Article

Differentially Private Clustered Federated Load Prediction Based on the Louvain Algorithm

Tingzhe Pan [1,*], Jue Hou [2], Xin Jin [1], Chao Li [2], Xinlei Cai [2] and Xiaodong Zhou [1]

[1] CSG Science Research Institute Co., Ltd., Guangzhou 510640, China; jinxin1@csg.cn (X.J.); zhouxiaodong202409@163.com (X.Z.)

[2] Power Dispatching Control Center of Guangdong Power Grid Co., Ltd., Guangzhou 510060, China; houjue111111@163.com (J.H.); lichao20230930@163.com (C.L.); caixinlei20232024@163.com (X.C.)

* Correspondence: pantz@csg.cn

Abstract: Load forecasting plays a fundamental role in the new type of power system. To address the data heterogeneity and security issues encountered in load forecasting for smart grids, this paper proposes a load-forecasting framework suitable for residential energy users, which allows users to train personalized forecasting models without sharing load data. First, the similarity of user load patterns is calculated under privacy protection. Second, a complex network is constructed, and a federated user clustering method is developed based on the Louvain algorithm, which divides users into multiple clusters based on load pattern similarity. Finally, a personalized and adaptive differentially private federated learning Long Short-Term Memory (LSTM) model for load forecasting is developed. A case study analysis shows that the proposed method can effectively protect user privacy and improve model prediction accuracy when dealing with heterogeneous data. The framework can train load-forecasting models with a fast convergence rate and better prediction performance than current mainstream federated learning algorithms.

Keywords: federated learning; load forecasting; adaptive differential privacy; Louvain algorithm; clustered

check for updates

Academic Editor: Arslan Munir

Received: 8 October 2024
Revised: 25 December 2024
Accepted: 25 December 2024
Published: 8 January 2025

Citation: Pan, T.; Hou, J.; Jin, X.; Li, C.; Cai, X.; Zhou, X. Differentially Private Clustered Federated Load Prediction Based on the Louvain Algorithm. *Algorithms* **2025**, *18*, 32. https://doi.org/10.3390/a18010032

1. Introduction

The new power system is a complex intelligent network that primarily integrates new energy, along with components of information, physics, society, and big data [1,2]. The continuous introduction of technologies, such as demand response, electricity retailers, load aggregators, digital twins, and virtual power plants, has led to increasingly complex and variable load characteristics [3]. Load forecasting is foundational in this new power system, essential for the planning, operation, and scheduling of future smart grids.

Limited local data hinder effective load forecasting, making collaborative modeling essential. Recently, many load-forecasting schemes based on machine learning have been proposed. Reference [4] presented a short-term load-forecasting model based on linear regression. Reference [5] combined decision trees with convolutional neural networks to predict user power consumption, and the results showed that the prediction error was significantly reduced. In [6], a series of multiobjective predictive models were created utilizing a range of cutting-edge machine learning (ML) methodologies. For renewable energy, using wind power, a deep learning-driven self-conscious distributed cyber-physical system was proposed in [7]. However, retail providers often refuse to share data due to privacy concerns. Federated learning (FL) [8,9] offers a distributed machine learning framework that keeps data local, presenting a novel approach to load forecasting while ensuring privacy [10,11]. Reference [12] compared FedAvg, single-meter prediction, multi-meter centralized prediction, and federated stochastic gradient descent (FedSGD) methods and verified the superiority of FedAvg in load-prediction performance. Studies [13] have indicated that FedAvg can achieve smaller household load-prediction errors while protecting user privacy. In FL, users train load data locally and only upload the trained model parameters to the server. Despite maintaining local data, curious servers or attackers may still infer private information from a shared model or gradient parameters [14], highlighting the need for stronger privacy mechanisms. Differential privacy (DP) [15] is a widely used privacy protection technique. Ref. [16] proposed a differentially private stochastic gradient descent algorithm to train neural networks within a moderate privacy budget, while Ref. [17] introduced a differentially private federated learning algorithm and analyzed its performance. Implementing DP requires gradient clipping, but the fixed-threshold clipping method has significant limitations [18,19]. If the clipping threshold is too small, clipped gradients may become overly distorted, losing valuable information from local updates. Conversely, a threshold set too large can introduce excessive random noise, negatively affecting the algorithm's performance. Therefore, both excessively high and excessively low clipping thresholds can reduce model accuracy and practicality. In addition to data privacy issues, other network security issues, such as data poisoning and evasion attacks, are also receiving increasing attention (see [20,21] for detailed discussions).

Moreover, data distribution inconsistency among clients, known as non-IID (Non-Independently and Identically Distributed) data, can introduce significant bias during model training, resulting in client drift and slower convergence [22]. This inconsistency poses challenges for achieving optimal model performance with a single global model. To tackle this issue, Ref. [23] introduced the clustered federated learning (CFL) framework, which uses clustering algorithms to group clients with similar data distributions, allowing for the training of clustered personalized models to improve performance and specialization. Ref. [24] proposed an adaptive clustering federated learning algorithm that accelerates similarity assessment. Clients are assigned to different community clusters based on their data distribution similarities, enabling those within the same cluster to share a federated model. This method effectively reduces the negative impact of non-IID data on federated learning models. However, existing clustered federated learning algorithms have limitations. Classic clustering methods like K-Means [25], the iterative federated clustering algorithm (IFCA) [26], and CFL often require significant computational resources, which can hinder their practicality.

To address the challenges of high computational complexity, performance limitations, and privacy concerns in traditional clustered federated learning, this paper proposes a load-forecasting framework specifically designed for residential energy users. It introduces an innovative federated clustering method and explores a personalized adaptive differentially private clustered federated learning algorithm. The framework selects weather and time factors as key load-related variables, constructs a user dataset, and establishes a load-forecasting model. Case studies demonstrate that the proposed algorithm surpasses existing mainstream federated algorithms in predictive performance. The key contributions of this paper are as follows:

(1) A federated learning load-forecasting framework tailored for residential energy users with heterogeneous data is proposed.
(2) A method for calculating the similarity of load patterns while ensuring privacy is introduced. This involves constructing a complex network and developing a novel federated clustering method based on the Louvain algorithm, which offers lower computational costs and does not require pre-specifying the number of clusters (K).

(3) An adaptive gradient-clipping differentially private clustered federated averaging algorithm, called pADP-FedAvg, is proposed, along with a differential privacy analysis.

(4) Weather and time factors are identified as key load-related variables, a user dataset is constructed, and a federated learning load-forecasting LSTM model is established. The effectiveness and advantages of the proposed methods are validated through case studies.

2. Related Technology

2.1. Federated Learning Model

The standard workflow for federated learning is as follows: A central server initializes a global model and distributes it to each client. Clients then perform multiple rounds of training on the model using their local data, generating locally updated models that are subsequently transmitted back to the central server. The server aggregates the local models it receives to form an updated global model, which is then broadcast to all clients. By repeating this process, the global model is progressively refined. Classical federated learning utilizes a star topology, as depicted in Figure 1, comprising a central server and multiple clients. In this setup, θ_i^k denotes the model parameters of the i-th client during the k-th communication round, while θ^{k+1} represents the aggregated global model parameters and the client model parameters during the $(k+1)$-th communication round.

