


Research Article

Full Data-Processing Power Load Forecasting Based on Vertical Federated Learning

Zhengxiong Mao,¹ Hui Li,¹ Zuyuan Huang,¹ Yuan Tian,¹ Peng Zhao ,² and Yanan Li³

¹Network Information Center, Yunnan Power Grid Company Limited, Kunming, China

²School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an, China

³School of Mathematics and Statistics, Xi'an Jiaotong University, Xi'an, China

Correspondence should be addressed to Peng Zhao; p.zhao@xjtu.edu.cn

Received 13 September 2022; Revised 15 December 2022; Accepted 23 December 2022; Published 13 January 2023

Academic Editor: Yang Li

Copyright © 2023 Zhengxiong Mao et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Power load forecasting (PLF) has a positive impact on the stability of power systems and can reduce the cost of power generation enterprises. To improve the forecasting accuracy, more information besides load data is necessary. In recent years, a novel privacy-preserving paradigm vertical federated learning (FL) has been applied to PLF to improve forecasting accuracy while keeping different organizations' data locally. However, two problems are still not well solved in vertical FL. The first problem is a lack of a full data-processing procedure, and the second is a lack of enhanced privacy protection for data processing. To address it, according to the procedure in a practical scenario, we propose a vertical FL XGBoost-based PLF, where multiparty secure computation is used to enhance the privacy protection of FL. Concretely, we design a full data-processing PLF, including data cleaning, private set intersection, feature selection, federated XGBoost training, and inference. Furthermore, we further use RSA encryption in the private set intersection and Paillier homomorphic encryption in the training and inference phases. To validate the proposed method, we conducted experiments to compare centralized learning and vertical FL on several real-world datasets. The proposed method can also be directly applied to other practical vertical FL tasks.

1. Introduction

Power load forecasting (PLF) has important significance in the development of the power grid and its upstream and downstream enterprises. Based on the temporal and spatial distributions derived from PLF, the power grid can effectively plan and make scientific decisions to stabilize the power system and avoid unnecessary grid connections. Meanwhile, upstream power generation enterprises can schedule the generator set to realize off-peak production and reduce the cost [1].

Thanks to smart grid and big data technology, the methods for PLF have been developed from a multivariate linear model to a deep learning model. Meanwhile, regardless of the methods, the performance highly depends on the available data, which have been also changed from a single source to multiple parties [2]. With the progress of

methods and federated data, EFL has been applied from short-term forecasting to long-term forecasting and the accuracy has been significantly improved. At the same time, some new challenges have emerged. One of which is how to ensure privacy and security when different parties provide their data for federated data analysis [3].

The top of Figure 1 demonstrates the basic pipeline of the centralized PLF. At first, the power grid collects the related data, such as weather and power generation data, from other industries. Then, the power grid is responsible for data processing. Finally, it trains the predefined model and uses it for inference. Note that in the centralized EFL, all required data have been collected before formal analysis. For example, the power grid needs to collect enterprises' production efficiency, equipment status, loans, etc., to improve the prediction accuracy of EFL. However, because the above-mentioned information is sensitive and private, the

enterprises are reluctant to provide the original data [4]. In addition, many countries have enacted privacy regulations to restrict and regulate data usage, such as the General Data Protection Regulation of the European Union and the Data Security Law of the People's Republic of China. Therefore, how to fulfil the data federation while ensuring all parties' data privacy has important practical significance and scientific value.

As a novel privacy-preserving paradigm, federated learning (FL) was formally proposed by Google in 2016 to securely collaborate on massive mobile devices and then applied to many practical scenarios. Generally, according to the data distribution among different parties, FL is further categorized as horizontal FL and vertical FL, where the former is an extension of samples and the latter is an extension of features. The paper considers a vertical FL for PLF, whose pipeline is shown in the bottom box of Figure 1. The difference between vertical FL and centralized learning is mainly reflected in the following three aspects: (1) In vertical FL, different parties cannot directly share the original data. (2) In vertical FL, different parties must execute the additional private set intersection (PSI) to obtain the shared sample ID. (3) In vertical FL, different parties only exchange the ciphertext to prevent indirect privacy leakage.

Although there exist some investigations about how to apply vertical FL to PLF, two problems are still not well solved. The first problem is a lack of a full data-processing procedure, and the second is a lack of enhanced privacy protection in a full data-processing procedure. Existing research focuses on model architecture selection and usually assumes that the dataset has been well preprocessed. However, in a practical scenario, several data pretreatments are necessary before formal training. Besides, FL only protects the original data ownership but cannot prevent indirect privacy leakage by inferring intermediate information.

