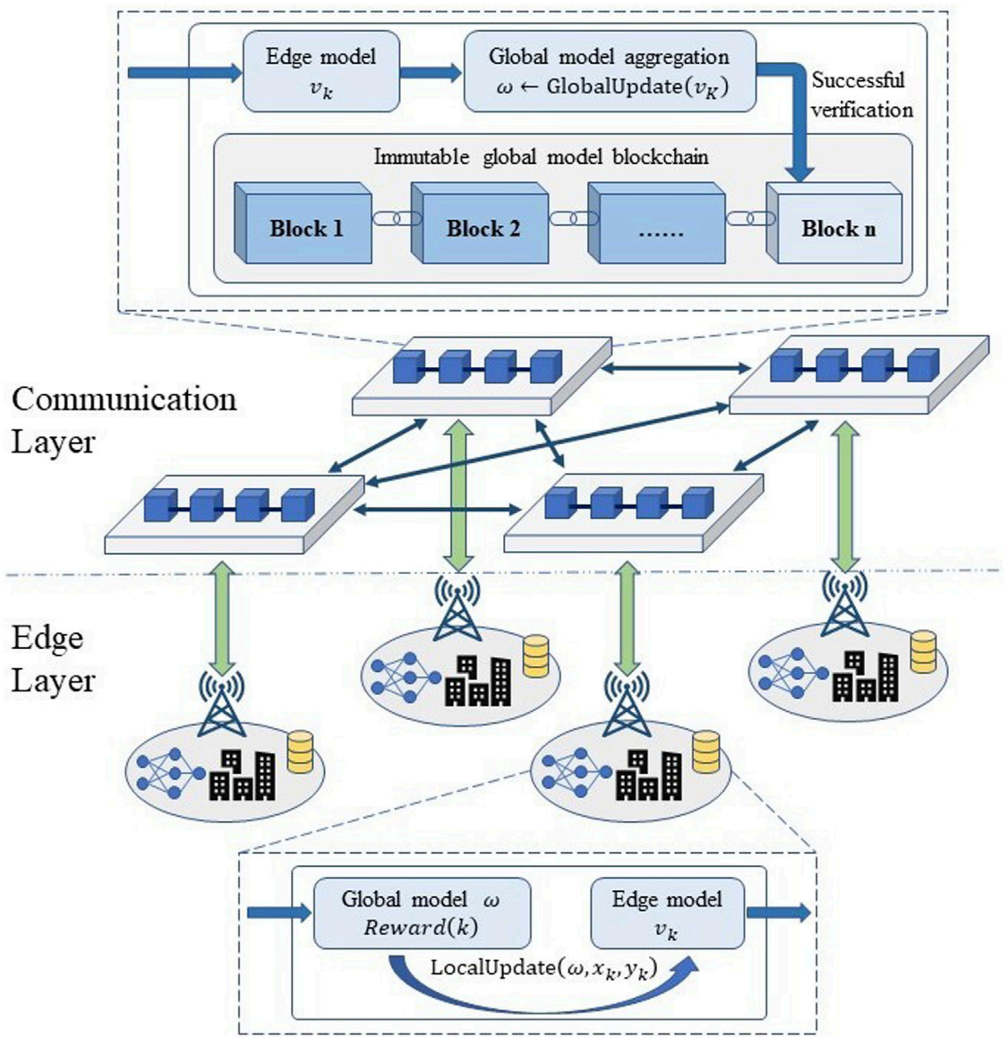

FIGURE 1
Distributed regional electricity price forecasting framework.

distributed regional time-series data prediction framework designed in this paper has a two-layer structure. The edge layer consists of each actual compute node, including the local power network and real-time data. The communication layer is the communication link between nodes, which is used to aggregate and update the model under the cooperation. First, each regional node locally trains its own deep network model Ω_k and uploads the node prediction model parameters v_k to the federated learning chain to realize local data protection. Then, the global model ω is updated asynchronously based on the distributed ledger mechanism in blockchain technology, and the aggregation of global models is realized by block mining, and each regional node obtains the latest global model ω through the shared immutable blockchain for subsequent prediction tasks.

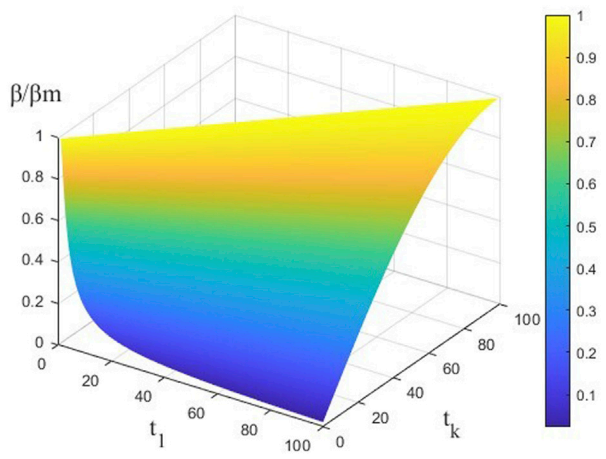

FIGURE 2
Weight adaptive adjustment function.

4.1 Traditional federated learning algorithm

Within the federated learning framework, each participating node conducts local training and uploads model parameters to the server. The

TABLE 1 Pseudocode of DAAFed.

Algorithm	DAAFed
Input:	K (number of the involved edge nodes) e (max number of iterations of the local model)
	E (max number of aggregations of the global model)
	S_k (data samples of each node)
Output:	v_k (local model parameter of each node)
	ω (global model parameter)
1	set global model parameters ω to initial values, set $v_k = \omega$
2	for $t = 0{:}\ (E-1)$ do
3	# Parallel iterative phase at the edge nodes
4	parfor $k = 1{:}\ K$ do
5	node#k updates the step-size η as in Eq. 28
6	node#k updates $v_k(t_1)$ and $\mu_k(t_1)$ as in Eqs 26, 27
7	node#k packages and broadcasts TX_0 as in Eq. 40
8	if the latest TX_0 is collected then
9	determine the corresponding β_k as in Eq. 20
10	perform global model aggregation $\omega(t_1)$ as in Eq. 19
11	package the global aggregation packet TX_1 as in Eq. 41
12	generate $Block_N$ and broadcast as in Eqs 42–46
13	end if
14	if a new block is received and verified then
15	add the block to the global model block set
16	update locally stored global model ω
17	update local task start moment t_k
18	go to step#5
19	end if
20	if no information feedback is collected then
21	go to step#5
22	end if
23	end parfor
24	end for

server aggregates parameters and updates the model, distributing the global model to each node. These procedures are iterative and continue until convergence is achieved. The most commonly used model aggregation algorithm for federated learning is FedAvg. The total number of nodes is K, the kth node contributes data samples as S_k, then the number of samples of this node is $|S_k|$, and the total number of data samples is $S = \sum_{k=1}^{K}|S_k|$.

