IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received 16 February 2023, accepted 22 March 2023, date of publication 27 March 2023, date of current version 22 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3262171

RESEARCH ARTICLE

Communication-Efficient Federated Learning for Power Load Forecasting in Electric IoTs

ZHENGXIONG MAO[1], HUI LI[1], ZUYUAN HUANG[1], CHUANXU YANG[1], YANAN LI[2], (Member, IEEE), AND ZIHAO ZHOU[3], (Member, IEEE)

[1]Network Information Center of Yunnan State Grid Company, Kunming 650011, China
[2]School of Software, Henan Polytechnic University, Jiaozuo 454003, China
[3]School of Mathematics and Statistics, Xi'an Jiaotong University, Xi'an 710049, China

Corresponding author: Yanan Li (liyn@hpu.edu.cn)

This work was supported by the Yunnan Power Grid Technology Project YNKJXM20210141.

ABSTRACT With the construction of the modern power system, power load forecasting is significant to keep the electric Internet of Things in operation. However, it usually needs to collect massive power load data on the server and may face the problem of privacy leakage of raw data. Federated learning can enhance the privacy of the raw power load data of clients by frequently transmitting model updates. Concerning the increasing communication burden of resource-heterogeneous clients resulting from frequent communication with the server, a communication-efficient federated learning algorithm based on Compressed Model Updates and Lazy uploAd (CMULA-FL) was proposed to reduce the communication cost. CMULA-FL also integrates the error compensation strategy to improve the model utility. First, the compression operator is used to compress the transmitted model updates, of which large norms are uploaded to reduce the communication cost of each epoch and transmission frequency. Second, by measuring the error of compression and lazy upload, the error is accumulated to the next epoch to improve the model utility. Finally, based on simulation experiments on the benchmark power load data, the results show that the communication cost decreases at least 60% with controlled loss of model prediction compared with baseline.

INDEX TERMS Power load forecasting, federated learning, quantization, lazy upload, error compensation.

I. INTRODUCTION

With the rapid increase of electricity demand, the modern power system should have the characteristics of digital enabling, flexible opening and high efficiency [1] better to enable electric Internet of Things (IoTs). Power load forecasting is the backbone of the construction of the modern power system. Accurate prediction of short-term power load data can improve prediction ability of emergencies and guarantee the growing power demand and the reliability of the electric IoTs [2], [3]. Moreover, power load forecasting can significantly improve the economic and social benefits of electric IoTs. Therefore, it is significant to forecast power loads accurately to accelerate the construction of the modern power system.

Recently, data-driven machine learning [4], [5], [6] has been widely used in power load forecasting and achieves marvelous prediction accuracy. Generally, the power grid company (the server) needs to collect massive power load data from enterprises or individuals (the clients), as shown in Fig. 1. However, it will lead to the risk of privacy disclosure in the process of collecting or storing power data [7], [8]. Moreover, with the rising privacy consciousness of people and the improvement of data security law, it is difficult for the power grid company to collect and analyze the power load data from clients [9].

Federated learning (FL) [10], [11], [12], [13], proposed by Google, can effectively protect the raw power load data privacy of all clients. Concretely, in the federated settings, the server coordinates massive clients to train a shared machine-learning model by frequently transmitting model parameters instead of collecting the raw power load data on each client. Therefore, FL significantly ensures the privacy of the raw

The associate editor coordinating the review of this manuscript and approving it for publication was Dongxiao Yu.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

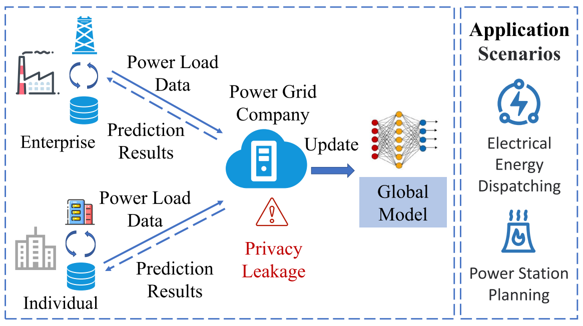


FIGURE 1. The workflow of power load forecasting and application scenarios.

power data of each client and realizes the efficient mining of power load data.

However, frequent communication between the server and clients will result in low communication efficiency [14], [15], [16], [17]. On the one hand, it leads to high communication cost to transmit high-dimension deep learning models [16], [17] and the high communication cost makes clients reluctant to take part in the federated training process. On the other hand, it is time-intensive to upload high-dimension models from clients to the server frequently [14], resulting in communication bottleneck in power grid data centers. Therefore, communication cost has become one of the most important bottlenecks of FL to ensure the privacy of clients' raw data.

To reduce the communication cost, the existing methods mainly consider the communication frequency and the communication cost of each epoch, such as periodic averaging [7], [18], lazy upload [17], gradient or model compression [14], [15], [16], sampling [19]. From the perspective of communication frequency, McMahan et al. [7] adopted the method of periodic averaging to reduce the communication cost. From the perspective of the communication cost in each epoch, Bernstein et al. [14] proposed SignSGD to compress the gradient, which significantly reduced the communication cost in the uploading stage of the model. Compression operators [15], [16] can also reduce the communication cost of each epoch by uploading compressed gradients with controllable error. To bring down the communication cost comprehensively, Sun et al. [17] proposed LAQ by combining lazy upload and model updating quantization to reduce the upload frequency and transmission cost in each epoch.

However, there are still three core problems. *Firstly*, most existing methods are not systematic and only reduce the communication cost from one dimension. *Secondly*, the existing methods ignore the high communication cost in the downloading stage of the model. *Finally*, both model compression and lazy upload strategies will generate a large bias, which significantly reduces the model utility and even diverges the model.