To address it, according to the procedure in a practical scenario, we propose a vertical FL XGBoost-based PLF [5, 6], where multiparty secure computation is used to enhance the privacy protection of FL. The reason for choosing XGBoost rather than LSTM is that we focus on the short-term PLF rather than the long-term EFL. In such cases, XGBoost has a faster computation speed. Besides, XGBoost prefers category data, which cannot be dealt by LSTM. The considered scenario is that power grid, weather bureau, and electric enterprises are trying to jointly improve the prediction accuracy of EFL, with the restriction of privacy protection. Different from the existing vertical XGBoost study, which just proposes the vertical XGBoost algorithm, we here consider a full data-processing vertical PLF, and multiparty secure computation is introduced to enhance the security of full data processing. The main contributions are as follows:

- (i) We propose a full data-processing vertical XGBoost method for PLF, which consists of data cleaning, PSI, feature selection, feature binning, federated training, and inference. The proposed method can also be applied to other vertical FL tasks.

- (ii) We further use RSA encryption in private set intersection and Paillier homomorphic encryption in the training and inference phases to enhance the security of the full data-processing procedure.
- (iii) We evaluate the proposed method on three real-world datasets to compare centralized learning, two-party, and three-party scenarios. Experimental results show that FL can significantly improve forecasting accuracy, and three-party FL outperforms two-party FL.

The organization is as follows: Section 2 introduces the related work, and Section 3 introduces preliminary knowledge. We introduce the main method in Section 4 and validate the proposed method in Section 5. We give a brief summary and future work in Section 6.

2. Related Work

2.1. Centralized Learning. In centralized learning, the main work for PLF is how to establish the model. The existing models include three aspects. The first aspect is using traditional statistical models, such as multivariate linear model [7] or Bayesian model [8]. López et al. [9] estimated the ARIMA parameter based on the frequency domain and the Bayes method and used the ARIMA for short-term EFL. The second aspect is using data-driven methods, such as machine or deep learning models. For example, Chen et al. [10] proposed a deep random forest for short-term PLF, and Singh and Dwivedi [11] introduced a neural network for EFL. The third aspect is using the combinational method. For example, Zhang and Wang [13] simultaneously used singular spectrum analysis, support vector machine, averaging regression model, and cuckoo searching for short-term PLF. Liang et al. [14] fulfilled EFL based on empirical mode decomposition, MIRMR, linear regression network, and fruit fly optimization algorithm.

2.2. Federated Learning. To mitigate privacy leakage caused by direct data sharing, FL has been applied to many applications in the smart grid [15–17]. Hudson et al. [18] proposed a deep recurrent network for nonintrusive load monitoring via FL. Wang et al. [19] proposed a horizontal FL approach to study electricity consumer characteristic identification, where privacy-preserving PCA is used to reduce the dimension. On the same topic, Lin et al. [20] proposed a hybrid model combining CNN and LSTM to learn consumer features and used an asynchronous strategy to improve the training speed. Cao et al. [21] used local differential privacy to enhance the security of FL, where users are divided into regular and sensitive groups, each with a different privacy budget. Su et al. [22] proposed a secure and efficient FL-enabled AI of things scheme for private energy data sharing, where a reinforcement learning-based incentive algorithm is used to improve the model's utility. However, all the above work only considered the horizontal FL framework, which cannot be extended to the scenario considered in the paper.

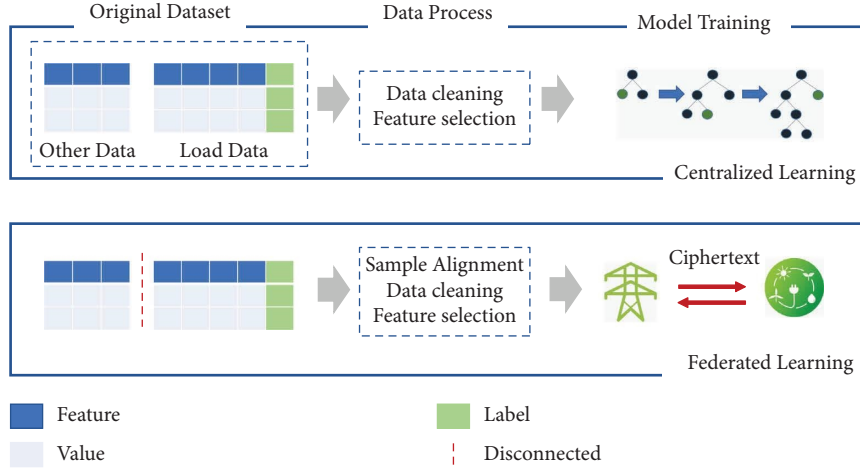


FIGURE 1: Pipeline of centralized learning pipeline and federated learning.

Taik and Cherkaoui [23] considered a similar application to ours, where an FL is used for household load forecasting. However, they also considered the horizontal FL, which cannot be extended to the vertical scenario considered in the paper. The work most similar to ours is [24], where vertical FL XGBoost is applied to power consumption forecast. The difference is that the study [24] only discusses how to use an encryption scheme to enhance the training phase, but we give a complete vertical FL framework based on a practical data processing procedure.

3. Preliminary Knowledge

As discussed above, we aim to propose the full data processing procedure for PLF. For a comprehensive understanding, we introduce the core concepts in this section, including the vertical FL framework and XGBoost algorithm.