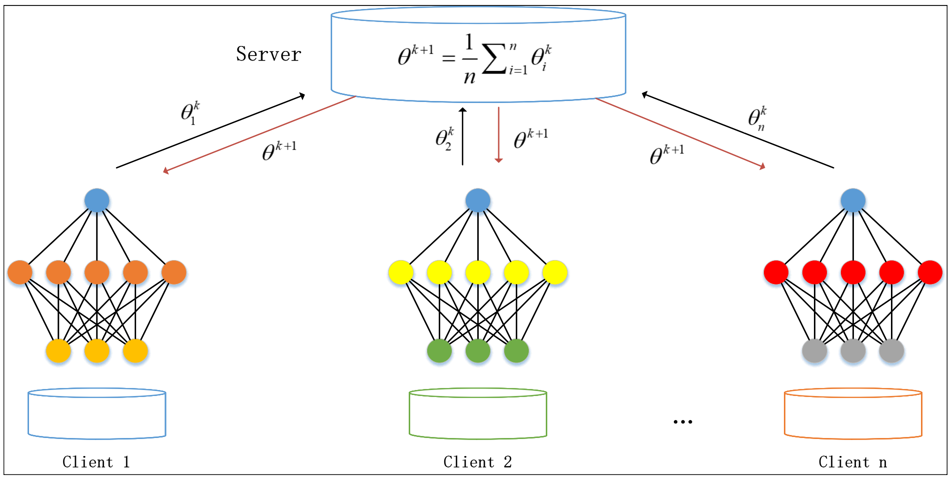


Figure 1. Star topology.

The local loss function of the local client i is

$$\min_{\theta} J_i(\theta) = \frac{1}{|D_i|}\sum_{j=1}^{|D_i|} F(\theta, \xi_j) \tag{1}$$

where θ represents the model parameters, $\xi_j \in D_i$ is a data sample, D_i is the client's dataset, and F is the loss function. During the model training process, it is necessary to minimize the global loss function $J(\theta)$, as defined in Equation (2):

$$\min_{\theta} J(\theta) = \sum_{i=1}^{|s|} \frac{|D_i|}{\sum_{i=1}^{|s|}|D_i|} J_i(\theta) \tag{2}$$

where S represents the set of clients selected to participate in the training. The client updates during the training process are provided by Equation (3):

$$\Delta\theta_i^{k+1} = \mathrm{SGD}(\theta^k, D_i, \eta_l) - \theta^k \tag{3}$$

where η_l denotes the learning rate for the local model and SGD refers to the stochastic gradient descent method. Equation (4) illustrates a single update applied by the server to the global model:

$$\theta^{k+1} = \theta^k + \sum_{i=1}^{|S|} \frac{|D_i|}{\sum_{i=1}^{|S|} |D_i|} \Delta\theta_i^k \tag{4}$$

Without loss of generality, we assume that the number of clients is n. The local dataset of client $i \in [n]$, denoted as $D_i = \{\xi_1^i, \dots, \xi_q^i\}$, consists of q data points.

2.2. Differential Privacy

Differential privacy protects data by adding noise that follows a certain distribution, thus providing privacy guarantees. It rigorously defines the strength of privacy protection. If two datasets D and D' differ by only one record, they are referred to as neighboring datasets. Differential privacy ensures that the results of the same query operation on any two neighboring datasets are nearly indistinguishable. Client-level differential privacy FL integrates privacy protection mechanisms into FL, ensuring that the learning model does not reveal whether a specific client participated in the training. This means that the entire dataset of the client is protected. The formal definition of differential privacy is shown below.

Definition 1 (Differential Privacy [15]**).** *A randomized mechanism M, where $Dom(M)$ denotes the domain and $Range(M)$ denotes the range. If for any two neighboring datasets $D, D' \subseteq Dom(M)$ and any subset $O \subseteq Range(M)$, the following holds:*

$$\Pr[M(D) \in O] \le e^{\epsilon} \Pr[M(D') \in O] + \delta, \tag{5}$$

then M is said to satisfy (ϵ, δ)-DP, where the parameter ϵ represents the privacy budget.

Definition 2 (l_2 Sensitivity [15]**).** *For a query function $f\colon D \to R^d$ on a given dataset, where D and D' are two neighboring datasets, the l_2 sensitivity of f is defined as*

$$\Delta(f) = \max_{D,D'} \|f(D) - f(D')\|_2. \tag{6}$$

Zero-Concentrated Differential Privacy (zCDP) is a new relaxed form of differential privacy. Compared to the standard (ϵ, δ)-DP, it offers a clearer and more precise analysis of the privacy loss over multiple iterative computations. The definition of zCDP is shown below.

Definition 3 (zCDP [27]**).** *For any two neighboring datasets D and D', and for any $\alpha > 1$, a randomized mechanism M satisfies ρ-zCDP if and only if*

$$D_\alpha\left(M(D) \middle\| M(D')\right) = \frac{1}{\alpha - 1} \log\left(E\left[e^{(\alpha-1)L^{(O)}}\right]\right) \le \rho \tag{7}$$

where $D_\alpha\left(M(D) \middle\| M(D')\right)$ denotes the α-Renyi divergence between $M(D)$ and $M(D')$. $L^{(O)}$ represents the privacy loss incurred by M between neighboring datasets D and D' when the outcome is O, that is,

$$L^{(O)}_{\left(M(D) \middle\| M(D')\right)} = \log \frac{\Pr(M(D) = O)}{\Pr(M(D') = O)} \tag{8}$$

The following propositions hold for zCDP:

Lemma 1 ([27]). *The Gaussian mechanism, which returns $f(D) + N(0, \Delta^2(f)\sigma^2 I)$, satisfies $(1/2\sigma^2)$-zCDP.*

Lemma 2 ([27]). *Suppose there are k mechanisms $M_1, \ldots, M_k$, and each M_i (for $i = 1, \ldots, k$) satisfies ρ_i-zCDP. Then, the composition of these mechanisms satisfies $\left(\sum_{i=1}^{k} \rho_i\right)$-zCDP.*

Lemma 3 ([28]). *Let M be composed of adaptive randomized mechanisms $M_1, \ldots, M_E$, and each M_i (for $i = 1, \ldots, E$) satisfies ρ_i-zCDP. When the dataset D is randomly partitioned into $D_1, \ldots, D_E$, then $M(D) = (M_1(D_1), M_2(D_2), \ldots, M_E(D_E))$ satisfies $\max_i \rho_i$-zCDP.*

Lemma 4 ([27]). *If M satisfies ρ-zCDP, then for any $\delta > 0$, M satisfies $\left(\rho + 2\sqrt{\rho \log(1/\delta)}, \delta\right)-DP$.*

3. Problem Description

This paper addresses the collaborative challenges and privacy protection issues faced by multiple data collectors during the process of power load forecasting. In the power grid, each user independently holds their own power load data, and due to privacy concerns, it is difficult to directly share their original data. Federated learning is a reasonable choice to solve this problem. In the k-th round of federated learning, each client i performs a local model update:

$$\Delta\theta_i^{k+1} = \mathrm{SGD}(\theta^k, D_i, \eta_l) - \theta^k, i = 1, 2, \ldots, n \tag{9}$$

The global model is updated as

$$\Delta\theta^{k+1} = \sum_{i=1}^{n} \frac{|D_i|}{|D|} \Delta\theta_i^{k+1} = \kappa_1 \Delta\theta_1^{k+1} + \kappa_2 \Delta\theta_2^{k+1} + \ldots + \kappa_n \Delta\theta_n^{k+1} \tag{10}$$

where $D = D_1 \cup D_2 \cup \ldots \cup D_n$, and $\kappa_i = \frac{|D_i|}{|D|}$. When federated training reaches the stable optimal solution θ^*, the global model update approaches zero, i.e., $\Delta\theta = 0$. At this point, two situations may occur:

(1) $\Delta\theta_1 = \Delta\theta_2 = \ldots = \Delta\theta_n = 0$, indicating that all clients have consistent data distributions and the optimal solution is achieved simultaneously.
(2) The local model updates $\Delta\theta_i$ are not all zero, which is caused by inconsistent data distributions during the training process. This scenario more accurately reflects practical applications of federated learning. In this case, the local model updates may not guide the global model toward optimization, resulting in generally lower performance and accuracy for the global model.