To deal with the above three problems and improve communication efficiency, we proposed a communication-efficient power load forecasting algorithm (CMULA-FL) based on model bidirectional compression and lazy upload. Moreover, CMULA-FL utilizes the error compensation strategy to diminish the impacts of generated bias. Compared with the existing power load forecasting algorithm, we summarize our main contributions below:

1) We propose CMULA-FL, a communication-efficient FL algorithm based on bidirectional compression and lazy upload, to reduce the communication cost systematically. The proposed CMULA-FL reduces the communication cost from the orthometric aspects of the transmission cost of each epoch and the transmission frequency of the model update simultaneously.
2) We propose an error compensation strategy to deal with the reduction of the model utility caused by biases from model update compression and lazy upload. The error compensation strategy improves the model utility by accumulating the model update errors of the last epoch to the next epoch.
3) We deployed massive experiments on the power load dataset to verify the efficiency of CMULA-FL. The experimental results validate that bidirectional compression and lazy upload can greatly reduce the communication cost, and the fusion with error compensation strategy can ensure the model utility.

The remaining parts of this paper are organized as follows. We first investigate the existing work related to power load forecasting and the communication-efficient algorithm in the federated scenarios in Section II. Then, the background knowledge involved in this paper is briefly introduced in Section III. The proposed CMULA-FL algorithm is introduced in detail in Section IV. In Section V, the effectiveness of the CMULA-FL algorithm and the influence of some important parameters are verified by deploying experiments and analyzing experimental results in detail. Finally, we present the conclusion in Section VI.

II. RELATED WORK

A. POWER LOAD FORECASTING

There are various methods to forecast power load data, such as mathematical statistics theory [20], [21] and deep learning model [22]. Jeong et al. [20] put forward the logistic mixture vector autoregressive model based on curve registration to predict the short-term power load of buildings. By integrating clustering and prediction algorithms through the expectation-maximization algorithm, it significantly achieved a better prediction effect. Similarly, Savari et a. [21] proposed a real-time load prediction algorithm for electric vehicles, which fulfilled the function of power dispatching management based on charging piles recommended by the state of charge. Muzaffar and Afshari [22] proposed a LSTM-based model, which has the capabilities to mine the long-term dependencies and extract useful information from power load data.

However, those centralized scenarios to forecast power load requires massive power load data from clients, and there exists a potential risk of privacy disclosure in the collection and storage process [7]. Coupled with the increase of people's privacy awareness and data security laws, data barriers are gradually forming [9]. Therefore, the fusion with FL can not only protect the privacy of the original power load data, but also coordinate amounts of clients to train the shared model.

B. FL-BASED POWER LOAD FORECASTING

In order to ensure the privacy and security of clients' raw data, FL has been widely applied in the field of power load forecasting and has achieved remarkable results. Taïk et al. [23] applied FL to short-term forecasting to protect clients' data privacy and obtained desired effectiveness. To address the problem that the load variation patterns in each region cannot be accurately grasped in time due to high communication delay, Li et al. [24] proposed an ultra FL algorithm for short-term power load forecasting. Besides, He et al. [25] adopted the federated K-means clustering algorithm to divide clients into separate clusters based on historical power load data, and learns local personalized models within each cluster to enhance the privacy security of local data. The horizontal federated LSTM algorithm [23], [26] was employed to predict the power demand of each client to enhance security.

However, in the FL scenarios, each client needs to communicate frequently with the grid power data center, incurring high communication cost that significantly increase the burden on resource heterogeneous clients [14], [15], [16], [17]. Therefore, it is crucial to design a communication-efficient power load forecasting algorithm.

C. COMMUNICATION-EFFICIENT FL

For the purpose of improving communication efficiency, the existing methods mainly consider reducing the communication frequency and communication cost of each epoch. Saputra et al. [27] proposed model sharing to reduce the communication frequency, significantly bringing down the communication overhead while safeguarding the privacy of clients' raw data. From the perspective of the high communication overhead in federated power load forecasting, Fekri et al. [28] adopted periodic averaging to reduce the communication frequency and guarantee the forecasting accuracy in resource-constrained federated scenarios. Gholizadeh and Musilek [29] found that there were a few abnormal nodes in FL training, which reduced the convergence speed. Accordingly, the abnormal node detection strategy was introduced, and the abnormal nodes were removed to ameliorate the convergence speed and reduce the communication overhead. In [14], the SignSGD method was adopted to compress the gradient in the uploading phase, which significantly reduced the communication cost of each epoch.

However, the approaches still face three significant problems. Firstly, the communication cost is determined by communication frequency and the communication cost in each epoch, so the existing methods lack comprehensiveness and systematicity. Secondly, the existing methods do not alleviate the communication cost in the transmission stage of the model, or only consider the cost in the uploading stage, ignoring the high communication cost in the downloading stage of the model. Finally, reducing the communication cost poses large errors and significantly reduces the model utility. To cope with these three problems, a FL-based power load forecasting algorithm based on lazy upload and bidirectional model quantization is proposed from the two dimensions of communication frequency and communication cost of each epoch. Apart from this, the error compensation strategy is utilized to improve the model utility.

III. PRELIMINARY

This section presents the optimization method and the basic framework of the proposed algorithm. First, Section III-A introduces the stochastic gradient descent method. Then, Section III-B describes the workflow of FL and serves it as the basic framework of the proposed algorithm.