3.1. Vertical Federated Learning. FL is a special framework of distributed learning, which consists of multiple parties. Apart from distributed learning, FL focuses the parties' privacy and the curator (if one exists) cannot access any party's raw data. The basic idea of FL is that transmitting the intermediate information will replace the raw data. The intermediate information is usually model parameters or gradients, in plain text or ciphertext.

Vertical FL is adapted to the scenario where the sample features overlap more than the sample IDs. Figure 2 shows a case of two parties' data, which satisfy the vertical FL. In particular, party A owns four samples with id_1, id_2, id_3 , and id_5 and features f_1 and f_2 . Party B owns four samples with id_1, id_2, id_3 , and id_4 and three features f_3, f_4 , and f_5 . Clearly, the purpose of federation in such a case is to enlarge the feature space. However, the beforehand step is to find the overlapped IDs id_1, id_2 , and id_4 , meanwhile, without revealing other exclusive IDs. After this step, parties A and B will apply the federated algorithm to the intersection. Different algorithms correspond to different procedures. For example, the core of XGBoost is

to compare the gain of all possible values of features, where the gradients can be independently computed by the party who owns labels. However, for the CNN model under the vertical FL, neither party can derive full gradients, and the core problem is how to securely exchange partial information to compose lossless gradients. In summary, whatever the algorithm, all parties in vertical FL need to compute the intersection at first, which is called the private set intersection (PSI).

3.2. XGBoost Algorithm. XGBoost [5] is an improved version of GBDT. For example, XGBoost uses the second expansion to approximate the loss function to improve the prediction accuracy and introduces a regular term to avoid overfitting. Besides, XGBoost utilizes blocks storage structure to fulfil parallel computation.

The objective function of XGBoost consists of loss function and regularization. Assume the training dataset is $\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$, with loss value $l(y_i, \hat{y}_i)$ and regular term $\Omega(f_k)$. Then, the objective function is

$$L(\phi) = \sum_{k=1}^K l(y_i, \hat{y}_i) + \sum_k \Omega(f_t(x_i)), \quad (1)$$

where k denotes the tree index and \hat{y}_i is the prediction value of input x_i . That is, $\hat{y}_i = \sum_{k=1}^t f_k(x_i)$. The core idea of XGBoost is that based on the first $t-1$ trained trees, the total loss value can be expressed as $\hat{y}_i = \sum_{k=1}^{t-1} f_k(t) = \hat{y}_i^{(t-1)} + f_t(x_i)$. To train the t -th tree, the objective is

$$L^{(t)} = \sum_{i=1}^n l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \sum_k \Omega(f_k). \quad (2)$$

Taking the quadratic approximation of the above function and removing the constant factor, we obtain the following expression of objective function:

$$L^{(t)} = \sum_{j=1}^T \left[G_j w_j + \frac{1}{2} (H_j + \lambda) w_j^2 \right] + \gamma T, \quad (3)$$

Party A			Party B			
ID	f_1	f_2	ID	f_3	f_4	f_5
id_1			id_1			
id_2			id_2			
id_4			id_3			
id_5			id_4			

FIGURE 2: Data split of parties in vertical FL. The horizontal axis denotes sample ID, and the vertical axis denotes feature.

where $G_j = \sum_{i \in I} g_i$ and $H_i = \sum_{i \in I} h_i$ with g_i and h_i are the 1st and 2nd derivations of sample i and the purpose is to minimize (2) with respect to w_j . It is observed that (2) is a summation of T independent quadratic functions. Through a basic calculation, we obtain the optimal argument $w_j = G_j/H_j + \lambda$ and the minimal value $L_{\min}^{(t)} = -1/2 \sum_{j=1}^T G_j^2/H_j + \lambda + \gamma T$. The remainder problem is how to generate the tree. The principle is to select a feature split, which ensures the fastest descend of (2). Assume I_l and I_r are the left and right sets based on a given split. Then, the corresponding loss function is

$$L_{\text{split}} = \frac{1}{2} \left[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda} \right] - \gamma. \quad (4)$$

Based on the maximal value selection principle, we select the corresponding feature value to split tree and write the feature name and value into the splitting nodes. Then, the depth of the tree increases by one. Repeating the above procedure until the tree grows to the given depth.

4. Main Methods

This section introduces two main methods, private set intersection and vertical XGBoost, for full data-processing vertical PLF.

4.1. Basic Idea. The paper proposes a lossless, full data-processing vertical PLF. The basic idea is to separate operations among participants based on the data/information flow, which also encrypts the exchanged information for enhanced privacy protection. In concrete, participants are divided into the active party (i.e., the server) and passive party (i.e., the client), where only the server owns the label.

Assume that the participants consist of one server and multiple clients. For a simple explanation, we assume in this section that the server has a dataset $\{u_1^A, u_2^A, u_3^A, u_4^A\}$, where $u_i^A = \{id_i, f_i^A, y_i\}$ is composed by the sample ID id_i , feature f_i^A , and label y_i . The client has $\{u_1^B, u_2^B, u_4^B, u_5^B\}$, where $u_i^B = \{id_i, f_i^B\}$. Note that the client does not have the label.