Data quality is a critical foundation for load-forecasting modeling in power systems. Existing methods often assume that user load data are independently and identically distributed. In reality, while the power consumption data of different users are independent, the assumption of identical distribution does not hold true. Moreover, although the FL model keeps the training data locally, it requires uploading model parameter updates. Attackers can infer users' sensitive information from these uploaded model parameters, making it challenging to effectively ensure data privacy.

4. Research Motivation

Due to the non-uniformity of user load data, a single global model is insufficient to meet personalized user requirements. This paper considers using clustering algorithms to aggregate clients with similar data distributions, group electricity users into different clusters, and learn personalized models for each cluster. Providing suitable differentiated models for each user can effectively prevent client drift, thereby enhancing prediction accuracy and model specialization.

Federated clustering can be divided into one-time clustering and iterative clustering based on timing. Ref. [25] adopted a one-time clustering method, which requires specifying the number of clusters K in advance. IFCA uses iterative clustering without the need to predefine K; however, it employs a centralized clustering algorithm, resulting in a significant increase in computational overhead at the central server. CFL also adopts iterative clustering, requiring substantial computational resources to confirm the cluster identities of clients.

To address the issues present in the aforementioned federated clustering methods, this paper constructs a complex network and considers the modularity function. Based on the classical Louvain algorithm for community detection in complex networks, we propose a new federated clustering method that can effectively protect user data privacy, does not require pre-specifying the number of clusters K, and achieves a well-performing differentially private federated clustering model with lower computational cost.

5. System Architecture and Design Scheme

5.1. System Architecture

To address the inconsistency of user load data and privacy protection issues, this paper proposes a differential privacy clustering federated learning method based on the Louvain algorithm for heterogeneous data. The Louvain algorithm is a community detection method for complex networks. The system architecture, illustrated in Figure 2, is divided into three modules.

Module One involves all clients conducting differential privacy federated learning until convergence. During this phase, the server calculates the similarity of the local update vectors from the clients for the current round. Module Two constructs a complex network and employs the Louvain algorithm for community clustering. Module Three focuses on personalized adaptive differential privacy clustering for federated learning load forecasting.

5.2. Similarity Calculation

Since user load data are not exchanged, federated learning cannot directly cluster the data at the nodes; instead, it clusters the model parameters of the clients. Similar clients are identified by analyzing their model parameters. To evaluate the similarity of the clients' model parameters while ensuring privacy, all clients first perform differential privacy federated learning until convergence. Subsequently, the server calculates the similarity of the local update vectors from the clients for the current round. The specific steps are as follows:

Step 1 Clients use the differential privacy federated averaging (DP-FedAvg) algorithm [29,30] to train until convergence.

Step 2 Equation (11) is used to calculate the cosine similarity between any two clients' local updates. The cosine similarity evaluates the similarity of two vectors by calculating the cosine of the angle between them. The closer the value is to 1, the more similar the two vectors are:

$$e_{ij} = e_{ji} = \frac{\langle \Delta\theta_i, \Delta\theta_j \rangle}{\|\Delta\theta_i\| \|\Delta\theta_j\|} \tag{11}$$

5.3. Complex Network Construction

As an abstract model for understanding complex systems, complex networks are a powerful tool for solving clustering problems. Each client is regarded as a network node, and the network edge set and edge weights between nodes are constructed based on the learned similarity between client parameters, establishing an undirected weighted complex network $G = (V, E, A)$, where V is the set of network nodes, E is the set of edges, and A is the adjacency matrix. When $e_{ij} < 0$, the similarity of clients i and j is very small, $(i, j) \notin E$; when $e_{ij} > 0$, $(i, j) \in E$. The adjacency matrix is specifically defined as follows:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \tag{12}$$

where, when $A \neq B$, $a_{ij} = \begin{cases} e_{ij}, & e_{ij} \geq 0 \\ 0, & e_{ij} < 0 \end{cases}$, and when $i = j$, $a_{ij} = 0$, that is, self-loops are not considered. A is a symmetric matrix.

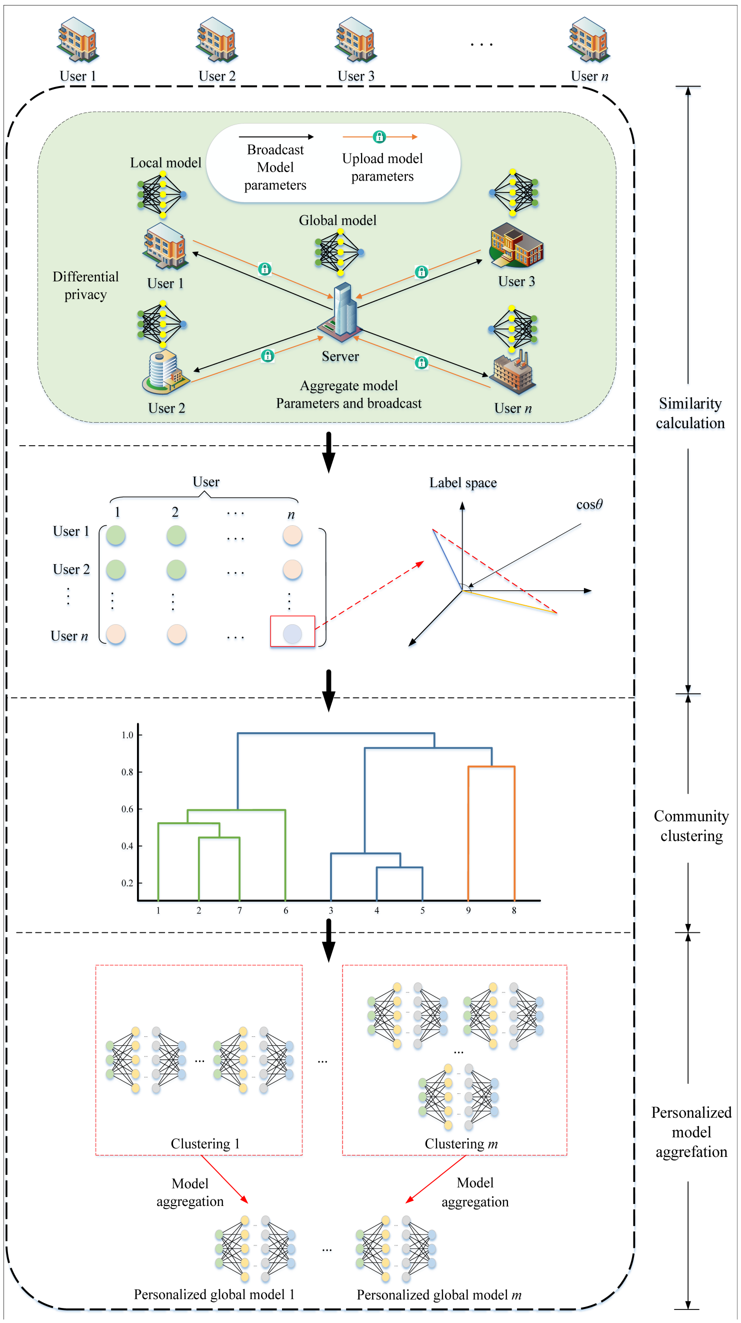


Figure 2. The framework for clustered federated learning.