A. STOCHASTIC OPTIMIZATION

In general, the objective of machine learning is to solve the following optimization problem.

$$\min_w F(w) = \frac{1}{N}\sum_{\xi_i \in \mathcal{D}} f(w, \xi_i), \tag{1}$$

where $\mathcal{D}$ and $f(w, \xi_i)$ represent the training set and the i-th loss respectively. The mini-batch stochastic gradient descent method can perform as a solver for this optimization problem and reduce the consumption of computing resources at the same time. The iterative updating rule is as follows.

$$w_{j+1} = w_j - \frac{\eta_j}{|\mathcal{D}_j|}\sum_{\xi_{j,i} \in D_j} \nabla f(w_j, \xi_{j,i}), \tag{2}$$

where $\mathcal{D}_j \subseteq \mathcal{D}$ is the mini-batch data in the j-th iteration, and η_j is the learning rate.

B. FEDERATED LEARNING

In order to ensure the clients' raw data privacy, Google proposed the FL algorithm [7]. We take a FL scenario with a central server and K clients as an example to describe the system model of FL in detail. Before federated training, the data of multiple clients are not exactly overlapping or mutually exclusive. Therefore, overlapping data needs to be filtered out to ensure fairness. First, the server uses the encryption algorithm to align the data of clients and confirms the common data among clients without disclosing the raw data. Then, based on the requirements in the application scenarios, a specific portion of the clients are screened for modeling. Once the target clients are identified, the FL training process begins. The training process of FL mainly is composed of the following four steps.

1) The server initializes some basic parameters, such as the model parameters, learning rate and so on, then broadcasts them to each client.
2) All clients train the received model separately on local data sets and calculate gradients or model parameters as intermediate results.

3) Clients meeting the upload criteria send intermediate results to the server.
4) The server receives and aggregates the gradients or model parameters. Then, the server broadcasts the updated model parameters individually to all clients.

Iterate steps 2-4 until the test loss value is less than the set threshold or a preset number of iterations is reached.

During the FL training process, the raw data for each clients is always stored locally, which protects the privacy and security of the raw data. Therefore, FL greatly reduces the risk of privacy disclosure of the raw data.

IV. CMULA-FL

To lower the communication cost of FL-based power load forecasting, we proposed a communication-efficient FL algorithm based on Compressed Model Updates and Lazy uploAd (CMULA-FL). CMULA-FL can not only reduce the communication cost effectively, but also couple the error compensation strategy to ensure the model utility. The basic idea of CMULA-FL is described in Section IV-A. Based on these ideas, the framework and details of CMULA-FL algorithm are presented in Section IV-B.

A. MAIN IDEA

The communication cost of the whole federated training process is mainly determined by the transmission bits in one epoch and transmission frequency. Therefore, the proposed CMULA-FL algorithm reduces the communication cost from these two orthometric dimensions. From the perspective of the communication cost in one epoch, model update of each client will be compressed before transmitting to the server. Similarly, the aggregated model update on the server will also be compressed before broadcasting to the clients. From the aspect of communication frequency, the lazy upload strategy is adopted by only selecting part clients to upload model updates, because some clients have inconspicuous changes in model updates and make fewer contributions to the process of aggregation. Therefore, The client can upload the model update until the accumulated model changes are greater than a threshold.

However, it will incur an obvious bias, which can reduce the model utility. The bias comes from compressed transmission parameters and lazy upload. With the increasing of compressed information and the lazy upload threshold, the bias will significantly reduce the model utility and even diverge the model. In order to ensure the model effectiveness, the proposed CMULA-FL algorithm adopts the error compensation to reduce the influence of errors by accumulating the errors generated in the current epoch to the next epoch.

B. DETAILED DESCRIPTION

Based on the above basic ideas, an efficient CMULA-FL algorithm is proposed in this paper to forecast power load data safely and efficiently. The pseudo-code of CMULA-FL algorithm is mainly composed of two parts, including the server side (Algorithm 1) and the client side (Algorithm 2). Fig. 2 shown the workflow of the proposed CMULA-FL. It mainly consists of three parts, namely the quantization of model update (③ and ⑧) and error compensation (④ and ⑦) on the server and the clients, and lazy upload (⑤) on the clients.

Algorithm 1 CMULA-FL: Server Side

```
Input: Compression operator $Q(\cdot)$, number of all clients $K$.
Output: Optimal model parameter $w^*$.
1: Initialize the model parameters, model update, and global error.
2: Quantize the model update and compute the error of quantization.
3: Broadcast the quantized model update.
4: while the preset stopping condition is not achieved do
5:     for $i = 1 \to K$ do
6:         The $i$-th client computes and uploads the quantized model update.
7:     end for
8:     Receive and aggregate the model updates.
9:     Accumulate the error of $j-1$-th epoch.
10:    Quantize the model update and compute the quantization error.
11:    Update the global model and broadcast the quantized model update.
12: end while
```

On the server side (⑥-⑧), as shown in Algorithm 1, the server first initializes parameters such as model and model updates (line 1). Then, the server quantizes model updates and broadcasts them to all clients (lines 2-3). After all the satisfied clients upload their model updates (lines 5-7), the server aggregates all received quantized model updates (line 8) and accumulates the error of the last epoch (line 9). In order to reduce communication costs, the server quantizes the aggregated updates and calculates quantized errors (line 10). Finally, the shared model is updated and the quantized model update is broadcast to clients (line 11). The server repeats the processes (lines 4-12) until the stopping criteria is satisfied.

On the client side (① - ⑤), as shown in Algorithm 2, each client first receives the model update and updates the local model (line 1). Then, each client updates the received model on local data for E iterations (line 2). After local iteration, the variation of local model is updated. To guarantee the model utility, each client' local model update is accumulated with the error of the last epoch (line 3). The corrected model update is quantized to reduce the communication cost (line 4). Finally, the lazy upload strategy is adopted to further reduce the communication cost (lines 5-11). That is, when the model update reaches the preset threshold ϵ or the client has not uploaded the model update for t_{max} epoch, clients will update the quantization error and upload the local quantized model update (line 6). If the upload conditions are not satisfied, the lazy upload error is updated (line 10).