4.2. Private Set Intersection. Private set intersection (PSI) is used for searching the sample ID intersection between the server and client, without revealing the individual's ID. The main procedure is shown in Figure 3, where the hash function and blind signature are used for ensuring private and correct information communication. The details are as follows:

- (1) Client generates the public and private keys (n, e, d) and sends public key (n, e) to the server
- (2) Server maps the sample ID X_A to $F(u)$ through a hash function and then sends the blind Y_A to the client
- (3) Client obtains Z_A through the blind signature of Y_A and D_B by handling X_B through hash-signature-hash operation, where Z_A is sent to the server
- (4) Server unblinds Z_A and uses hash mapping to obtain D_A
- (5) Server finds the intersection between D_A and D_B and then sends the intersection to the client
- (6) Client receives the intersection

4.3. Vertical XGBoost. XGBoost is an improved version of GBDT. XGBoost consists of multiple trees, and the prediction result is the summation of all the trees. The purpose of XGBoost is to generate the t -th tree to fit the residual of the first $t-1$ trees. We use XGBoost for regression prediction, where the loss function is defined as

$$\mathbf{l}(y - \hat{y}) = \sum_{i=1}^n [l(y_i - \hat{y}_i)]^2. \quad (5)$$

The first derivation g_i and the second derivation h_i are $g_i = |y_i - \hat{y}_i|$, $h_i = 1$. Denote by I the set of samples ID. Based on (4), the gain of a splitting node is

$$L_{\text{split}} = \frac{1}{2} \left[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda} \right] - \gamma, \quad (6)$$

where $G_R = \sum_{i \in I_R} g_i$, $H_R = \sum_{i \in I_R} h_i$, $G_L = \sum_{i \in I_L} g_i$, and $H_L = \sum_{i \in I_L} h_i$, with I_R and I_L are sets of sample ids which are, respectively, split into the right and left sides of the splitting node. The growth of the tree depends on the selection of the splitting node, which corresponds to the maximal value L_{split} . Once a node is selected in a tree, the following growth is restricted to the subsets I_R or I_L .

When the growth of a tree is complete, its structure will only be restored only on the server. For protecting all participants' privacy, each party maintains herself/himself lookup table, which has the information of splitting feature and value. The splitting node contains two messages. One is the party identity who owns the splitting feature, and the other is the sample index of the lookup table. Therefore, even if the server owns the complete structure, he or she cannot complete the inference independently. Figure 4 shows the procedure of vertical XGBoost:

- (1) The server computes the first-order and second-order derivations (G, H) based on labels $\{y_i\}_{i \in I}$, where G and H are sets which consist of g_i and h_i shown as (4). Then, the server sends the encryption of (G, H) to the client. Meanwhile, the server obtains the optimal splitting node S_A by computing all splitting gains on its own feature values.
- (2) The client receives the encrypted (G, H) , which is used for computing the splitting gains of the client's

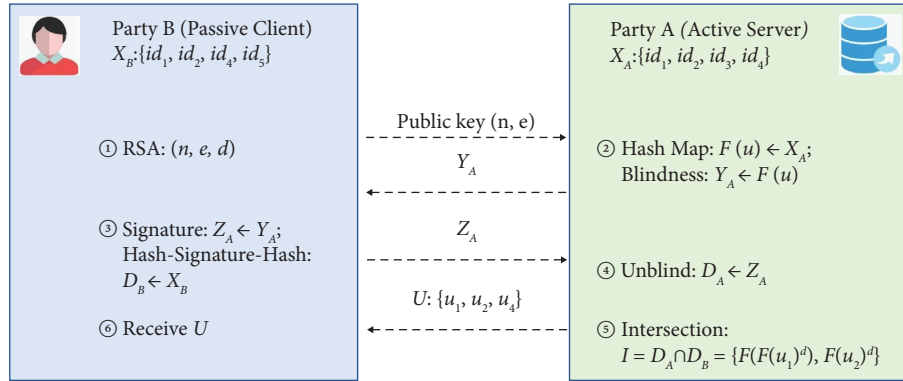


FIGURE 3: Demonstration of private set intersection.

feature values. Then, the client sends all splitting gains to the server.

- (3) The server receives the splitting gains sent by the client and obtains the optimal splitting node S_B of client. By comparing S_A and S_B , the server obtains the globally optimal splitting node from $\{S_A, S_B\}$.
- (4) The participant who owns the optimal splitting node writes the corresponding information into the lookup table and generates a new node.

Remark 1. The proposed vertical PFL can be easily extended to multiple clients. Due to the fact that all clients concurrently calculate their own splitting gains, the local computation time is almost the same as for a single client when they are homogeneous. Note that only the server needs to aggregate more intermediate information uploaded from multiple clients; therefore, the proposed method can be applied to multiple clients. This is validated by the experimental results in Section 5.4.