5.4. Louvain Algorithm

The Louvain algorithm is a community detection method for complex networks based on modularity maximization [31]. It has an approximate linear time complexity and is regarded as one of the best-performing clustering algorithms. The modularity function Q is defined by the following equation:

$$Q = \frac{1}{2\tau} \sum_{i,j} \left[a_{ij} - \frac{k_i k_j}{2\tau} \right] \delta(c_i, c_j) = \frac{1}{2\tau} \left[\sum_{i,j} a_{ij} - \frac{\sum_i k_i \sum_j k_j}{2\tau} \right] \delta(c_i, c_j) \tag{13}$$

where $k_i = \sum_j a_{ij}$ is the sum of the weights of all edges connected to node i. $\tau = \frac{1}{2} \sum_{i,j} a_{ij}$, c_i is the cluster containing node i. When $c_i = c_j$ and $c_i = c_j$, $\delta(c_i, c_j) = 1$; otherwise, $\delta(c_i, c_j) = 0$. The value of Q reflects the quality of community clustering. The closer it is to 1, the more distinct the clustering structure.

The basic idea of the algorithm is to first treat each node in the network as a cluster. Then, node i is moved into the cluster c of its neighboring node j, and the modularity increment ΔQ, expressed by Equation (14), is calculated to determine the movement method that has the greatest positive impact on modularity:

$$\Delta Q = \left[\frac{\sum_{in} + k_{i,in}}{2\tau} - \left(\frac{\sum_{tot} + k_i}{2\tau} \right)^2 \right] - \left[\frac{\sum_{in}}{2\tau} - \left(\frac{\sum_{tot}}{2\tau} \right)^2 - \left(\frac{k_i}{2\tau} \right)^2 \right] \tag{14}$$

where $\sum_{in}$ is the sum of the weights of the internal edges within cluster c, $k_{i,in}$ represents the sum of the weights of the edges connecting node i to the nodes within cluster c, and $\sum_{tot}$ is the sum of the weights of all edges connecting to the nodes within cluster c. The similarity measure is based on the distribution of load data that groups users, which manifests as community clustering at the level of the complex network.

5.5. Adaptive Differential Privacy Personalized Clustering Federated Learning Algorithm

In the implementation of DP, gradient clipping is essential. The fixed-threshold clipping method has notable drawbacks: setting the clipping threshold too high or too low can lead to decreased model accuracy and practicality. This paper proposes an adaptive gradient-clipping method, where the clipping threshold is defined by Equation (15):

$$C_i^{t+1} = \left\| g(y_i^{t,L-1}) \right\|_2 \times \min(1,\ C_i^t / \left\| g(y_i^{t,L-1}) \right\|_2) + N(0, (C_i^t \sigma)^2) \tag{15}$$

where C_i^t is the clipping threshold for client i during the t-th round of training, $g(y_i^{t,L-1})$ is the gradient after the L-th local iteration in the t-th round of training for client i, and σ is the noise parameter. The idea behind this method is to use the clipped gradient from the t-th round to estimate the clipped gradient for the $(t+1)$-th round, with Gaussian noise added to ensure privacy protection.

Building on adaptive gradient clipping, this paper proposes a personalized federated averaging algorithm, pADP-FedAvg. First, the server initializes a global model and distributes it to each client. Next, clients sample mini-batch local data to compute gradients, apply adaptive gradient clipping, and incorporate DP to update the local model. After L local iterations, the local updates are sent back to the central server. The server then aggregates the received local models to generate a new global model, which is broadcast to all clients. By repeating these steps, the global model is gradually optimized. The specific process is summarized in Algorithm 1 below.

Algorithm 1 pADP-FedAvg algorithm.

Input: Number of communication rounds T; Local iteration period L; Initial value of the personalized model y^0; Local update step size η; Noise parameter σ; Initial value of the adaptive clipping threshold $\{C_i^0\}_{i=1}^n$; Number of clients selected in each round r

Output: Personalized model y^T

```
Server executes:
1:  initialize $y^0$
2:  for each round $t = 0, \dots, T-1$ do
3:    Sample a set of $r$ clients at random without replacement, denoted by $W^t$
4:    Broadcast $y^t$ to all clients in $W^t$
5:      for each client $i \in W^t$ in parallel do
6:          $y_i^t \leftarrow$ClientUpdate$(i, y^t)$
7:      end for
8:    $y^{t+1} = (1/r)\sum_{i \in W^t} y_i^t$
9:  end for
10: Return $y^T$
ClientUpdate$(i, y^t)$:
11: $y_i^{t,0} \leftarrow y^t$
12: for $l = 0, \dots, L-1$ do
13:     Sample a mini-batch $X_i \subseteq D_i$ of size $\lambda$ and compute gradient
          $g(y_i^{t,l}) \leftarrow (1/\lambda) \sum_{\xi_i \in X_i} \nabla F(y_i^{t,l}, \xi_i)$
14:     Adaptive gradient clipping:
15:       $\tilde{g}(y_i^{t,l}) \leftarrow g(y_i^{t,l}) \times \min(1, C_i^t / \left\| g(y_i^{t,l}) \right\|_2)$
16:     Add DP noise to gradient:
17:       $\bar{g}(y_i^{t,l}) \leftarrow \tilde{g}(y_i^{t,l}) + N(0, (C_i^t\sigma)^2)$
18:       $y_i^{t,l+1} = y_i^{t,l} - \eta_l \bar{g}(y_i^{t,l})$
19: end for
20: Computes clipping threshold $C_i^{t+1}$ by Equation (15)
21: upload $y_i^{t,L}$ to server
```

5.6. Privacy Analysis

The introduction of differential privacy aims to address differential attacks and prevent honest but curious servers or participants from inferring and stealing sensitive private information of clients based on shared data.