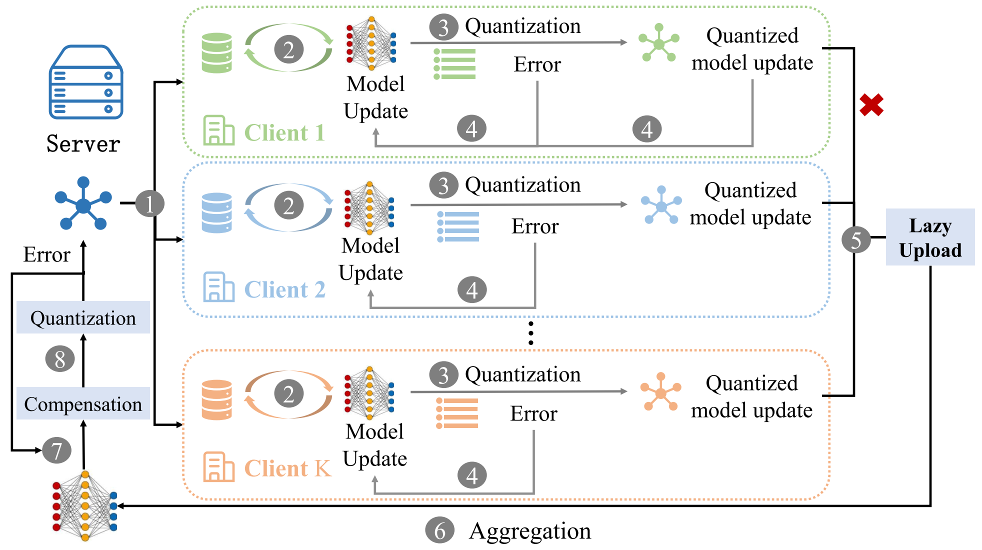


FIGURE 2. Framework of CMULA-FL. CMULA-FL algorithm contains three main parts: model update quantization (③, ⑧), error compensation (④, ⑦) on both clients and the server, and lazy upload (⑤) on the clients.

Algorithm 2 CMULA-FL: Client Side

```
Input: learning rate $\eta$, lazy upload parameter $\epsilon$.
Output: Quantized model update $\Delta\tilde{w}_{i,j}$.
 1: Receive the model update and update local model parameter $w_{i,j}^0$.
 2: Train the model on local data for $E$ iterations.
 3: Compute the model update and accumulate the error $err_{i,j-1}$.
 4: Quantize the model update $\Delta\tilde{w}_{i,j} = Q(\Delta\bar{w}_{i,j})$.
 5: if $||\Delta\tilde{w}_{i,j}|| \geq \epsilon$ or $t_i \geq t_{max}$ then
 6:     Compute the quantization error $err_{i,j}$ and upload the model update $\Delta\tilde{w}_{i,j}$.
 7:     $t_i \leftarrow 1$;
 8: else
 9:     $t_i \leftarrow t_i + 1$;
10:     Update the lazy upload error $err_{i,j}$.
11: end if
```

In the remaining parts, we will present the three core parts of CMULA-FL in Sections IV-B1-IV-B3 and model aggregation in Section IV-B4.

1) MODEL BIDIRECTIONAL QUANTIZATION

Transmission of complete model parameters requires a large amount of communication cost, which can be significantly reduced by model compression. In this paper, the transmission models of the uploading and downloading stages are compressed by the γ-compression operator. The γ-compression operator is defined as follows

Definition 1 (γ-Compression Operator): If the compression operator Q satisfies

$$||x - Q(x)||^2 \leq (1-\gamma)||x||^2, \tag{3}$$

then the compression operator Q is defined as γ-compression operator.

Common compression operators include Top-k sparsity operator [16] and 1-bit quantization [30] and so on. However, compared with these compression operators, the multi-bits quantization method can adjust the quantization bits flexibly and control the precision of transmitted model parameters to realize the controllable error. Therefore, we adopt the multi-bits quantization.

By utilizing the γ-compression operator to compress the model update of the clients and server, CMULA-FL can effectively reduce the communication cost effectively in the downloading and uploading process with controllable error. The compression rules for the clients and the server are as follows:

$$\Delta\tilde{w}_{i,j} = Q(\Delta\bar{w}_{i,j}),$$
$$\Delta\tilde{w}_j = Q(\Delta w_j), \tag{4}$$

where $\Delta\tilde{w}_{i,j}$ and $\Delta\tilde{w}_j$ are the model updates of the i-th client and server after quantization respectively.

2) LAZY UPLOAD

If the change of a model update is small, it has few impacts on the aggregated model. Therefore, skipping this redundant gradient can not only reduce the communication cost of the client, but also have few impacts on the aggregated model. Before model training, the server initializes a small norm threshold ϵ. When the norm of the model update is smaller than the set threshold ϵ, the model update can be skipped.

If the model update obtained by the client continues to be small, the client will make little contribution to the federated process. When the client satisfies the uploading criteria, the accumulated model update will have a large staleness, which will result in a large model deviation. Therefore, it is necessary to ensure the freshness of the model updates. Then, we can set a maximum number of local iterations to t_{max} empirically. If the norm of the client's model update continues to be less than the set threshold ϵ for t_{max} epochs, the quantized model update will be uploaded to control the staleness of the model update. By setting a smaller ϵ and t_{max}, the model updates will be fresher and model utility can be improved.