5. Evaluation

Based on a real-world dataset, the experimental results show that the proposed federated XGBoost can significantly improve the performance of the centralized XGBoost, in terms of root mean square error (RMSE) and R^2 score.

5.1. Dataset. We validated the proposed full data processing FL method on three real-world datasets, which were loaded from the Kaggle website: (1) The first dataset is made for project hourly electricity consumption prediction of Homestead city (available at <https://www.kaggle.com/datasets/unajtheb/homesteadus-electricity-consumption>). (2) The second dataset is made for optimizing the electricity production of the Electrolysis company (available at <https://www.kaggle.com/datasets/utathya/electricity-consumption>). (3) The third dataset is made for forecasting electricity load or power price (available at <https://www.kaggle.com/datasets/nicholasjhana/energy-consumption-generation-prices-and-weather>). More information is listed in Table 1, and detailed information about three datasets can be found on the above websites.

5.2. Experimental Settings

5.2.1. Simulated Scenarios. For PLF using XGBoost, we consider three modes, including centralized learning, two-party vertical FL, and three-party vertical FL. In each mode, we take the input as consecutive t data (such as electric load, weather, etc.) and output the power load from the timestamp $t + 1$ to $t + p$. Furthermore, we set t as 6, 12, 24, and p as 1, 2, 3, 4, 5, 6. For example, $t = 6, p = 1$ in centralized mode denotes that we use the consecutive 6 power loads to predict the 7-th power load. The considered modes are described as follows:

- (1) Centralized learning: the power grid only uses the historical power load for PLF.
- (2) Two-party FL: participants include the power grid and weather bureau. Model inputs consist of historical power load and weather information.
- (3) Three-party FL: participants include power grid, weather bureau, and power generation enterprises. Model inputs consist of power load, weather information, and power generation values.

5.2.2. Platform Parameters. We used three microservers to simulate the active server and two passive clients. The configuration of each microserver was as follows: Intel Xeon CPU E5-2650 and network card RTL8111 with bandwidth 1000 Mb/s. The operating system version was Ubuntu 18.04. All experimental codes were written in Python 3.7.

5.2.3. Algorithm Parameters. We used grid search to tune the main parameters. Five key parameters of XGBoost were set as follows: the tree number was 40, left sampling weight was 1, and initial label value was 0.5 for all three datasets. The maximum depth was set as 5, 3, and 2, and the regularization was set as 10, 15, and 40, for dataset-1, dataset-2, and dataset-3, respectively. Besides, the learning rate was 0.1, and the ratio of training to test dataset was 9: 1.

5.3. Experimental Procedure. In this section, we take dataset-3 as example to introduce the standard FL procedure, which can be applied to other similar vertical FL XGBoost tasks:

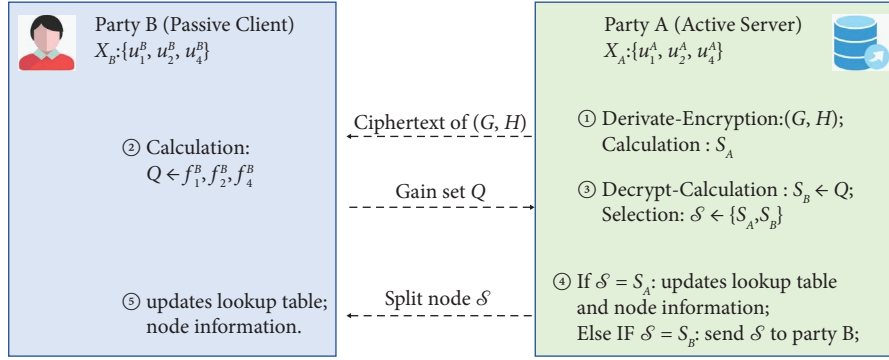


FIGURE 4: Demonstration of the splitting node in vertical XGBoost.

TABLE 1: Descriptions of three used real-world datasets.

Dataset	Party	Original size	After cleaning	After aligning	After selecting	Experiment
Dataset-1	Electricity	26496.2	26496.2	26496.2	26496.2	Centralized
	Weather	26496.6	26496.6	26496.6	26496.5	
Dataset-2	Electricity	22200.2	22200.2	22200.2	22200.1	Two-party FL
	Weather	22200.16	22200.16	22200.16	22200.13	
Dataset-3	Electricity	35017.2	35017.2	35017.2	35017.2	Centralized
	Energy	178396.17	178396.17	35017.17	35017.14	Two-party FL
	Weather	35017.28	35017.21	35017.21	35017.9	Three-party FL