As in Eq. 6, the global objective function that requires optimization is $\min F(\omega)$, where $\omega \in R^d$:

$$\boldsymbol{F(\omega) = \frac{1}{S}\sum_{i=1}^{S} f_i(\omega)} \tag{6}$$

where $f_i(\omega) = l(x_i, y_i; \omega)$ represents the prediction loss for the sample (x_i, y_i) under the model parameter ω, then Eq. 6 can be rewritten as Eqs 7, 8:

$$\boldsymbol{F_k(v_k) = \frac{1}{|S_k|}\sum_{(x_i,y_i)\in S_k} f_i(v_k)} \tag{7}$$

$$\boldsymbol{F(\omega) = \sum_{k=1}^{K}\frac{|S_k|}{S}F_k(v_k)} \tag{8}$$

where $F_k(v_k)$ is the loss function of the kth node, while $F(\omega)$ is the overall federated loss function.

Each iteration process of FedAvg comprises a local update and a global update. During every iteration, the following process is undertaken:

Initially, the latest global model from the server is received as the local model for the current round during the local update of each node. Then, e iterations of the local model are carried out, updating the model parameter ω^k for the kth node in each iteration as in Eq. 9:

$$\boldsymbol{v_k^t = v_k^t - \eta\nabla F_k(v_k^t)} \tag{9}$$

where, η is the learning rate of the local update at each node.

Secondly, once the local update is executed, the model parameters are uploaded to the server. Subsequently, a server-side global aggregation update is implemented using the following aggregation formula as in Eq. 10:

$$\boldsymbol{\omega_{t+1} = \sum_{k=1}^{K}\frac{|S_k|}{S}v_k^t} \tag{10}$$

4.2 Asynchronous adaptive mechanism

The asynchronous adaptive federated learning algorithm proposed in this paper makes improvements based on the FedAvg algorithm. In the process of global model aggregation, an asynchronous mechanism and adaptive adjustment of node model weights based on the determination of time obsolescence are introduced. In the process of updating local node models, personalized learning and adaptive step length adjustment are introduced to ensure asynchronous real-time and regional personalization, and to improve the prediction accuracy while enhancing the generalization ability of the global model. The specific implementation process of the asynchronous adaptive federal learning algorithm is shown as follow.

First, the overall objective function is determined based on Eq. 10. In the asynchronous adaptive federal learning algorithm, the overall objective function is Eq. 11:

$$\boldsymbol{min\sum_{k=1}^{K}\frac{|S_k|}{S}F_k(v_k)} \tag{11}$$

Let $\lambda_k = \frac{|S_k|}{S}$ denote the weight of each node. Then add model personalization constraints as in Eq. 12:

$$\boldsymbol{s.t.\ \|v_k - \omega\|^2 \le T_k} \tag{12}$$

where $\{T_k, k = 1, 2, \ldots, K\}$ is the model personalization tolerance threshold, which is used to reflect the difference between the global model and the local model to achieve the personalized learning of the node's local model. The above optimization objective function with constraints can be rewritten by using the Lagrange equation as in Eq. 13:

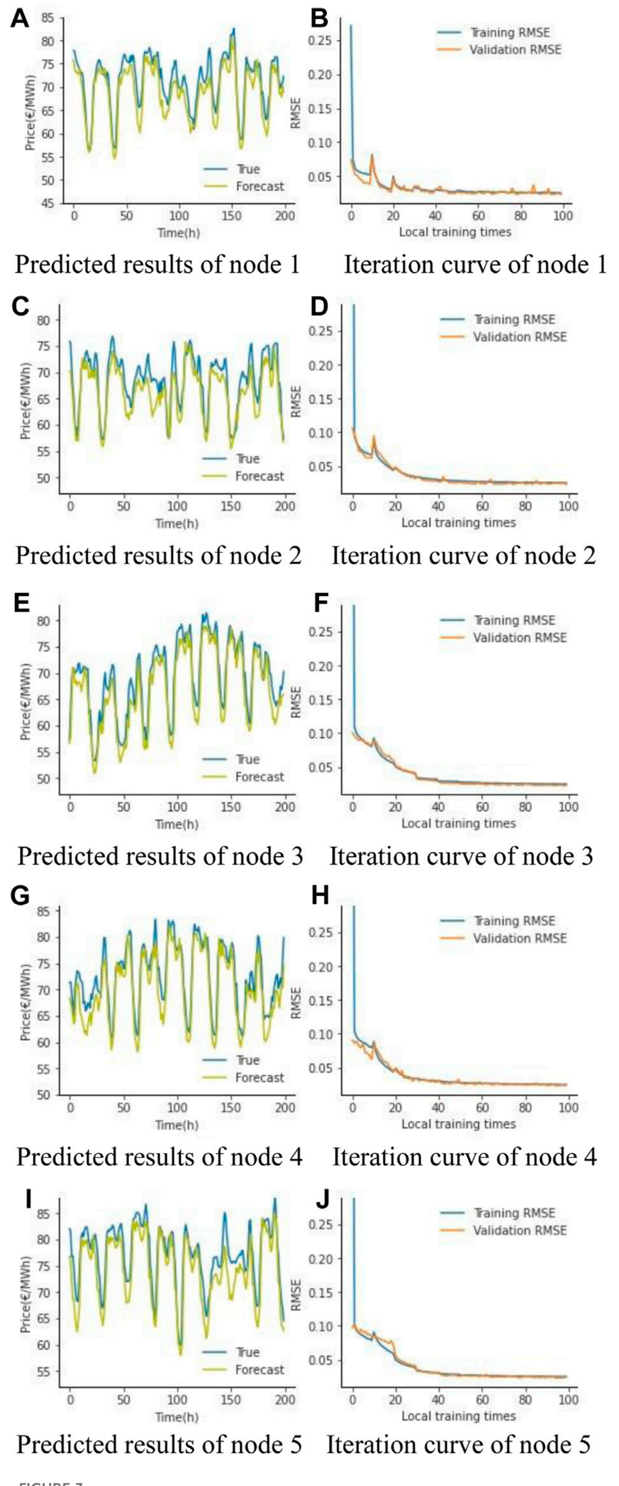

FIGURE 3
Overall predicted results and performance. **(A)** Predicted results of node 1. **(B)** Iteration curve of node 1. **(C)** Predicted results of node 2. **(D)** Iteration curve of node 2. **(E)** Predicted results of node 3. **(F)** Iteration curve of node 3. **(G)** Predicted results of node 4. **(H)** Iteration curve of node 4. **(I)** Predicted results of node 5. **(J)** Iteration curve of node 5.

$$L(v_k,\omega,\mu_k) = \sum_{k=1}^{K} \lambda_k F_k(v_k) + \frac{u_k}{2}\left(\|v_k-\omega\|^2 - T_k\right) \tag{13}$$

The partial derivative of this objective function can be obtained:

$$\nabla_{v_k} L = \lambda_k \nabla F_k(v_k) + u_k(v_k-\omega) \tag{14}$$

$$\nabla_{\omega} L = \sum_{k=1}^{K} u_k(v_k-\omega) \tag{15}$$

$$\nabla_{u_k} L = \frac{1}{2}\left(\|v_k-\omega\|^2 - T_k\right) \tag{16}$$

where $k = 1, 2, \ldots, K$.