3) ERROR COMPENSATION

After the compressed quantization model is updated, there is a large quantization error between the original model update and the quantization results. With the increasing epochs, the cumulative effects will significantly reduce the model utility and even diverge the model. To improve the model utility, it is indispensable to adopt the method of error compensation for both clients and the server to decrease the impacts of compression error.

For each client, after training the model for E times locally, the error of the last epoch can be accumulated to the current model change to alleviate the impacts of the quantization error.

$$\begin{aligned} err_{i,j-1} &= \Delta\tilde{w}_{i,j-1} - Q(\Delta\bar{w}_{i,j-1}), \\ \Delta\bar{w}_{i,j} &= \Delta w_{i,j} + err_{i,j-1}, \end{aligned} \tag{5}$$

where $err_{i,j-1}$ represents the quantization error generated by the i-th client in the $j-1$-th epoch.

For the server, the local model updates of clients are aggregated first, and then the results are added to the errors of the previous epoch to decrease the impacts of compressed model update,

$$\begin{aligned} err_{j-1} &= \Delta\tilde{w}_{j-1} - Q(\Delta w_{j-1}), \\ \Delta w_j &= Agg(\Delta\tilde{w}_{i,j}) + err_{j-1}, \end{aligned} \tag{6}$$

where err_{j-1} represents the quantization error generated by the server in the $j-1$-th epoch.

Apart from the quantization error, the lazy upload can also generate a large bias. If the model updates of one client continue to be small and are abandoned directly, it will incur two problems. On the one hand, frequent skipping of model updates leads to extravagant computing resources. On the other hand, the skipped gradients make the aggregated model update be biased and decrease the model utility. Therefore, when the client's local model update carries little information, the model update can be accumulated to the next epoch with the quantization error. When the client is skipped, the error is given as

$$err_{i,j} = w_{i,j}^{E} - w_{i,j}^{0} + err_{i,j-1}. \tag{7}$$

where $w_{i,j}^{0}$ and $w_{i,j}^{E}$ are the models of i-th client at the beginning and end of the local training process respectively.

4) MODEL AGGREGATION

After the model updates on all satisfied clients reach the server side, the server aggregates the model updates and accumulates the error err_{j-1}. The aggregation and accumulation rules are as follows

$$\Delta w_j = \frac{1}{S_j}\sum_{i\in S_j}\Delta\tilde{w}_{i,j} + err_{j-1}, \tag{8}$$

where S_j defines the set of clients who upload the model updates.

V. EXPERIMENTS

To validate the performance of the proposed algorithm CMULA-FL, we deployed CMULA-FL under the PySyft framework [31]. The performance is measured from the perspective of training speed and communication efficiency. Section V-A presents the experimental setup, such as three existing comparison algorithms and power load datasets. Then, detailed experimental results are shown and analyzed to demonstrate the advantages of CMULA-FL in Section V-B.

A. EXPERIMENTAL SETUP

1) DATASETS AND MODELS

In the experiments, we made use of a benchmark power load dataset from the Chinese Society of Power Engineering,[1] which records the power load data of a region from 2009 to 2015. The power load data can be converted into a matrix $\mathbf{X}$. However, due to the fact of improper operation and other human factors in the process of collecting data, there exist some bad data, bringing down the model utility. Therefore, it is significant to preprocess the dataset before the training model. The data is preprocessed in three main steps, namely data standardization, generation of time-series data, generation of training data and test data. The three steps will be introduced in detail as follows.

a: DATA STANDARDIZATION

The power load data has the characteristic of periodicity. Therefore, the power load data is roughly similar between two adjacent periods. In this paper, we used some basic statistical metrics and an empirical threshold to judge the similarity of data in two adjacent periods and identify bad data. The details are as follows.

- Divide the power load data into different groups (N groups) in days;
- Compute the mean value $\bar{X}_q$ and variance σ_q^2 of q-th group ($q = 1, \ldots, N$);
- Pick out the bad data based on an empirical threshold $\epsilon_q \in [1, 1.5]$. The n-th data $X_{n,q}$ in group q is divided into a bad data if it satisfies

$$|X_{n,q} - \bar{X}_q| > 3\sigma_q\epsilon_q.$$

- Correct the bad data based on adjacent data,

$$X_{n,q} = (X_{n-1,q} + X_{n+1,q})/2.$$

- Normalize the power load data.

[1]http://shumo.nedu.edu.cn.

TABLE 1. Experimental parameter settings.

Parameters	Meaning	Value
hidden_size	Number of hidden nodes	128
num_layers	Number of LSTM layers	2
η	Learning rate	0.001
P	Total number of clients	15
batch_size	Batch size of data	50
K	Number of selected clients	7

b: GENERATION OF TIME-SERIES DATA

In order to deploy LSTM to train power load data, it is significant to generate time-series data and labels. In order to ensure the model utility, we generate time-series data through a sliding window, which is set to a larger number of 12.

c: GENERATION OF TRAINING DATA AND TEST DATA

The training set and test set are obtained by dividing the generated time-series data in a ratio of 4:1. Then, the training power load data was randomly assigned to each client, and each client keeps a similar amount of data.

d: MODEL

For the sake of ensuring the model utility of the power load dataset, the neural network LSTM is adopted in this paper. The LSTM consists of two layers, each of which contains an input gate, a forgetting gate and an output gate. The outputs of the first layer are the input parameters for the input gate in the second layer. The activation function for each layer set as the Softmax function.

e: PARAMETER SETTINGS

In our experiments, to keep the model updates fresh, t_{max} is set to 10. Other default parameters are as listed in Table 1.