- (1) Data cleaning: we replaced the categorical data with integers and removed the null data, including the weather main, weather description, weather icon, and city name, with integers, which ranged from 1 to the domain size. We removed eight null features from the energy dataset. The cleaned dataset size is described in Table 1.
- (2) Private set intersection: we used the private set intersection to obtain and align the common sample IDs among the used datasets. The basic procedure is described in Figure 3, where the sample ID is the timestamp. The dataset size after aligning is described in Table 1.
- (3) Feature selection: considering some features are irrelative to PLF, we calculated the related coefficients between features and label, and removed the features whose value was less than 0.3. The final remaining features have been listed in Section 5.2 and Table 1.
- (4) Federated training: participants collaboratively compute the splitting gains of all candidate values to grow the tree according to Figure 4. Due to that only the server has the label, the computation of the first-order and second-order derivations are computed by the server. Then, the crypted derivations were sent to clients who computed their own gains by splitting the ciphertext. Finally, the server obtained the optimal splitting value by comparing the union set of splitting gains.
- (5) Federated inference: the server and client collaboratively used themselves to predict the power load. If the split feature belongs to the server, then the server judges where the data flow goes, into the left or right

child nodes. Else, it is the client's turn to make the judgement based on splitting information stored in the lookup table, which is owned by the server.

5.4. Experimental Results. Based on the experimental settings described in Section 5.1, the comparison results are shown in Figure 5 and Table 2. From this, we obtain the overall conclusion that FL outperforms centralized learning. More detailed results are as follows:

- (1) Two-party federated learning versus centralized learning: Figure 5 shows R^2 on three datasets when input size t was set as 6, 12, 24 and prediction length p was set from 1 to 6. In centralized learning, we only used the electricity consumption data. However, in two-party FL, we collaboratively used electricity and weather data. Three conclusions were observed from Figure 5.

Firstly, it is observed from each subfigure that FL outperforms centralized. The advantage increases with the horizontal axis, i.e., the prediction length. The reason is that FL utilizes weather information, which is related to power consumption. Secondly, it is observed that R^2 increases with input parameter t . Taking $p = 6$ on dataset-3 as example (Figures 5(g)–5(i)), R^2 of centralized learning when $t = 6, 12, 24$ is 0.54, 0.88, 0.89, respectively. Meanwhile, R^2 of FL in the same setting is 0.84, 0.90, 0.92, which also presents an increasing trend. The reason is that we observe a clearer tendency from more historical power load data. Similar conclusions are also observed from other figures. Thirdly, it is observed that