Secondly, in the aggregation of global models, the asynchronous characteristics of multi-node collaboration should be considered. Asynchronous training allows each node to update model parameters independently with the server without worrying about the calculation pace of other nodes. In the case of asynchronous training, the model parameter update on the server side is carried out asynchronously, that is, there is no need to wait for the progress of other nodes that are being calculated. Considering the different time required for each node to complete the local task, this paper assumes that global aggregation begins whenever a single edge node completes uploading model parameters. Therefore, the variable $\varphi_k(t) \in \{0,1\}$ is introduced to indicate the completion of node updates as in Eq. 17:

$$\varphi_k(t) = \begin{cases} 1, kth\ node\ completes\ update\ at\ t \\ 0, others \end{cases} \tag{17}$$

The above variables satisfy the constraints as in Eq. 18:

$$\sum_{k=1}^{K} \varphi_k(t) = 1 \tag{18}$$

According to Eq. 15, the global model aggregation is given by:

$$\omega(t_1) = \omega(t_0) + \sum_{k=1}^{K} \varphi_k(t_1)\beta_k(t_k,t_1)(v_k(t_1) - \omega(t_0)) \tag{19}$$

where t_1 is the generation moment of the global model of the current generation, which is also the moment when the local model of the kth node completes this update, t_0 is the generation moment of the global model of the previous generation, t_k is the generation moment of the global model adopted by the local model of the kth node when it performs the current update, β_k is the weight adaptive adjustment function based on the determination of the temporal staleness, which is determined according to the before and after of the local update of the kth node. If this model is outdated, the current node model is given a smaller weight, the specific form can be defined as:

$$\beta_k(t_k,t_1) = \beta_m \cos\left(\frac{\pi(t_1-t_k)}{2(t_1+\varepsilon)}\right) \tag{20}$$

where β_m is the maximum learning weight, ε is a constant value close to zero. And when t_k is similar to t_1, β_k takes a value close to β_m.

$$\frac{\partial\beta_k}{\partial t_k} = \beta_m \sin\left(\frac{\pi(t_1-t_k)}{2(t_1+\varepsilon)}\right)\left(\frac{\pi}{2(t_1+\varepsilon)}\right) > 0 \tag{21}$$

$$\frac{\partial\beta_k}{\partial t_1} = -\beta_m \sin\left(\frac{\pi(t_1-t_k)}{2(t_1+\varepsilon)}\right)\left(\frac{\pi(t_k+\varepsilon)}{2(t_1+\varepsilon)^2}\right) < 0 \tag{22}$$

$$\frac{\partial^2\beta_k}{\partial {t_k}^2} = -\beta_m \cos\left(\frac{\pi(t_1-t_k)}{2(t_1+\varepsilon)}\right)\left(\frac{\pi}{2(t_1+\varepsilon)}\right)^2 < 0 \tag{23}$$

$$\frac{\partial^2\beta_k}{\partial {t_1}^2} = -\beta_m \cos\left(\frac{\pi(t_1-t_k)}{2(t_1+\varepsilon)}\right)\left(\frac{\pi(t_k+\varepsilon)}{2(t_1+\varepsilon)^2}\right)^2 + 2\beta_m \sin\left(\frac{\pi(t_1-t_k)}{2(t_1+\varepsilon)}\right) \times \left(\frac{\pi(t_k+\varepsilon)}{2(t_1+\varepsilon)^3}\right) \tag{24}$$

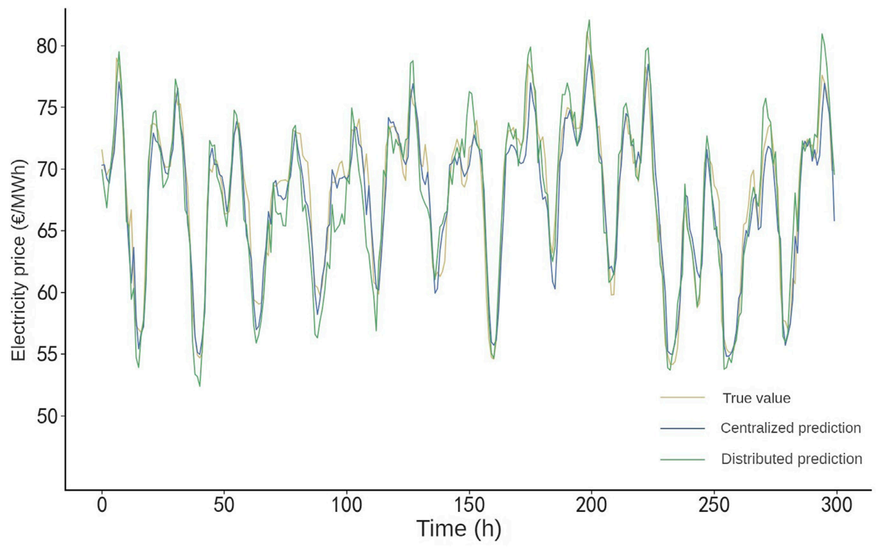

FIGURE 4
Comparison of centralized and distributed prediction.

TABLE 2 Comparison of centralized and distributed prediction.

Mode	Running time of each node (S)	RMSE	Data transmitted in a single transmission
Centralized	961.62	2.3	5 × 35,064 × 33
Distributed	817.79	2.313	53,601

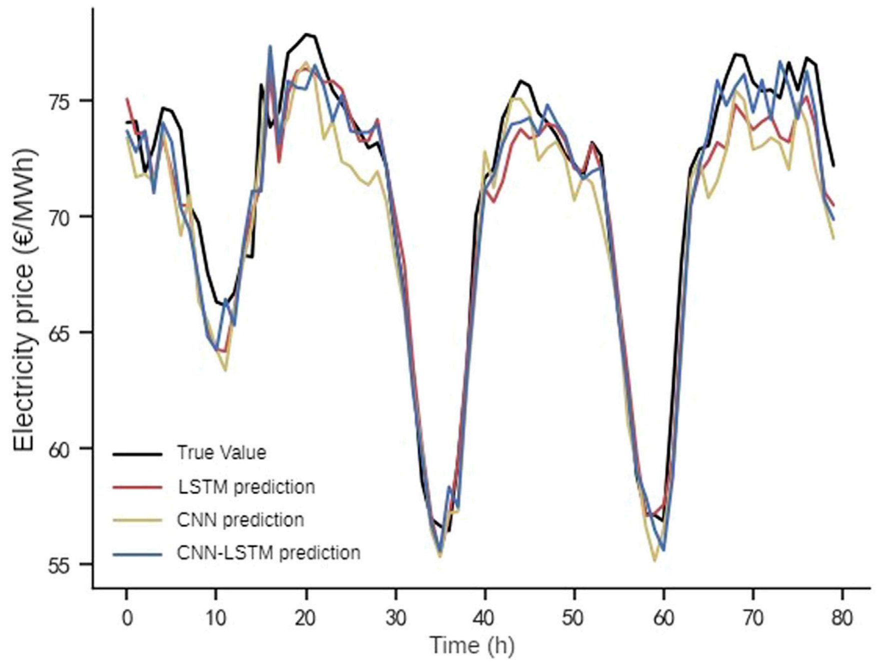

FIGURE 5
Comparison of prediction results under different models.