Theorem 1. *In Algorithm 1, for the local dataset D_i of client i participating in training, a small batch sample of size λ is randomly selected without replacement. Assuming that client i participates in training and uploads updates T_i times during T rounds of learning, then for client i, Algorithm 1 satisfies the following condition (ε_i, δ)-DP, where*

$$\varepsilon_i = \frac{2T_iL}{qr\lambda\sigma^2} + 2\sqrt{\frac{2T_iL}{qr\lambda\sigma^2}\log\frac{1}{\delta}} \tag{16}$$

Proof. For client i, assuming that two adjacent datasets X_i and $X_i{}'$ differ only in the j-th data point, i.e., $\xi_j \neq \xi'{}_j$, then

$$\left\| \tilde{g}(y_i^{t,l})(X_i) - \tilde{g}(y_i^{t,l})(X_i') \right\|_2 = \frac{1}{\lambda}\left\| \tilde{g}(y_i^{t,l})(\xi_j) - \tilde{g}(y_i^{t,l})(\xi_j') \right\| \leq \frac{2C_i^t}{\lambda} \tag{17}$$

Therefore, $\Delta_2(\tilde{g}(y_i^{t,l})) \leq \frac{2C_i^t}{\lambda}$. Noting that, in the absence of added noise,

$$\begin{aligned} y_i^{t,L} &= y_i^{t,L-1} - \eta_l \tilde{g}(y_i^{t,L-1}) \\ &= y_i^{t,L-2} - \eta_l \tilde{g}(y_i^{t,L-2}) - \eta_l \tilde{g}(y_i^{t,L-1}) \\ &= \cdots \\ &= y_i^{t,0} - \eta_l \tilde{g}(y_i^{t,0}) - \eta_l \tilde{g}(y_i^{t,1}) - \cdots - \eta_l \tilde{g}(y_i^{t,L-1}) \end{aligned} \tag{18}$$

By the definition of sensitivity and Equations (17) and (18), we have

$$\begin{aligned} \Delta(y_i^{t,K}) = \eta_l \Big\| & \tilde{g}(y_i^{t,0})(X_i^{t,0}) - \tilde{g}(y_i^{t,0})(X_i^{t,0\prime}) + \tilde{g}(y_i^1)(X_i^{t,1}) - \tilde{g}(y_i^1)(X_i^{t,1\prime}) + \cdots \\ & + \tilde{g}(y_i^{t,L-1})(X_i^{t,L-1}) - \tilde{g}(y_i^{t,L-1})(X_i^{t,L-1\prime}) \Big\| \\ & \leq \frac{2\eta_l L C_i^t}{\lambda} \end{aligned} \tag{19}$$

From Equation (17) and Lemma 1, it is known that in Algorithm 1, each participating client satisfies $\frac{2}{\lambda^2\sigma^2}$-zCDP during a single local iteration. During one round of local iteration, the total number of accesses to the local dataset by the participating client is $\frac{L\lambda}{q}$. According to Lemma 2 and Lemma 3, the participating client satisfies $\frac{2L}{q\lambda\sigma^2}$-zCDP during one round of local iteration. Combining Equation (19) and Lemma 1, Algorithm 1 satisfies $\frac{2L}{qr\lambda\sigma^2}$-zCDP in the *t*-th round of training. If client *i* participates in training and uploads updates T_i times during T rounds of learning, then client *i* achieves $\frac{2T_iL}{qr\lambda\sigma^2}$-zCDP throughout the learning process. By Lemma 4, the theorem is proved. □

6. Electricity Load-Forecasting Model

6.1. Load Influencing Factors

Electric load is influenced by various factors, including weather, location, time, electricity prices, user consumption habits, and unexpected events. The load curve can be decomposed into regular components that change periodically, uncertain components that change non-periodically, and noise components from unexplainable physical factors [32]. Key weather factors, such as temperature, humidity, and atmospheric pressure, significantly impact short-term load variations.

6.2. Construction of User Dataset

In this paper, weather and time factors are selected as correlated influences on load. The input data feature types include weather, load, and time. Weather factors, such as temperature, humidity, and atmospheric pressure, are used to capture the impact of weather on load. Time factors, including year, month, day, and hour, reflect the periodicity of the load. The feature selection and transformation process for the dataset is detailed in Table 1.

6.3. Federated Learning Model Based on LSTM

The LSTM (Long Short-Term Memory) network excels at handling time-series problems, and the electricity load of each user is inherently a type of time series. Figure 3 shows the internal unit structure of LSTM, which consists of input gates, forget gates, and output gates.

Table 1. Feature selection and feature transformation of datasets.

Feature Type	Feature Name	Data Type	Unit/Range	Transformation Method	Transformed Feature
Weather	Temperature	Continuous	°F	Normalization	temperature
	Humidity	Continuous	%	Normalization	humidity
	Pressure	Continuous	kPa	Normalization	pressure
Load	Load	Continuous	kW	Normalization	load
Time	Year	Discrete	2016–2018	One-hot encoding	oh-2016, oh-2017, oh-2018
	Month	Discrete	1–12	sin/cos Cyclic encoding	month-sin, month-cos
	Day	Discrete	1–30	sin/cos Cyclic encoding	day-sin, day-cos
	Hour	Discrete	0–23	sin/cos Cyclic encoding	hour-sin, hour-cos

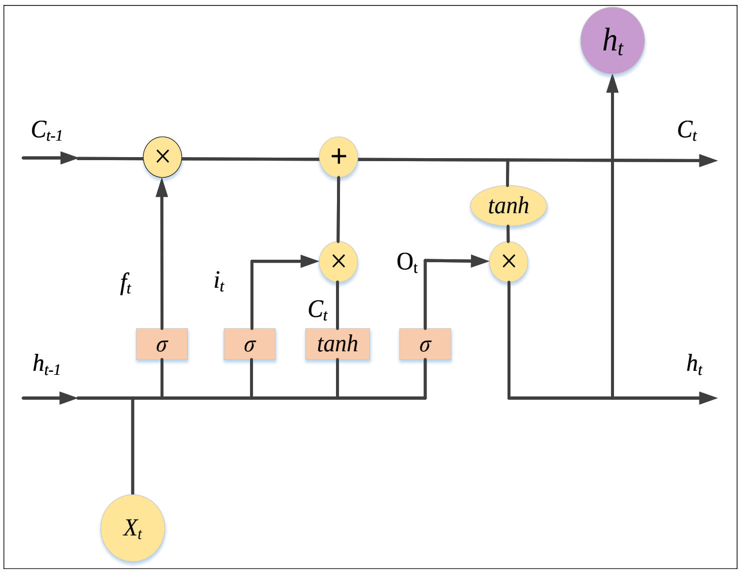


Figure 3. Internal structure of LSTM.

In Figure 3, X_t is the input, h_{t-1} and h_t are the outputs, and C_{t-1} and C_t are the storage cell states. The calculation formulas for these variables are as follows:

$$f_t = \sigma\left(W_f X_t + U_f h_{t-1} + b_f\right) \tag{20}$$

$$i_t = \sigma(W_i X_t + U_i h_{t-1} + b_i) \tag{21}$$

$$o_t = \sigma(W_o X_t + U_o h_{t-1} + b_o) \tag{22}$$

$$\tilde{C}_t = \tanh(W_c X_t + U_c h_{t-1} + b_c) \tag{23}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{24}$$

$$h_t = o_t \odot \tanh(C_t) \tag{25}$$

where σ is the sigmoid function and f_t, i_t, and o_t are the outputs of the forget gate, input gate, and output gate, respectively, W_c, U_c, W_o, U_o, W_i, U_i, W_f, and U_f are the corresponding weights, and b_c, b_o, b_i, and b_f are the bias terms. This paper adopts the framework of federated learning, using LSTM as the load-forecasting model, leveraging the three gates mentioned above to utilize long-term user historical data for load forecasting. Each participating user acts as an independent client, and the network topology is illustrated in Figure 1.

6.4. Clustered Federated Learning Load-Forecasting Process Based on Louvain Algorithm

The entire process of personalized adaptive differential privacy clustering federated learning for electricity load forecasting using the Louvain algorithm consists of the following five steps:

Step 1 All clients participate in training with the classical differential privacy federated averaging algorithm until convergence.

Step 2 The central server calculates the cosine similarity based on the local update vectors at the convergence of the model from local clients.