2) COMPARISON ALGORITHMS

The proposed CMULA-FL was compared with two communication-efficient algorithms in our experiments, including a Lazily Aggregated Quantized gradient approach (LAQ, [17]) and FL based on Periodic Averaging and Quantization (FedPAQ, [32]). FedAvg [7], one of the most classical FL algorithms, is set as the baseline. A brief introduction of the two comparison algorithms is as follows.

a: LAQ

To improve communication efficiency, LAQ quantizes gradients as well as skips some quantized gradients with less information by utilizing previous gradients, which can simultaneously save communication bits and rounds without sacrificing the desired convergence guarantees.

b: FedPAQ

To reduce the communication cost, FedPAQ adopts the low-precision quantizer to decrease the communication cost of one epoch. Apart from that, FedPAQ utilizes partial client participation and periodic averaging to reduce the communication frequency.

3) METRICS

We compared CMULA-FL with the above algorithms from the aspects of training speed and communication efficiency. The training speed is directly measured by the number of iterations before convergence which is judged based on training loss. Communication efficiency is measured by the transmission bits during the training process. We describe the metrics of training loss and transmission bits as follows.

a: TRAINING LOSS

We use Mean Squared Error (MSE) as the metrics of the performance on the test data. It reflects the magnitude of the difference between the true label and the predicted value of the model and is calculated as

$$MSE = \frac{1}{m}\sum_{i=1}^{m}(y_i - f(x_i))^2, \tag{9}$$

where m is the number of test data, y_i and $f(x_i)$ are the true and predicted value respectively. MSE can reflect two aspects of model performance. On the one hand, the declining rate of MSE on the test dataset reflects the convergence speed. On the other hand, a smaller MSE means that the corresponding algorithm predicts more accurately.

b: COMMUNICATION COST

The communication cost is measured by the bits of transmission parameters uploaded by all clients and the rule is given as

$$\mathcal{C}_{tot} = \sum_{j=1}^{J}\sum_{i\in S_j}\sum_{k=1}^{M_{num}}\mathcal{C}(\Delta\widetilde{w}_{i,j,k}), \tag{10}$$

where $\mathcal{C}(\Delta\widetilde{w}_{i,j,k})$ defines the bits of $\Delta\widetilde{w}_{i,j,k}$, the k-th model parameter of model $\Delta\widetilde{w}_{i,j}$.

B. EXPERIMENTAL RESULTS AND ANALYSIS

To validate the performance of CMULA-FL and explore the impacts of some important parameters on the model utility, four groups of experiments were conducted from two aspects: the communication cost and the loss value. The specific settings of each group are as follows:

1) The first group counted up the communication cost of the model in the training phase under different algorithms and quantization bits.
2) In the second group, CMULA-FL was compared with the baseline and two comparison algorithms to verify the model utility.
3) In the third group, some ablation experiments and comparison experiments were presented to verify the function of some components of CMULA-FL. In the ablation experiments, the error compensation strategy was removed from CMULA-FL to explore the impacts of error compensation and we rename it CMULA-FL

TABLE 2. Communication cost of different algorithms under different quantization bits.

Algorithm	Quantization Bits	Communication Cost
FedAvg [7]	–	2.62×10^7
FedPAQ [32]	–	2.17×10^7
LAQ [17]	4	1.58×10^7
LAQ	8	1.66×10^7
LAQ	16	1.79×10^7
CMULA-FL	4	$\mathbf{4.38 \times 10^6}$
CMULA-FL	8	$\mathbf{7.30 \times 10^6}$
CMULA-FL	16	$\mathbf{1.02 \times 10^7}$

(No Compensation). In the comparison experiments, CMULA-FL was compared with FedAvg and the centralized settings with only one client (One-Client).

4) In the fourth group, various quantization bits were set to explore the influence of quantization on the model utility.

1) IMPACTS OF QUANTIZATION BITS ON COMMUNICATION COST

In order to explore the communication cost for different quantization bits and algorithms, we dynamically set the quantization bits for CMULA-FL and varied the algorithms to verify the effectiveness of CMULA-FL. To ensure fairness, the number of training epochs was 100 and Table 1 shows the settings of other parameters. Table 2 shows the communication cost for different algorithms and different quantization bits.

As can be seen from Table 2, the larger quantization bits will result in a larger communication cost for LAQ and CMULA-FL, which indicates that the bits of transmission parameters are positively correlated with the communication cost. However, the relation is not linear due to the lazy upload strategy. Table 2 also shows that CMULA-FL can significantly reduce the communication cost compared with the baseline, FedAvg. Compared with LAQ and FedPAQ, the proposed CMULA-FL demonstrates a superior performance in terms of communication cost since CMULA-FL compresses the bidirectional transmission parameters. Apart from that, the communication cost of CMULA-FL decreases more rapidly with the diminution of quantization bits than those of LAQ due to the fact that LAQ only compresses the model during the uploading stages, whereas CMULA-FL also reduces the communication cost in the downloading stages.

To verify the model utility, CMULA-FL was compared with the baseline, FedAvg and two comparison algorithms. All algorithms adopted the same parameters as shown in Table 1 and the quantization bits for CMULA-FL and LAQ were set to 8. Fig. 3 shows the relation between MSE and epochs for different algorithms.