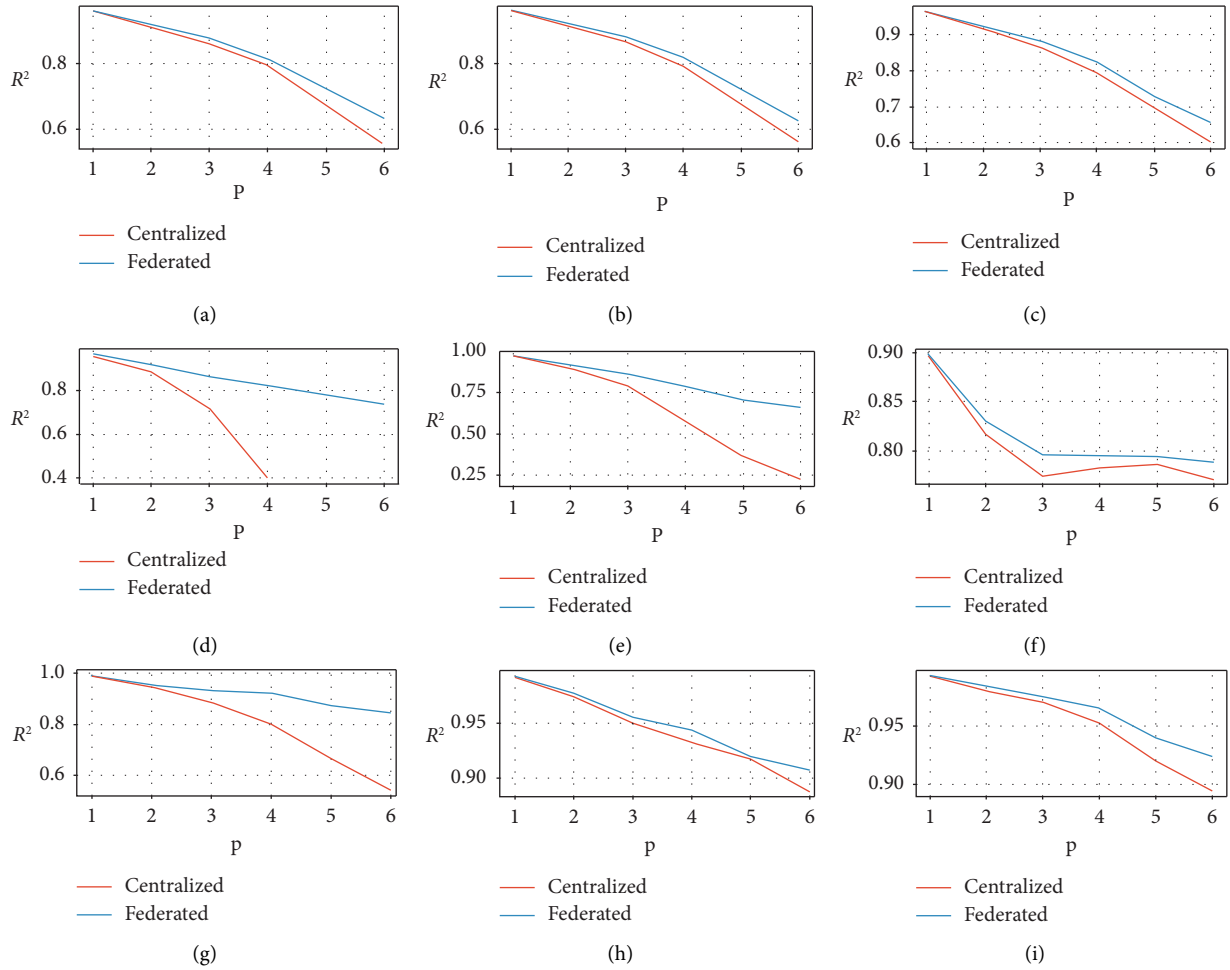


FIGURE 5: Comparison of centralized learning and two-part federated learning when $t = 6, 12, 24$. (a) Dataset-1, $t = 6$. (b) Dataset-1, $t = 12$. (c) Dataset-1, $t = 24$. (d) Dataset-2, $t = 6$. (e) Dataset-2, $t = 12$. (f) Dataset-2, $t = 24$. (g) Dataset-3, $t = 6$. (h) Dataset-3, $t = 12$. (i) Dataset-3, $t = 24$.

TABLE 2: Comparison of R^2 and runtime between centralized learning and three-party FL on dataset-3.

$p =$		1	2	3	4	5	6	Average runtime
$k = 6$	Centralized	0.987	0.947	0.886	0.803	0.665	0.540	—
	Two-party	0.988	0.953	0.933	0.922	0.874	0.843	66 m
	Three-party	0.992	0.975	0.957	0.938	0.921	0.911	79 m
$k = 12$	Centralized	0.992	0.974	0.951	0.933	0.918	0.887	—
	Two-party	0.992	0.977	0.956	0.943	0.920	0.906	84 m
	Three-party	0.994	0.981	0.968	0.952	0.938	0.926	98 m
$k = 24$	Centralized	0.993	0.980	0.970	0.952	0.921	0.895	—
	Two-party	0.993	0.985	0.976	0.964	0.940	0.924	108 m
	Three-party	0.994	0.984	0.973	0.963	0.952	0.944	122 m

compared to centralized learning, the advantage of FL decreases as t . For example, when $t = 6$ on dataset-3, the increasement is 0.3. However, the increasement is only 0.03 when $t = 24$. The reason is that the information gain of weather data decreases as more historical load data are available.

- (2) Three-party federated learning versus two-party federated learning: we further conducted three-party FL where additionally used energy dataset to validate

the advantage of more data collaboration. The differences between three-party FL and two-party FL reflect two parts. The first difference is that the three parties have to consequently aligned the sample ID one by one. The second is that there are two passive parties in three-party FL but one passive party in two-party FL. The comparison of R^2 and runtime are listed in Table 2, from which it is clearly observed that three-party FL achieved a higher R^2 than two-

party FL, at the cost of little more computation time. The reason is that the additional energy data can provide more useful information for PLF, while the active server in three-party FL has to aggregate more information than two-party FL. In particular, it is observed that the runtime is mainly determined by client number, which is consistent with the remark in Section 4.3.

6. Conclusion

Considering data silos in the power load forecast, we propose a vertical federated learning method to collaboratively collect more data to improve the forecast accuracy. Furthermore, according to data processing procedure, we design a full data-processing XGBoost algorithm, where we use homomorphic encryption for protecting privacy. Experimental results on real-world datasets show that the proposed method can outperform the traditional centralized learning method. Therefore, the proposed vertical federated learning XGBoost algorithm with full data processing consideration can be directly applied to improving the utility of power load forecast.

However, due to the fact that the server has to send the encrypted first-order and second-order derivations of each sample to clients, the communication cost increases linearly with database size. There will be a barrier to applying XGBoost when the database size is huge in practice. This can be mitigated by clustering derivations and only the clustered results, instead of each of derivations, are sent to clients. Meanwhile, the batch-encrypted method, which encrypts a batch at a time, can also be utilized. We will investigate these strategies to further reduce the communication cost in our future work [12].

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study was supported by the Yunnan Power Grid Technology Project (YNKJXM20210141).