Step 3 Each client serves as a network node, constructing a network edge set and edge weights between nodes based on the similarity of learned client parameters, forming an undirected weighted complex network.

Step 4 Community clustering is performed using the Louvain algorithm.

Step 5 The pADP-FedAvg algorithm is utilized for personalized adaptive differential privacy clustering federated learning for electricity load forecasting.

7. Analysis of Arithmetic Examples

This section evaluates the effectiveness of the proposed scheme through arithmetic examples. The experimental environment is as follows: the CPU is an Intel Core i7-12700K, the GPU is an NVIDIA RTX 3080 Ti with 12 GB of memory, the RAM is 64 GB, the operating system is Ubuntu 21.04, and the software versions include PyTorch 2.0.0, CUDA 11.8, and Python 3.8.16. The experiments are conducted in a single-computer deployment mode. The parameter settings for the LSTM model used for load forecasting are detailed in Table 2.

Table 2. Model parameters.

Parameter	Value
Number of epochs of training on clients	8
Total communication rounds	60
LSTM hidden layers	2
Model structure	LSTM layers with 256 hidden states LSTM layers with 128 hidden states Fully connected layers with 64 neurons Fully connected layers with 32 neurons
Input feature dimensions	27
Batch size	128
Learning rate η	0.0015
Dropout probability	0.2
Output feature dimensions	15
Privacy budget ϵ	0.4, 0.6, 0.8
Loss	MSE

7.1. Data Sources

The dataset used is HUE [33], which contains hourly energy use data and meteorological data for 22 homes in British Columbia over the past three years, with a time granularity of 24 points per day. The algorithm selects data from 1 June 2016 to 29 January 2018, specifically for houses with IDs 3–14 and 18–20, totaling 15 users. These users are numbered in descending order by their ID numbers, designated as user 1, 2, . . . , 15. For each responding user, 80% of the dataset is used for training, 10% is used for testing, and 10% is used for validation.

7.2. Data Pre-Processing

After selecting the appropriate data, the data need to be pre-processed. For missing data, this paper utilizes averaging or interpolation methods as follows:

$$\gamma_i = \begin{cases} \frac{\gamma_{i-1}+\gamma_{i+1}}{2}, & \gamma_i \in \text{Null}, \gamma_{i-1}, \gamma_{i+1} \notin \text{Null}; \\ 0, & \gamma_i \in \text{Null}, \gamma_{i-1} \text{ or } \gamma_{i+1} \in \text{Null} \end{cases} \tag{26}$$

where γ_i is the value of electricity load at a certain time period. In addition, in order to accelerate the convergence speed during the model training process and improve training efficiency, the data of continuous values are normalized:

$$\gamma_{\text{nom}} = \frac{\gamma - \gamma_{\text{min}}}{\gamma_{\text{max}} - \gamma_{\text{min}}} \tag{27}$$

where γ is the original data, γ_{nom} is the normalized data, and γ_{max} and γ_{min} are the maximum and minimum values, respectively.

7.3. Evaluation Metrics

To evaluate the accuracy of the algorithm, three metrics are employed: the mean-square error (MSE), root-mean-square error (RMSE), and mean absolute percentage error (MAPE).

The MSE is calculated as follows:

$$MSE = \frac{\sum_{i=1}^{N} (\hat{y}_i - y_i)^2}{N} \tag{28}$$

The RMSE is calculated as follows:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (\hat{y}_i - y_i)^2}{N}} \tag{29}$$

The MAPE is calculated as follows:

$$MAPE = \frac{1}{N} \sum_{i=1}^{N} \frac{|\ \hat{y}_i - y_i\ |}{y_i} \times 100\% \tag{30}$$

where N is the number of test samples and $\hat{y}_i$ and y_i are the actual and predicted readings in kWh, respectively.

7.4. Analysis of Simulation Results

After calculating the similarity, the Louvain algorithm is applied to classify the associations in the weighted network, and the results are displayed in Figure 4.

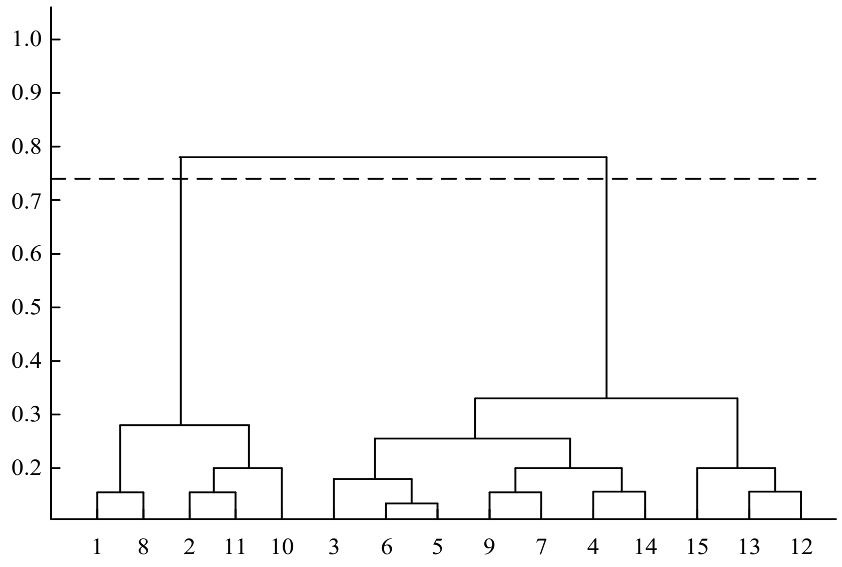


Figure 4. The community division results of the Louvain algorithm.

During the operation, the Q-values are as follows: -0.0832, 0.1569, -0.0275, 0.3124, 0.2017, 0.4753, 0.3695, 0.5936, 0.4821, 0.6427, 0.5339, 0.7052, 0.6185, 0.7514, 0.6793, 0.7836, 0.7253, 0.7453, 0.7292, and 0.7217. After reaching a peak value of 0.7836, the Q-value begins to decline. At this point, the optimal division results in one association clustering for users 1, 2, 8, 10, and 11, and another for users 3, 4, 5, 6, 7, 9, 12, 13, 14, and 15. Compared to commonly used clustering algorithms, such as K-Means, CFL, and IFCA, our proposed method has lower computational costs and does not require pre-specifying the number of clusters.

The comparison results of the DP-FedAvg and pADP-FedAvg algorithms across the three evaluation metrics—MSE, RMSE, and MAPE—are presented in Table 3. With a privacy budget of $\epsilon = 0.6$, it is clear that the pADP-FedAvg algorithm significantly enhances the accuracy of the classical DP-FedAvg algorithm and demonstrates superior performance. Due to the introduction of random noise, the DP noise degrades model performance. Compared with a traditional DP strategy, this suggests that our approach significantly mitigates the negative effects of DP on the model.

Table 3. Comparison of algorithm performance.