As shown in Fig. 3, FedAvg has the smallest MSE value and the MSE of CMULA-FL is a little larger within acceptable ranges. However, LAQ and FedPAQ have larger biases and fluctuate heavily since LAQ uses stale gradients while

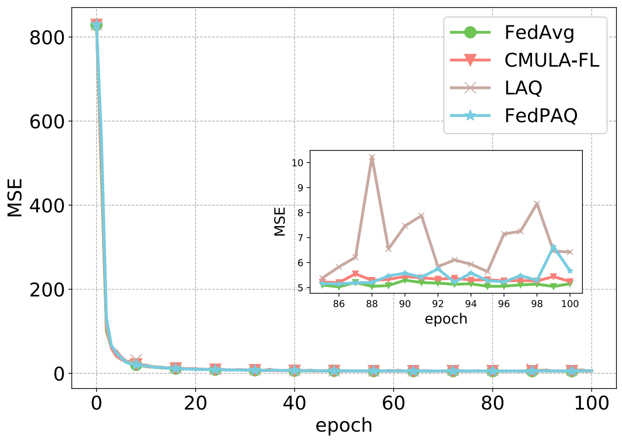


FIGURE 3. Relation between MSE and epochs for different algorithms.

FedPAQ ignores the error caused by model compression. It shows that CMULA-FL can reduce the communication cost without sacrificing much prediction accuracy by utilizing the error compensation strategy.

By combining the analysis of Table 2, Fig. 3 also shows that CMULA-FL can not only reduce the communication cost effectively by compressing bidirectional model update and uploading model update lazily, but also guarantee the model utility by utilizing error compensation strategy.

2) IMPACTS OF DIFFERENT COMPONENTS ON MSE

We deployed some ablation experiments and comparison experiments to verify the function of some components of CMULA-FL. To ensure the fairness of the comparison, the initialized model parameters were set to the same values in different scenarios and the number of epochs was 100. Table 1 shows specific parameter settings. Fig. 4 shows the relation between MSE and epochs under different settings.

As shown in Fig. 4, FedAvg converges the fastest, while CMULA-FL has a slightly lower convergence speed. However, compared with CMULA-FL-NoCom, the performance of CMULA-FL has been significantly improved, which shows that compression and lazy upload will significantly reduce the model utility and the impacts can be decreased by error compensation strategy.

Fig. 4 also shows that the model trained with a single client converges the slowest. It is due to the fact that all clients can jointly train the model and more data can significantly improve the model generalization. The FL scenarios are similar to aggregating all the data from clients, which enhances the data volume and thus improves the convergence speed.

3) IMPACTS OF QUANTIZATION BITS ON MSE

To explore the impacts of different quantization bits on CMULA-FL, we dynamically set quantization bits to observe the changes of MSE. To ensure fairness, the number of training epochs was 100, and Table 1 shows the settings of other parameters. Fig. 5 shows the trends of MSE corresponding to different quantization bits.

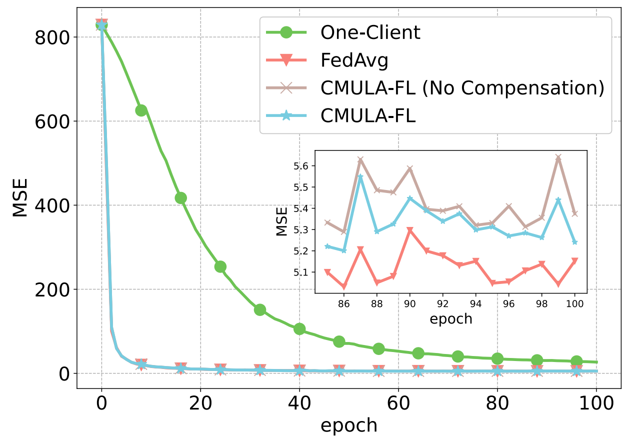


FIGURE 4. **Relation between MSE and epochs under different settings (the centralized settings with one client, FedAvg, CMULA-FL without error compensation strategy (No Compensation) and CMULA-FL).**

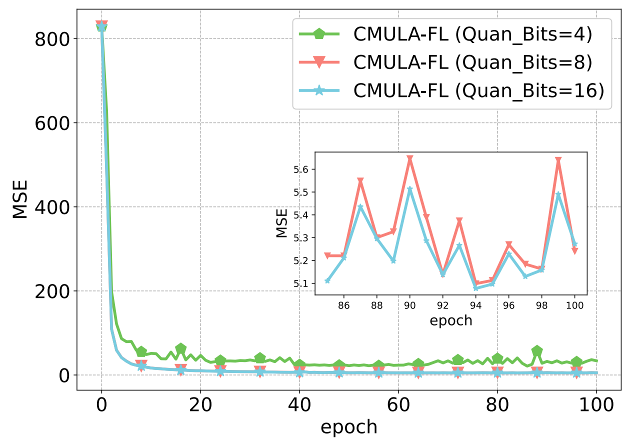


FIGURE 5. **Relation between MSE and epochs under different quantization bits (Quan_Bits) for CMULA-FL.**

As shown in Fig. 5, a larger number of quantization bits results in converging faster. If the quantization bits are set larger, the transmission model will be more accurate. Therefore, the model will be updated in a righter direction and converge faster. It shows that on the premise of sufficient communication resources, the larger quantization bits can lead to a higher prediction accuracy of the model.

VI. CONCLUSION

We propose communication-efficient CMULA-FL to forecast power load data on enterprises or individuals efficiently and safely to better enable electric IoTs. Based on bidirectional quantization and lazy upload, CMULA-FL effectively reduces the communication cost of the federated prediction process. Apart from that, CMULA-FL also utilizes the error compensation strategy to ensure prediction accuracy and convergence speed. *Firstly*, the model updates of the clients and server are quantized to bring down the communication cost. CMULA-FL also adopts lazy upload to further improve communication efficiency. *Secondly*, the deep fusion with error compensation strategy can effectively reduce the impacts caused by quantization errors and lazy upload errors. Finally, the model effectiveness of CMULA-FL is verified by deploying massive experiments in different scenes and parameter settings. However, the proposed CMULA-FL algorithm is still imperfect and needs to be improved. For example, CMULA-FL has many hyper-parameters and it is difficult to set hyper-parameters based on experiences. In the subsequent work, the influence of hyper-parameters will be analyzed theoretically, such as the local update change threshold and time threshold. By analyzing the impacts of different parameters, it will be more convenient to deploy CMULA-FL.