TABLE 3 Comparison of RMSE under different models.

Evaluation indicators	LSTM	CNN	CNN-LSTM
RMSE	2.377	2.353	2.313

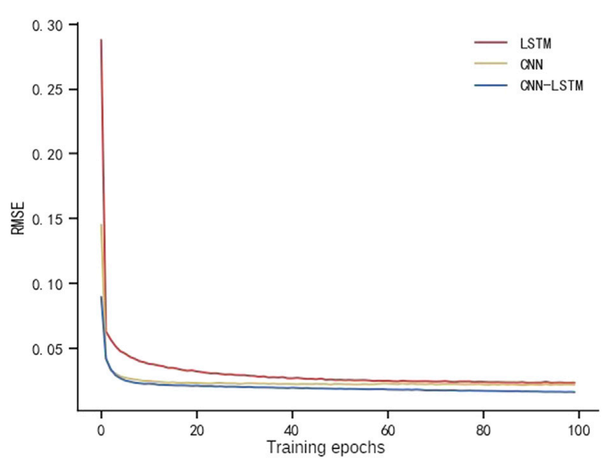

FIGURE 6
Comparison of iteration curves under different models.

The values of β_k vary with t_k and t_1 as shown in Figure 2. According to Eqs 21–24, β_k increases with the increase of t_k, and the magnitude of the increase decreases gradually, indicating that when the local model is updated with the newer global model, the weight of the updated node model should be increased accordingly. Also β_k decreases with the increase of t_1. When t_k is smaller, $\frac{\partial^2 \beta_k}{\partial {t_1}^2} > 0$, which means β_k is convex. When t_k is larger, $\frac{\partial^2 \beta_k}{\partial {t_1}^2} < 0$, which means β_k is concave. It indicates that the global model parameters obtained at the early stage of prediction have large weights only at the early stage, and their weights decrease rapidly in the later stage of prediction.

Convergence analysis is carried out on the aggregation of the global model. When $T \to \infty$, the algorithm is considered convergent if the statistic $R(T) = \sum_{t=1}^{T} [F(\omega(t)) - F(\omega^*)]$ meets the predetermined convergence criteria that $R(T)/T \to 0$, which means $\omega \to \arg\min \sum_{t=1}^{T} F(\omega(t)) \triangleq \omega^*$, ω converges to ω^* to minimize the objective function.

The convergence analysis is shown as in Eq. 25.

$$\begin{aligned} R(T) = \sum_{t=1}^{T}[F(\omega(t)) - F(\omega^*)] &\le \sum_{t=1}^{T} < \omega(t) - v, \omega(t) - \omega^* > \\ &\le \sum_{t=1}^{T} \frac{1}{2\beta_t}\left[\|\omega(t) - \omega^*\|^2 - \left\|\omega(t') - \omega^*\right\|^2\right] + \frac{\beta_t}{2}\|\omega(t) - v\|^2 \end{aligned} \tag{25a}$$

According to (22), it can be inferred that β_t decreases monotonically. Using the assumption that $\|\omega(t) - \omega^*\|^2 \le D^2$ and local node personalization constraints $\|\omega(t) - v\|^2 \le G^2$, the above equation can be simplified and transformed as follows:

$$\begin{aligned} R(T) &\le \frac{1}{2\beta_1}D^2 + D^2\sum_{t=2}^{T}\left(\frac{1}{2\beta_t} - \frac{1}{2\beta_{t-1}}\right) + \frac{G^2}{2}\beta_T \\ &= \frac{D^2}{2\beta_T} + \frac{G^2}{2}\beta_T \le \frac{D^2}{2\beta_{min}} + \frac{G^2\beta_{max}}{2} \end{aligned} \tag{25b}$$

Consequently, $R(T)$ has an upper bound and mean value of the statistic $R(T)/T \to 0$ when $T \to \infty$, ensuring the convergence of the global model aggregation.

Furthermore, for the update process of the edge model, according to Eqs 14, 16, the update process can be expressed as follow:

$$\begin{aligned} v_k(t_1) = v_k(t_k) \\ - \eta_1(\nabla F_k(v_k(t_k))) \times [\lambda_k \nabla F_k(v_k(t_k)) + u_k(v_k(t_k) - \omega(t_k))] \end{aligned} \tag{26}$$

$$u_k(t_1) = u_k(t_k) - \eta_2\left(\|v_k(t_k) - \omega(t_k)\|^2\right) \times \left(\|v_k(t_k) - \omega(t_k)\|^2 - T_k\right) \tag{27}$$

where η_1 and η_2 are the adaptive adjustment function of the step size, which can be expressed as

$$\eta(z) = max\{\eta_{min}, min\{\eta_{max}, p^{q^*z}\}\} \tag{28}$$

where η_{min} is the minimum value of the step, η_{max} is the maximum value of the step, $p > 1$ and $q > 0$, z is the independent variable of the adaptive function, and $\eta(z)$ is a non-decreasing function taking values in the interval $[\eta_{min}, \eta_{max}]$. As in Eq. 28, it is indicated that when the gradient of the loss function is small and the local model is close to the global model, the step size is adaptively reduced to prevent non-convergence in the learning of the local model. Compared with the fixed step size, the adaptive step size can adjust the amplitude of the parameter changing direction along the gradient according to the training stage, and improve the convergence rate while ensuring the convergence accuracy.

Convergence analysis of the update process of the edge model is performed. Since the parameters of the above process are solved according to a variant of gradient descent, a proof is required:

$$E\|v_k(t_1) - \omega^*\|^2 \le l(v_k(t_k) - \omega^*) \tag{29}$$

As in Eq. 29, the distance between the parameters of the current iteration and the optimal parameters ω^* is less than the distance between the parameters of the previous iteration and the optimal parameters ω^*, which means that the upper bound of the distance between the parameters of the current iteration and the optimal parameters is decreasing.

There are some assumptions on the functions $F_1, \ldots, F_K$ for the algorithm as the premise of convergence analysis:

1) As in Eq. 30, $F_1, \ldots, F_K$ are all *L-smooth*: for all v and ω,

$$F_k(v) \le F_k(\omega) + (v - \omega)^T \nabla F_k(\omega) + \frac{L}{2}\|v - \omega\|^2 \tag{30}$$

2) As in Eq. 31, $F_1, \ldots, F_K$ are all *μ-convex*: for all v and ω,

$$F_k(v) \ge F_k(\omega) + (v - \omega)^T \nabla F_k(\omega) + \frac{\mu}{2}\|v - \omega\|^2 \tag{31}$$

3) The variance of the stochastic gradient is bounded as in Eq. 32:

$$E\|\nabla F_k(v) - \nabla F_k(\omega)\| \le \sigma_k^2 \tag{32}$$

4) The expectation of the 2-Norm of the stochastic gradient is consistently bounded as in Eq. 33:

TABLE 4 Comparison of RMSE under different federated learning algorithms.