References

- [1] J. Chen, "Application of grey theory and artificial neural network in medium- and long-term load forecasting of power system," Master's thesis, Suzhou University, Jiang Su Sheng, China, 2019.
- [2] C. Kuster, Y. Rezgui, and M. Mourshed, "Electrical load forecasting models: a critical systematic review," *Sustainable Cities and Society*, vol. 35, pp. 257–270, 2017.
- [3] M. Massaoudi, H. Abu-Rub, S. S. Refaat, I. Chihi, and F. S. Oueslati, "Deep learning in smart grid technology: a review of recent advancements and future prospects," *IEEE Access*, vol. 9, pp. 54558–54578, 2021.
- [4] K. Li, R. Shi, and E. Li, "Survey on data aggregation and privacy protection of user query in smart grid," *Netinfo Security*, vol. 11, pp. 65–73, 2021.
- [5] T. Chen, T. He, M. Benesty, V. Khotilovich, Y. Tang, and H. Cho, "Xgboost: extreme gradient boosting," *R package version 0.4-2*, vol. 1, no. 4, pp. 1–4, 2015.
- [6] K. Cheng, T. Fan, Y. Jin et al., "Secureboost: a lossless federated learning framework," *IEEE Intelligent Systems*, vol. 36, no. 6, pp. 87–98, 2021.
- [7] S. Lei, C. Sun, Q. Zhou, and X. Zhang, "The research of local linear model of short-term electrical load on multivariate time series," in *Proceedings of the Chinese Society for Electrical Engineering*, pp. 1–5, IEEE, St. Petersburg, Russia, June 2005.
- [8] H. Cui and X. Peng, "Summer short-term load forecasting based on ARIMAX model," *Power System Protection and Control*, vol. 4, pp. 108–114, 2015.
- [9] J. C. López, M. J. Rider, and Q. Wu, "Parsimonious short-term load forecasting for optimal operation planning of electrical distribution systems," *IEEE Transactions on Power Systems*, vol. 34, no. 2, pp. 1427–1437, 2019.
- [10] L. Chen, L. Yin, T. Yu, and K. Wang, "Short-term power load forecasting based on deep forest algorithm," *Electric Power Construction*, vol. 39, no. 11, pp. 42–50, 2019.
- [11] P. Singh and P. Dwivedi, "Integration of new evolutionary approach with artificial neural network for solving short term load forecast problem," *Applied Energy*, vol. 217, pp. 537–549, 2018.
- [12] K. Wang and R. Zhang, "Research on short-term power load forecasting method based on improved BP neural network," *Electrical Measurement & Instrumentation*, vol. 56, no. 24, pp. 115–121, 2020.
- [13] X. Zhang and J. Wang, "A novel decomposition-ensemble model for forecasting short-term load-time series with multiple seasonal patterns," *Applied Soft Computing*, vol. 65, pp. 478–494, 2018.
- [14] Y. Liang, D. Niu, and W.-C. Hong, "Short term load forecasting based on feature extraction and improved general regression neural network model," *Energy*, vol. 166, pp. 653–663, 2019.
- [15] Y. Li, J. Li, and Y. Wang, "Privacy-preserving spatiotemporal scenario generation of renewable energies: a federated deep generative learning approach," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 4, pp. 2310–2320, 2022.
- [16] Y. Li, X. Wei, Y. Li, Z. Dong, and M. Shahidepour, "Detection of false data injection attacks in smart grid: a secure federated deep learning approach," *IEEE Transactions on Smart Grid*, vol. 13, no. 6, pp. 4862–4872, 2022.
- [17] G. Zhang, S. Zhu, and X. Bai, "Federated learning-based multi-energy load forecasting method using CNN-Attention-LSTM model," *Sustainability*, vol. 14, no. 19, Article ID 12843, 2022.
- [18] N. Hudson, M. J. Hossain, and M. Hosseinzadeh, "A framework for edge intelligent smart distribution grids via federated learning, International Conference on Computer Communications and Networks (ICCCN)," *IEEE*, pp. 1–9, 2021.
- [19] Y. Wang, I. L. Bennani, X. Liu, M. Sun, and Y. Zhou, "Electricity consumer characteristics identification: a federated learning approach," *IEEE Transactions on Smart Grid*, vol. 12, no. 4, pp. 3637–3647, 2021.
- [20] J. Lin, J. Ma, and J. Zhu, "Privacy-preserving household characteristic identification with federated learning method,"

IEEE Transactions on Smart Grid, vol. 13, no. 2, pp. 1088–1099, 2022.

- [21] H. Cao, S. Liu, R. Zhao, and X. Xiong, “IFed: a novel federated learning framework for local differential privacy in Power Internet of Things,” *International Journal of Distributed Sensor Networks*, vol. 16, no. 5, Article ID 155014772091969, 2020.
- [22] Z. Su, Y. Wang, T. H. Luan et al., “Secure and efficient federated learning for smart grid with edge-cloud collaboration,” *IEEE Transactions on Industrial Informatics*, vol. 18, no. 2, pp. 1333–1344, 2022.
- [23] A. Taik and S. Cherkaoui, “Electrical load forecasting using edge computing and federated learning,” in *Proceedings of the IEEE International Conference on Communications (ICC)*, pp. 1–6, IEEE, Dublin, Ireland, June 2021.
- [24] J. D. Fernandez, S. P. Menci, C. Lee, and G. Fridgen, “Secure federated learning for residential short-term load forecasting,” 2021, <https://arxiv.org/abs/2111.09248>.