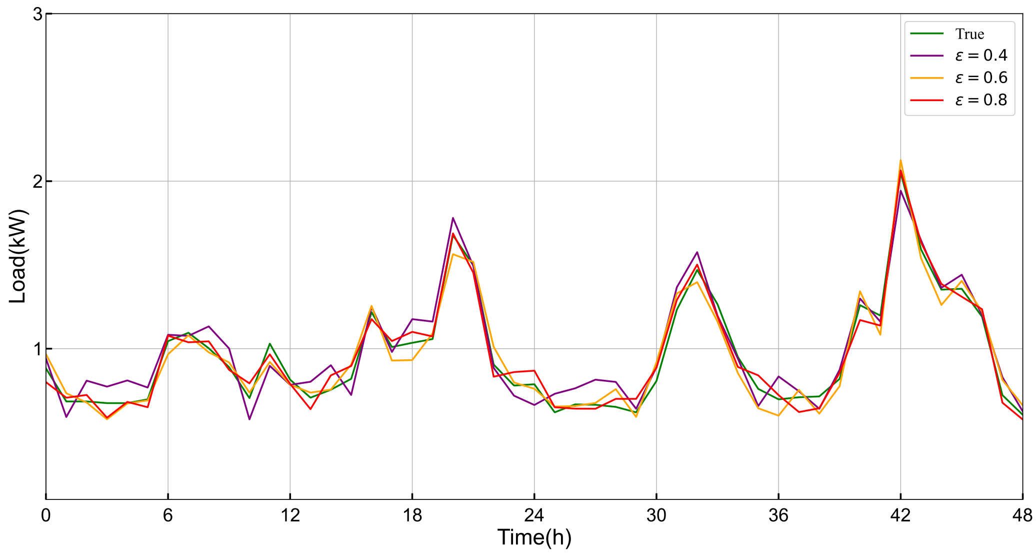
Algorithm	**MSE**	**RMSE**	**MAPE**
DP-FedAvg	0.278	0.527	33.5
pADP-FedAvg	0.205	0.452	29.3

The model performance comparison, as shown in Table 4, evaluates three models under different privacy budgets: the traditional global model, personalized cluster model 1, and personalized cluster model 2. The results indicate that the traditional global model exhibits the lowest accuracy, while the personalized cluster models demonstrate high performance at the same privacy budget. As the privacy budget ϵ increases, the loss metrics (MSE, RMSE, and MAPE) decrease, leading to higher model accuracy; however, this also results in reduced privacy protection. Conversely, a smaller privacy budget ϵ provides greater privacy protection but results in lower prediction accuracy.

Table 4. Comparison of model performance.

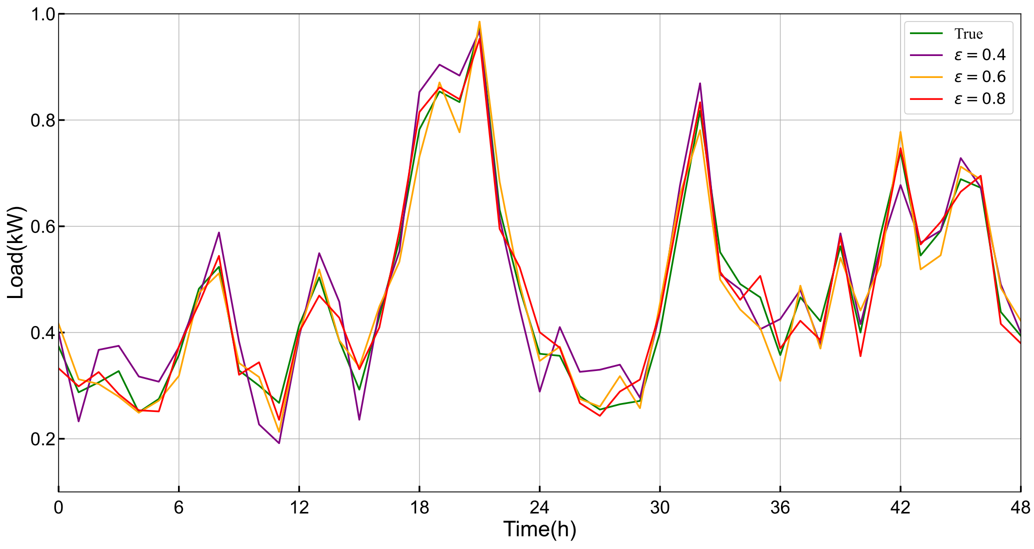
Model	Evaluation Metrics	$\epsilon = 0.4$	$\epsilon = 0.6$	$\epsilon = 0.8$
Global	MSE	0.251	0.205	0.192
	RMSE	0.501	0.452	0.438
	MAPE	35.5	29.3	19.7
Cluster 1	MSE	0.042	0.017	0.009
	RMSE	0.205	0.130	0.095
	MAPE	12.2	6.2	2.9
Cluster 2	MSE	0.031	0.021	0.015
	RMSE	0.176	0.145	0.122
	MAPE	15.1	8.5	3.3

Figure 5 illustrates a comparison of the prediction curves under different privacy-preserving budgets. The experimental results indicate that both models, cluster 1 and cluster 2, exhibit strong prediction performance. When the privacy budget is set higher, their prediction curves align more closely with the true value curves, although this comes at the expense of privacy protection. These findings highlight the trade-off between the privacy-preserving budget and prediction accuracy. In practical applications, it is essential to balance privacy risks and model performance based on specific scenarios to determine the appropriate privacy budget settings. We know that the smaller the privacy budget setting in DP, the better the privacy protection, and the worse the prediction accuracy. Through vertical comparison, under the same privacy budget, the clustered federated load prediction achieves better prediction performance than the global model, which means that our proposed system can achieve a better trade-off between privacy protection and prediction performance.



(**a**)

Figure 5. *Cont.*



(**b**)

Figure 5. Comparison of forecasting results. (**a**) Cluster 1. (**b**) Cluster 2.

8. Conclusions

Accurate forecasting of residential electricity loads is crucial for integrating demand-side resources and fulfilling demand-side response requirements. By addressing the data heterogeneity and security issues in load forecasting for smart grids, this paper presents a load-forecasting framework for residential energy users that enables collaborative training of personalized forecasting models without the need to exchange load data. Weather and time factors are identified as key influences on user loads, and the analysis of arithmetic examples demonstrates that the proposed algorithm outperforms current mainstream federated learning algorithms in prediction performance. This approach enhances load-prediction accuracy while ensuring the privacy and security of user data, which is essential for smart grid planning, operation, control, and dispatch. Future research may focus on other cyber security issues, such as data poisoning and evasion attacks, the influence of unexpected events on power load forecasting, further optimizing the federated learning algorithm, exploring additional application scenarios, and advancing the development of smart grids. The trade-off between model complexity and interpretability is attracting increasing attention. This is critical for the broader adoption of predictive models in real-world settings. While complex models like deep learning algorithms can provide high accuracy, their "black-box" nature makes it difficult to understand how they make predictions. This lack of interpretability can be a significant barrier in practical settings, where understanding the rationale behind a decision is crucial. Therefore, in the future, we will focus on studying the interpretability of the models to enhance user trust in smart grid systems.

Author Contributions: Methodology, T.P., J.H., C.L. and X.C.; Project administration, J.H.; Software, X.J., C.L. and X.C.; Writing—original draft, T.P.; Writing—review and editing, J.H., X.J. and X.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the China Southern Power Grid Corporation Technology Project under grant GDKJXM20222125 (036000KK52222009).

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study.