Federated learning algorithms	epochs = 5		epochs = 10		epochs = 20	
	rounds = 20		rounds = 10		rounds = 5	
	RMSE	Average RMSE	RMSE	Average RMSE	RMSE	Average RMSE
FedAvg	2.685	2.547	2.377	2.362	2.419	2.368
	2.405		2.495		2.228	
	2.479		2.484		2.288	
	2.506		2.269		2.259	
	2.660		2.185		2.646	
FedProx	2.550	2.543	2.267	2.358	2.218	2.365
	2.516		2.336		2.323	
	2.489		2.447		2.388	
	2.505		2.284		2.391	
	2.655		2.456		2.505	
FedPer	2.690	2.546	2.302	2.357	2.227	2.372
	2.495		2.406		2.203	
	2.411		2.333		2.237	
	2.553		2.309		2.399	
	2.581		2.435		2.794	
DAAFed	2.447	2.460	2.282	2.313	2.398	2.334
	2.317		2.294		2.255	
	2.386		2.316		2.429	
	2.414		2.308		2.267	
	2.736		2.365		2.321	

$$E\|\nabla F_k(v)\|_2^2 \leq G^2 \tag{33}$$

Meanwhile, we measure the system heterogeneity by Γ as in Eq. 34, which denotes the difference between the optimal value of the global objective function and the weighted local objective function. Γ converges to 0 when the data are distributed iid independently and identically.

$$\Gamma = F^{\star} - \sum\nolimits_k \lambda_k F_k^{*} \tag{34}$$

Let $v' = v_k(t_1)$ and $v = v_k(t_k)$, then

$$\begin{aligned} & E\left\|v' - \omega^{\star}\right\|^2 \\ &= E\left\|v - \omega^{\star} - \eta_1 \nabla F_k(v) \times [\lambda_k \nabla F_k(v) + u_k(v - \omega)]\right\|^2 \\ &= E(\|v - \omega^{\star}\|^2 + \eta_1{}^2[\lambda_k \nabla F_k(v) + u_k(v - \omega)]^2 - 2\eta_1 \times \\ & \quad < v - \omega^{\star}, [\lambda_k \nabla F_k(v) + u_k(v - \omega)]\nabla F_k(v) >) \end{aligned} \tag{35}$$

in which

$$\begin{aligned} & -2\eta_1 < v - \omega^{\star}, [\lambda_k \nabla F_k(v) + u_k(v - \omega)]\nabla F_k(v) > \\ &= -2\eta_1 < v - \omega + \omega - \omega^{\star}, [\lambda_k \nabla F_k(v) + u_k(v - \omega)]\nabla F_k(v) > \\ &\leq -2\eta_1\left[u_k T_k + \frac{\mu}{2}\|v - \omega^{\star}\|^2\right] + 6\Gamma L \eta_1{}^2 - \lambda_k T_k \end{aligned} \tag{36}$$

$$\begin{aligned} & [\lambda_k \nabla F_k(v) + u_k(v - \omega)]^2 \\ &= \lambda_k{}^2 \nabla F_k(v)^2 + u_k{}^2\|v - \omega\|^2 + 2\lambda_k \mu_k \nabla F_k(v)(v - \omega) \\ &\leq \lambda_k{}^2 G^2 + u_k{}^2 T_k + 2\lambda_k u_k \frac{\mu}{2} T_k \end{aligned} \tag{37}$$

In summary, the original equation can be converted as follow:

$$E\left\|v' - \omega^{\star}\right\|^2 \leq (1 - \eta_1 \mu)E\|v - \omega^{\star}\|^2 + \eta_1{}^2 B - (\lambda_k + 2\eta_1 u_k)T_k \tag{38}$$

$$B = 6\Gamma L + \lambda_k{}^2 G^2 + u_k{}^2 T_k + \lambda_k u_k \mu T_k \tag{39}$$

As in Eqs 35–39, this convergence analysis is formally similar to that of FedAvg. However the equation in this paper subtracts one more positive term, which show that the iterative process is convergent.

4.3 Decentralization mechanism

Considering the demand for distributed power time-series data prediction and the disadvantages of centralized processing, the approach of applying blockchain technology in this case is proposed in this paper in order to decentralize prediction,

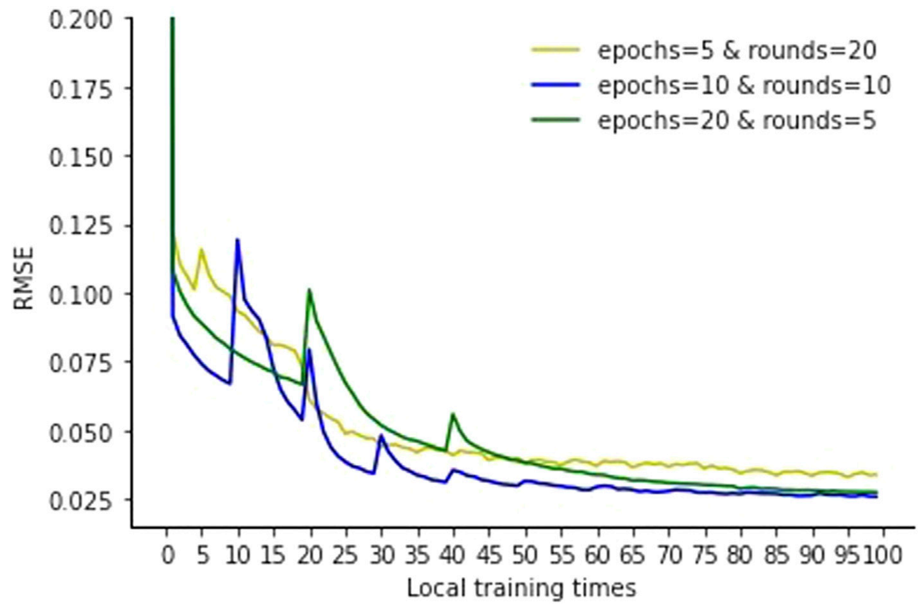

FIGURE 7
Comparison of iteration curves under different iteration parameter settings.

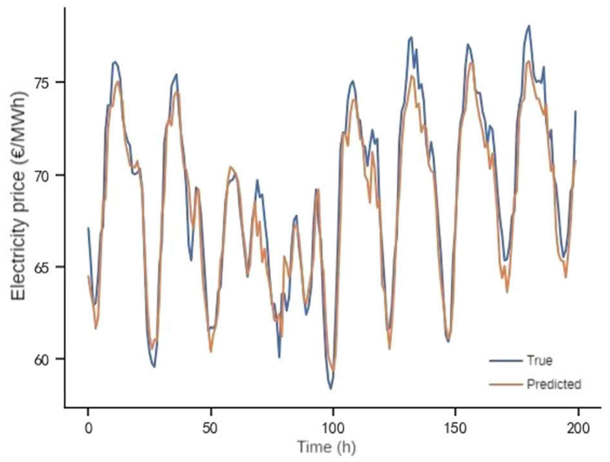

FIGURE 8
Predicted results of FedAvg.

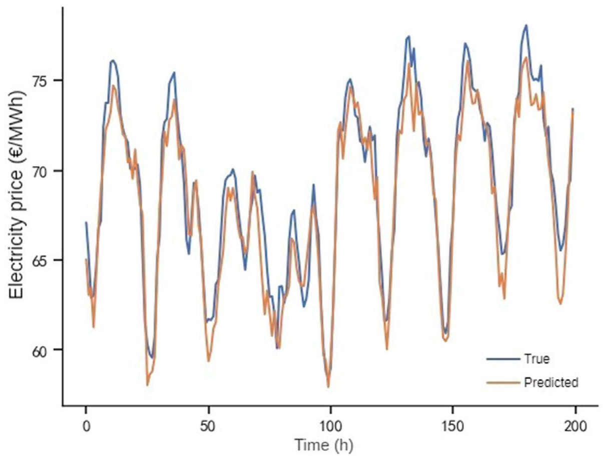

FIGURE 9
Predicted results of FedProx.

leveraging its global data ledger sharing and non-tamperability, which can further ensure the trust, security, transparency and traceability of data in the power industry.

In this paper, the federated learning global model parameter updating process described in the previous subsection is implemented by placing it on a blockchain. The blockchain guarantees the operation of a shared and trusted distributed ledger on a peer-to-peer network, which in the prediction scenario of this paper is the shared set of global model parameters blocks. The global model used for prediction is placed on the blockchain for updating so that it is not under the control of any single node, but each regional node has equal rights to validate and access the global model in the blockchain in such a way that it not only avoids a single point of failure, but also protects against data attacks in terms of security.

The workflow of the communication layer of the decentralized federated learning framework used in this paper broadly includes the processes of block generation, consensus verification and global maintenance. The decentralized implementation of the prediction framework includes the following:

1) Setting permissions for each regional participant, only participants with permissions for electricity data prediction can jointly train the federation model, and

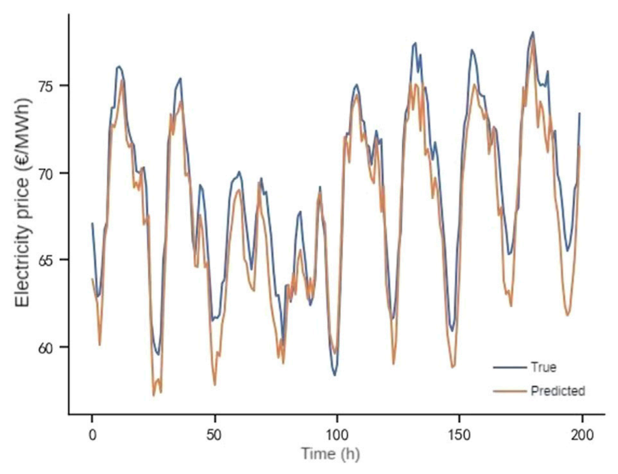

FIGURE 10
Predicted results of FedPer.

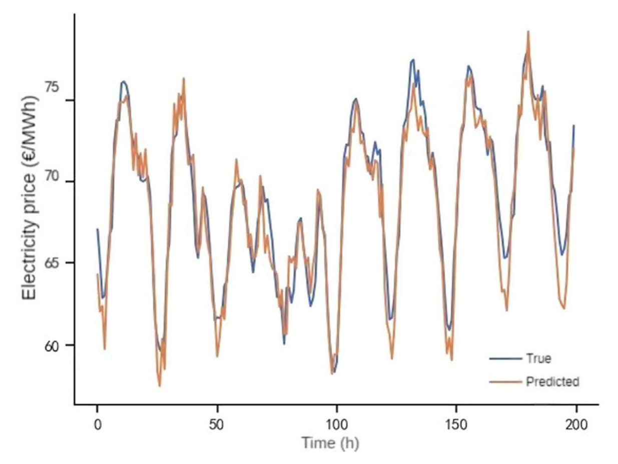

FIGURE 11
Predicted results of DAAFed.

new nodes can join and receive the current federation model only if they meet the permission conditions when accessing.

2) After each edge node completes a round of local training, it packages and broadcasts the privacy protected local model parameters and other data to the network, which includes the local task start time t_k, the local task completion time t_1, the local model parameters $v_k(t_1)$, and the task completion situation $\varphi_k(t_1)$ and other data to be recorded.

The local update is completed to form a data packet:

$$\boldsymbol{TX_0} = \{\boldsymbol{t_k}, \boldsymbol{t_1}, \boldsymbol{v_k}(\boldsymbol{t_1}), \ldots\} \tag{40}$$

3) Each edge node collects the latest packet TX_0 broadcast in the network, completes the update of the global model by block mining, calculates the global model aggregation $\omega(t_1)$, and packages the data before and after the aggregation.

The global aggregation is completed to form a data packet:

$$\boldsymbol{TX_1} = \{\boldsymbol{\varphi_k}(\boldsymbol{t_1}), \boldsymbol{\omega}(\boldsymbol{t_1}), \ldots\} \tag{41}$$

4) The node that completes block mining packages the above packets to form a block, and broadcasts it to the network of nodes waiting for verification. The block consists of a block body and a block header, where the block body contains the data before and after aggregation, and the block header contains the predecessor hash and the root hash, and the specific process of block generation includes:

Local updating and global aggregation of data packets together form the block body:

$$\boldsymbol{Body_N} = \{\boldsymbol{TX}\} = \{\boldsymbol{TX_0}, \boldsymbol{TX_1}, \ldots\} \tag{42}$$

Hash algorithm achieves the irreversible mapping from plaintext to ciphertext, this paper uses SHA256 to calculate the hash value. The successor hash is the hash value of the previous block and the root hash is generated based on the hash value of the packet $\{TX\}$:

$$\boldsymbol{PrevHash} = \boldsymbol{Hash}(\boldsymbol{Header_{N-1}}) \tag{43}$$

$$\boldsymbol{RootHash} = \boldsymbol{MerkleTree}(\{\boldsymbol{TX}\}) \tag{44}$$

The successor hash and the root hash together form the block header of the block:

$$\boldsymbol{Header_N} = \{\boldsymbol{PrevHash}, \boldsymbol{RootHash}\} \tag{45}$$

The block header and the block body together form the block:

$$\boldsymbol{Block_N} = \{\boldsymbol{Header_N}, \boldsymbol{Body_N}\} \tag{46}$$

5) After receiving blocks, all nodes in the network collaboratively verify updates to the global model. The fastest node to complete verification within each region sends its signature to the other nodes for verification. Once all nodes have reached a consensus, the block is added to the global model block set. The most recent global model, along with signatures and timestamps, form new blocks and attach to the blockchain. Finally, each node receives a copy of the most recent global model block to synchronise the model. Additionally, all nodes collectively store and maintain the block set of the global model.