References

1. Voropai, N. Electric Power System Transformations: A Review of Main Prospects and Challenges. *Energies* **2020**, *13*, 5639. [CrossRef]
2. Zhang, L.; Ren, J.; Mu, Y.; Wang, B. Privacy-Preserving Multi-Authority Attribute-Based Data Sharing Framework for Smart Grid. *IEEE Access* **2020**, *8*, 23294–23307. [CrossRef]
3. Hou, H.; Liu, C.; Wang, Q.; Wu, X.; Tang, J.; Shi, Y.; Xie, C. Review of load forecasting based on artificial intelligence methodologies, models, and challenges. *Electr. Power Syst. Res.* **2022**, *210*, 108067. [CrossRef]
4. Dudek, G. Pattern-based local linear regression models for short-term load forecasting. *Electr. Power Syst. Res.* **2016**, *130*, 139–147. [CrossRef]
5. Walser, T.; Sauer, A. Typical load profile-supported convolutional neural network for short-term load forecasting in the industrial sector. *Energy AI* **2021**, *5*, 100104. [CrossRef]
6. Karaman, O.A. Prediction of Wind Power with Machine Learning Models. *Appl. Sci.* **2023**, *13*, 11455. [CrossRef]
7. Cicceri, G.; Tricomi, G.; D'Agati, L.; Longo, F.; Merlino, G.; Puliafito, A. A Deep Learning-Driven Self-Conscious Distributed Cyber-Physical System for Renewable Energy Communities. *Sensors* **2023**, *23*, 4549. [CrossRef]
8. McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; Arcas, B.A.Y. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, Lauderdale, FL, USA, 20–22 April 2017; Singh, A., Zhu, J., Eds.; Volume 54, pp. 1273–1282.
9. Niknam, S.; Dhillon, H.S.; Reed, J.H. Federated Learning for Wireless Communications: Motivation, Opportunities, and Challenges. *IEEE Commun. Mag.* **2020**, *58*, 46–51. [CrossRef]
10. Husnoo, M.A.; Anwar, A.; Hosseinzadeh, N.; Islam, S.N.; Mahmood, A.N.; Doss, R. A Secure Federated Learning Framework for Residential Short-Term Load Forecasting. *IEEE Trans. Smart Grid* **2024**, *15*, 2044–2055. [CrossRef]
11. Fernández, J.D.; Menci, S.P.; Lee, C.M.; Rieger, A.; Fridgen, G. Privacy-preserving federated learning for residential short-term load forecasting. *Appl. Energy* **2022**, *326*, 119915. [CrossRef]
12. Fekri, M.N.; Grolinger, K.; Mir, S. Distributed load forecasting using smart meter data: Federated learning with Recurrent Neural Networks. *Int. J. Electr. Power Energy Syst.* **2022**, *137*, 107669. [CrossRef]
13. Savi, M.; Olivadese, F. Short-Term Energy Consumption Forecasting at the Edge: A Federated Learning Approach. *IEEE Access* **2021**, *9*, 95949–95969. [CrossRef]
14. Zhu, L.; Liu, Z.; Han, S. Deep leakage from gradients. In Proceedings of the 33rd International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; Curran Associates Inc.: Red Hook, NY, USA, 2019.
15. Dwork, C.; Roth, A. The Algorithmic Foundations of Differential Privacy. *Found. Trends® Theor. Comput. Sci.* **2014**, *9*, 211–407. [CrossRef]
16. Abadi, M.; Chu, A.; Goodfellow, I.; McMahan, H.B.; Mironov, I.; Talwar, K.; Zhang, L. Deep Learning with Differential Privacy. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, CCS '16, Vienna, Austria, 24–28 October 2016; pp. 308–318. [CrossRef]
17. Hu, R.; Guo, Y.; Gong, Y. Concentrated Differentially Private Federated Learning with Performance Analysis. *IEEE Open J. Comput. Soc.* **2021**, *2*, 276–289. [CrossRef]
18. Sun, X.; Yuan, Z.; Kong, X.; Xue, L.; He, L.; Lin, Y. Communication-Efficient and Privacy-Preserving Aggregation in Federated Learning with Adaptability. *IEEE Internet Things J.* **2024**, *11*, 26430–26443. [CrossRef]
19. Xue, R.; Xue, K.; Zhu, B.; Luo, X.; Zhang, T.; Sun, Q.; Lu, J. Differentially Private Federated Learning with an Adaptive Noise Mechanism. *IEEE Trans. Inf. Forensics Secur.* **2024**, *19*, 74–87. [CrossRef]
20. Liu, M.; Teng, F.; Zhang, Z.; Ge, P.; Sun, M.; Deng, R.; Cheng, P.; Chen, J. Enhancing Cyber-Resiliency of DER-Based Smart Grid: A Survey. *IEEE Trans. Smart Grid* **2024**, *15*, 4998–5030. [CrossRef]
21. Zhang, Z.; Liu, M.; Sun, M.; Deng, R.; Cheng, P.; Niyato, D.; Chow, M.Y.; Chen, J. Vulnerability of Machine Learning Approaches Applied in IoT-Based Smart Grid: A Review. *IEEE Internet Things J.* **2024**, *11*, 18951–18975. [CrossRef]
22. Ma, X.; Zhu, J.; Lin, Z.; Chen, S.; Qin, Y. A state-of-the-art survey on solving non-IID data in Federated Learning. *Future Gener. Comput. Syst.* **2022**, *135*, 244–258. [CrossRef]
23. Sattler, F.; Müller, K.R.; Samek, W. Clustered Federated Learning: Model-Agnostic Distributed Multitask Optimization Under Privacy Constraints. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *32*, 3710–3722. [CrossRef]
24. Gao, Z.; Xiong, Z.; Zhao, C.; Feng, F. Clustered Federated Learning Framework with Acceleration Based on Data Similarity. In *Algorithms and Architectures for Parallel Processing*; Tari, Z., Li, K., Wu, H., Eds.; Springer: Singapore, 2024; pp. 80–92.
25. Ghosh, A.; Hong, J.; Yin, D.; Ramchandran, K. Robust Federated Learning in a Heterogeneous Environment. *arXiv* **2019**, arXiv:1906.06629.
26. Ghosh, A.; Chung, J.; Yin, D.; Ramchandran, K. An Efficient Framework for Clustered Federated Learning. *IEEE Trans. Inf. Theory* **2022**, *68*, 8076–8091. [CrossRef]

27. Bun, M.; Steinke, T. Concentrated Differential Privacy: Simplifications, Extensions, and Lower Bounds. In *Theory of Cryptography*; Hirt, M., Smith, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2016; pp. 635–658.
28. Yu, L.; Liu, L.; Pu, C.; Gursoy, M.E.; Truex, S. Differentially Private Model Publishing for Deep Learning. In Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 19–23 May 2019; pp. 332–349. [CrossRef]
29. Geyer, R.C.; Klein, T.; Nabi, M. Differentially Private Federated Learning: A Client Level Perspective. *arXiv* **2017**, arXiv:1712.07557.
30. McMahan, H.B.; Ramage, D.; Talwar, K.; Zhang, L. Learning Differentially Private Recurrent Language Models. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
31. Blondel, V.D.; Guillaume, J.L.; Lambiotte, R.; Lefebvre, E. Fast unfolding of communities in large networks. *J. Stat. Mech. Theory Exp.* **2008**, *2008*, P10008. [CrossRef]
32. Shi, H.; Xu, M.; Li, R. Deep Learning for Household Load Forecasting—A Novel Pooling Deep RNN. *IEEE Trans. Smart Grid* **2018**, *9*, 5271–5280. [CrossRef]
33. Makonin, S. HUE: The hourly usage of energy dataset for buildings in British Columbia. *Data Brief* **2019**, *23*, 103744. [CrossRef]