At the same time, as shown in Eqs 47, 48, the node that successfully mines a new block can obtain the block reward $BlockReward(k)$, which in turn encourages the nodes to jointly maintain the update of the global model. Meanwhile, the node under the proof-of-work mechanism can obtain the reward $f(S_k)$ proportional to the size of the sample data, which in turn encourages the participants to contribute the data and the model. After joining the prediction federation, participants can obtain additional financial benefits by earning rewards while obtaining accurate prediction models.

$$\boldsymbol{Reward}(\boldsymbol{k}) = \boldsymbol{BlockReward}(\boldsymbol{k}) + \boldsymbol{f}(\boldsymbol{S_k}) \tag{47}$$

$$\boldsymbol{BlockReward}(\boldsymbol{k}) = \begin{cases} \boldsymbol{BaseReward}, \textbf{successful}\text{ mining} \\ \mathbf{0}, \quad \textbf{others} \end{cases} \tag{48}$$

4.4 The overall process

In summary, the pseudo-code of the decentralized asynchronous adaptive federated learning algorithmic process is shown in Table 1. Combining the advantages of federated learning and blockchain, the framework achieves privacy protection for local multi-source data while removing the dependence on a centralized central server. The approach improves training reliability and reduces communication costs, while ensuring the scalability of the edge node network and reducing the risk of single point of failure.

5 Experiments and results

5.1 Data and setup

The proposed method is implemented on a high performance server equipped with Intel Xeon Gold 6136 and NVIDIA TITAN XP. The Python libraries, including TensorFlow and Keras are used to build our model. The experimental data was selected as a dataset of historical data from 1st January 2015 to 31st December 2018 for five regions of the Spanish electricity market, which consists of time-series data on electricity price, generation, load, weather, and date information, sampled every 1 h, where the price of electricity is used as a label for the prediction. Missing values and outliers in the series are filled and replaced using linear interpolation, normalization is used before prediction training to prevent the effect of different magnitudes, and inverse normalization is used for prediction output to restore the relevance of the output data.

5.2 Analysis of the overall implementation effect

First of all, to show the overall implementation effect of the proposed framework in this paper, the data set is divided into training set, validation set and test set for prediction training, the RMSE of the training set and validation set in the iterative process is recorded, and finally the prediction of the test set is performed and the result is compared with the real value, and the partial tariff prediction results and RMSE iteration curves of the nodes are shown in Figure 3.

To verify the improvement of the prediction performance of the proposed framework in this paper, the distributed regional tariff prediction model and the centralized tariff prediction model are compared and analyzed, and the prediction results of the two are shown in Figure 4.

As shown in Figure 4, the predicted and actual values of the two prediction models are relatively close to each other, and the predicted value of the distributed model is slightly different from the actual value, because the samples used for training in the centralized model are the data of all nodes, while the samples used for training in each node in the distributed model are only the local data, but the distributed model strengthens the correlation between nodes by protecting the transmission of the model parameters and overcomes the phenomenon of data islanding.

The prediction performances of the distributed and centralized modes are shown in Table 2, except for the prediction accuracy, the distributed mode mostly outperforms the centralized mode. In terms of running time, since the computational tasks are assigned to each edge node in the distributed mode, and only local data is used for single-point training, the computational volume of a single computational unit is greatly reduced, and the single-point training time in the distributed mode is less than that in the centralized mode. In terms of data transmission, only the model parameters are transmitted in the distributed mode, which protects the data of each node and does not reveal the private data such as power generation information and user habits, and "53601" represents the sum of the elements of the weight matrix and bias vector of the prediction model. The centralized mode requires the transmission of all sample data, which has the risk of data leakage, where "$5 \times 35064 \times 33$" means that there are 35,064 sample data in each of the five nodes, and each sample contains 33 dimensions of data. Therefore, the distributed model can effectively reduce the amount of data transmitted in a single transmission and provide a certain level of security to protect the privacy of electricity.

5.3 Comparative analysis of edge node prediction models

Choosing a more appropriate model for distributed prediction at the edge nodes can effectively improve the overall prediction accuracy. To improve the implementation effect of the method proposed in this paper, LSTM model, CNN model and CNN-LSTM model are used at the edge nodes to predict the electricity price for the same data set. The model parameters are adjusted based on the method proposed in the related literature, and the prediction performance under the three models is compared and analyzed, as shown in Figure 5. The RMSE is calculated again after converting the model output into real size, and the results of the RMSE of each model evaluation index are shown in Table 3.

From Figures 5, 6 and Table 3, it can be seen that the overall prediction value of CNN-LSTM is closest to the real value, and the efficiency of the training iteration is also higher. The main reason is that CNN-LSTM combines the advantages of CNN and LSTM, capturing both local and global temporal features at different time scales through sliding window and loop structure, while the feature of parameter sharing of convolutional layer reduces the number of parameters of LSTM network and improves the computational efficiency. Therefore, choosing CNN-LSTM model for prediction at edge nodes can have good accuracy and efficiency.

5.4 Comparative analysis of federated learning algorithms

To verify that the DAAFed algorithm proposed in this paper has better prediction performance, FedAvg, FedProx (Li et al., 2020b), FedPer (Arivazhagan et al., 2019) and DAAFed are used to predict the electricity price on the same dataset, and the prediction performance of the above algorithms are compared and analysed. FedProx adds regularization terms to constrain the similarity between the global model and the local model to prevent overfitting (Li et al., 2020a). Each node model in FedPer has its own personalization layer that preserves the characteristics of the node data (Arivazhagan et al., 2019).

The RMSE and average RMSE for each node using the above federated learning algorithms with different iteration settings are shown in Table 4. FedProx adds a regularization term to limit the similarity between the global model and the local model and prevent overfitting; and each node has its own personalization layer in each node's model in FedPer, which is used to preserve the node data's own characteristics. Table 4 shows that the prediction performance of these two improved algorithms is similar to that of FedAvg. The DAAFed algorithm used in this paper has a smaller error compared to the other three algorithms, because the asynchronous mechanism used in the aggregation of the global model allows the nodes to update the model based on the latest global model, and at the same time, the step size is adaptively adjusted in the updating of the node's local model, so the model learns to the better parameter faster, and thus the overall performance is better. Therefore, the overall performance is better.

From Table 4, it can be seen that setting different local iteration epochs and global aggregation rounds has different prediction effects, and choosing the appropriate settings can fully exploit the advantages of the federated learning framework.

Figure 7 shows the iteration curves under different settings. It shows that when epochs = 10 and rounds = 10 there is better iteration efficiency than the other two settings. When epochs = 5 there is a slower decrease in the iteration curve because when the number of local iterations is small, the local model of each node does not have enough performance and cannot make full use of the local data. When epochs = 20 there is a slow decrease in the iteration curve before global aggregation, which means that a higher number of local iterations leads to a slower decrease of the iteration curve before global aggregation. This means that if the number of local iterations is high, it can lead to overfitting, which affects the ability of global model generalization, so the setting of epochs = 10 and rounds = 10 can give better federated learning results.

Based on this setting, the prediction results of a node under each of the above algorithms are shown in Figures 8–11, from which it can be seen that the predicted value of DAAFed is closer to the real value compared to several other federated learning algorithms.

6 Conclusion

In order to solve the problems of high computational cost and low data security under the traditional centralized electricity data forecasting method, this paper proposes a distributed regional electricity market electricity price forecasting method based on decentralized adaptive federated learning. The following conclusions are drawn from the analysis of the arithmetic example of the Spanish electricity market.

Compared to the conventional centralized prediction method, the distributed data prediction method reduces the amount of prediction calculation and data transmission, and approaches the centralized prediction accuracy in the implementation process, while protecting the local data in each region. Precise selection of prediction models on each node and appropriate configuration of the number of local iterations and global update rounds can be highly effective in enhancing the accuracy and efficiency of prediction. Compared to the Federated Learning algorithm, DAAFed enhances prediction accuracy by introducing an asynchronous adaptive mechanism. The validity of this proposed method is verified.

At present, privacy protection is becoming increasingly important in the power industry, which brings new challenges to distributed electricity price prediction. Therefore, how to further consider the protection effect of privacy protection mechanism and communication efficiency optimization will be the focus of the next research work.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

QL: Writing–original draft, Writing–review and editing, Conceptualization, Methodology, Validation. DL: Software, Supervision, Writing–review and editing. HC: Funding acquisition, Resources, Writing–review and editing. XiL: Formal Analysis, Project administration, Writing–review and editing. XuL: Data curation, Methodology, Visualization, Writing–review and editing. WC: Conceptualization, Investigation, Writing–review and editing.

Funding

The authors declare financial support was received for the research, authorship, and/or publication of this article. This research is supported by the State Grid Information and Telecommunication Group scientific and technological innovation projects "Research on Power Digital Space Technology System and Key Technologies" (Grant No. SGIT0000XMJS2310456).

Conflict of interest

Authors QL, DL, XiL, and WC were employed by State Grid Information & Telecommunication Group Co., Ltd.

The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

References

Ahmed, F. K., Hussain, J. S., Attique, M. K., Sharif, M., Tariq, U., and Kadry, S. (2022). A review on federated learning towards image processing. *Comput. Electr. Eng.* 99, 107818. doi:10.1016/j.compeleceng.2022.107818

Ali, H., Chandren, R., Kamrul, M., Paw, J. K. S., and Singh, M. J. (2023). Big data analytics using cloud computing based frameworks for power management systems: status, constraints, and future recommendations. *Sensors* 23 (6), 2952. doi:10.3390/s23062952

Antal, M., Mihailescu, V., Cioara, T., and Anghel, I. (2022). Blockchain-based distributed federated learning in smart grid. *Mathematics* 10 (23), 4499. doi:10.3390/math10234499

Arivazhagan, M., Aggarwal, V., and Singh, A. (2019). Federated learning with personalization layers. ArXiv:1912.00818. Available at: https://doi.org/10.48550/arXiv.1912.00818.

Badhiye, S., Chatur, P., and Raghuwanshi, M. (2022). Time series forecasting using range regression automata. International journal of uncertainty. *Fuzziness Knowledge-Based Syst.* 30 (06), 1035–1063. doi:10.1142/S0218488522500325

Chen, Y., Lu, W., Qin, X., Wang, J., and Xie, X. (2023). MetaFed: federated learning among federations with cyclic knowledge distillation for personalized healthcare. *IEEE Trans. neural Netw. Learn. Syst.* 2023, 1–12. doi:10.1109/tnnls.2023.3297103

Frizzo, S., Oriel, L., Cocco, V., and Coelho, L. d. S. (2023). Aggregating prophet and seasonal trend decomposition for time series forecasting of Italian electricity spot prices. *Energies* 16 (3), 1371. doi:10.3390/en16031371

He, W., and Zhao, L. (2022). Application of federated learning algorithm based on K-means in electric power data. *J. New Media* 4 (4), 191–203. doi:10.32604/jnm.2022.032994

Husnoo, M., Anwar, A., Reda, H., Hosseinzadeh, N., Islam, S., Mahmood, A., et al. (2023). FedDiSC: a computation-efficient federated learning framework for power systems disturbance and cyber attack discrimination. *Energy AI* 14, 100271. doi:10.1016/j.egyai.2023.100271

Li, T., Anit, K., Ameet, T., and Smith, V. (2020a). Federated learning: challenges, methods, and future directions. *IEEE Signal Process. Mag.* 37 (3), 50–60. doi:10.1109/msp.2020.2975749

Li, T., Sahu, A., and Zaheer, M. (2020b). Federated optimization in heterogeneous networks. *Proc. Mach. Learn. Syst.* 2020 (2), 429–450. doi:10.48550/arXiv.1812.06127

Li, Y., Wang, R., Li, Y., Zhang, M., and Long, C. (2023a). Wind power forecasting considering data privacy protection: a federated deep reinforcement learning approach. *Appl. Energy* 329, 120291. doi:10.1016/j.apenergy.2022.120291

Li, Y., Yan, Y., Liu, Z., Yin, C., Zhang, J., and Zhang, Z. (2023b). A federated learning method based on blockchain and cluster training. *Electronics* 12 (19), 4014. doi:10.3390/electronics12194014

Liu, Y., Wang, G., Guo, W., Zhang, Y., Dong, W., Wei, G., et al. (2021). Power data mining in smart grid environment. *J. Intelligent Fuzzy Syst.* 40 (2), 3169–3175. doi:10.3233/jifs-189355

Mcmahan, H., Moore, E., and Ramage, D. (2017). "Communication-efficient learning of deep networks from decentralized data," in Proceedings of the 20th International Conference on Artificial Intelligence and Statistics(AISTATS), 1273–1282.

Mohamed, A., Ibrahim, A., Hossam, H., and Karam, S. (2023). Towards efficient and trustworthy pandemic diagnosis in smart cities: a blockchain-based federated learning approach. *Mathematics* 11 (14), 3093. doi:10.3390/math11143093

Qi, T., Chen, L., Li, G., Li, Y., and Wang, C. (2023). FedAGCN: a traffic flow prediction framework based on federated learning and Asynchronous Graph Convolutional Network. *Appl. Soft Comput. J.* 138, 110175. doi:10.1016/j.asoc.2023.110175

Qu, Z., Li, Y., Liu, B., Gupta, D., and Tiwari, P. (2023). DTQFL: a digital twin-assisted quantum federated learning algorithm for intelligent diagnosis in 5G mobile network. *IEEE J. Biomed. health Inf.* 2023, 1–10. doi:10.1109/jbhi.2023.3303401

Wang, Z., Zhou, M., Zhao, Y., Zhang, F., Wang, J., Qian, B., et al. (2023). Electrical power edge-end interaction modeling with time series label noise learning. *Electronics* 12 (18), 3987. doi:10.3390/electronics12183987

Wu, X., Liang, Z., and Wang, J. (2020). FedMed: a federated learning framework for language modeling. *Sensors* 14, 4048. doi:10.3390/s20144048

Yu, H., Yang, S., and Zhu, S. (2019). "Parallel restarted SGD with faster convergence and less communication: demystifying why model averaging works for deep learning," in Proceedings of the AAAI Conference on Artificial Intelligence, 5693–5700.

Zeng, M., Wang, X., Pan, W., and Zhou, P. (2023). Heterogeneous training intensity for federated learning:a deep reinforcement learning approach. *IEEE Trans. Netw. Sci. Eng.* 10 (2), 990–1002. doi:10.1109/tnse.2022.3225444

Zhang, X., Zhong, C., Zhang, J., and Wang, T. (2023). Robust recurrent neural networks for time series forecasting. *Neurocomputing* 526, 143–157. doi:10.1016/j.neucom.2023.01.037