REFERENCES

[1] Y. Saleem, N. Crespi, M. H. Rehmani, and R. Copeland, "Internet of Things-aided smart grid: Technologies, architectures, applications, prototypes, and future research directions," *IEEE Access*, vol. 7, pp. 62962–63003, 2019.

[2] K. Chen, K. Chen, Q. Wang, Z. He, J. Hu, and J. He, "Short-term load forecasting with deep residual networks," *IEEE Trans. Smart Grid*, vol. 10, no. 4, pp. 3943–3952, Jul. 2018.

[3] A. Prakash, A. Kumar, A. Kaushal, K. Namrata, and N. Kumar, "Load forecasting and analysis of power scenario in bihar using time series prediction and machine learning," in *Smart Energy and Advancement in Power Technologies*. Singapore: Springer, 2023, pp. 851–860.

[4] B. A. S. Oliveira, A. P. De Faria Neto, R. M. A. Fernandino, R. F. Carvalho, A. L. Fernandes, and F. G. Guimaraes, "Automated monitoring of construction sites of electric power substations using deep learning," *IEEE Access*, vol. 9, pp. 19195–19207, 2021.

[5] Z. Chen, J. Xiang, P.-O. Bagnaninchi, and Y. Yang, "MMV-Net: A multiple measurement vector network for multifrequency electrical impedance tomography," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Mar. 9, 2022, doi: 10.1109/TNNLS.2022.3154108.

[6] Q. Wang, H. Zhang, X. Li, X. Duan, J. Wang, R. Zhang, H. Zhang, Y. Ma, H. Wang, and J. Jia, "Error-constraint deep learning scheme for electrical impedance tomography (EIT)," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–11, 2022.

[7] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. Cambridge, MA, USA: MIT Press, 2017, pp. 1273–1282.

[8] X. Ren, C.-M. Yu, W. Yu, S. Yang, X. Yang, J. A. McCann, and S. Y. Philip, "LoPub: high-dimensional crowdsourced data publication with local differential privacy," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 9, pp. 2151–2166, Sep. 2018.

[9] Q. Yang, Y. Liu, Y. Cheng, Y. Kang, T. Chen, and H. Yu, "Federated learning," *Synth. Lectures Artif. Intell. Mach. Learn.*, vol. 13, no. 3, pp. 1–207, 2019.

[10] Z. Zhou, Y. Li, X. Ren, and S. Yang, "Towards efficient and stable K-asynchronous federated learning with unbounded stale gradients on non-IID data," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 12, pp. 3291–3305, Dec. 2022.

[11] C. Zhao, X. Sun, S. Yang, X. Ren, P. Zhao, and J. McCann, "Exploration across small silos: Federated few-shot learning on network edge," *IEEE Netw.*, vol. 36, no. 1, pp. 159–165, Jan. 2022.

[12] X. Sun, S. Yang, and C. Zhao, "Lightweight industrial image classifier based on federated few-shot learning," *IEEE Trans. Ind. Informat.*, early access, Sep. 29, 2022, doi: 10.1109/TII.2022.3210600.

[13] C. Wu, F. Wu, L. Lyu, Y. Huang, and X. Xie, "Communication-efficient federated learning via knowledge distillation," *Nature Commun.*, vol. 13, no. 1, pp. 1–8, 2022.

[14] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar, "signSGD: Compressed optimisation for non-convex problems," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 560–569.

[15] Y. Lin, S. Han, H. Mao, Y. Wang, and W. J. Dally, "Deep gradient compression: Reducing the communication bandwidth for distributed training," 2017, *arXiv:1712.01887*.

[16] S. U. Stich, J.-B. Cordonnier, and M. Jaggi, "Sparsified SGD with memory," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018, pp. 1–12.

[17] J. Sun, T. Chen, G. B. Giannakis, Q. Yang, and Z. Yang, "Lazily aggregated quantized gradient innovation for communication-efficient federated learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 4, pp. 2031–2044, Apr. 2020.

[18] F. Haddadpour, M. M. Kamani, A. Mokhtari, and M. Mahdavi, "Federated learning with compression: Unified analysis and sharp guarantees," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2021, pp. 2350–2358.

[19] Q. Han, S. Yang, X. Ren, P. Zhao, C. Zhao, and Y. Wang, "PCFed: Privacy-enhanced and communication-efficient federated learning for industrial IoTs," *IEEE Trans. Ind. Informat.*, vol. 18, no. 9, pp. 6181–6191, Sep. 2022.

[20] D. Jeong, C. Park, and Y. M. Ko, "Short-term electric load forecasting for buildings using logistic mixture vector autoregressive model with curve registration," *Appl. Energy*, vol. 282, Jan. 2021, Art. no. 116249.

[21] G. F. Savari, V. Krishnasamy, J. Sathik, Z. M. Ali, and S. H. E. A. Aleem, "Internet of Things based real-time electric vehicle load forecasting and charging station recommendation," *ISA Trans.*, vol. 97, pp. 431–447, Feb. 2020.

[22] S. Muzaffar and A. Afshari, "Short-term load forecasts using LSTM networks," *Energy Proc.*, vol. 158, pp. 2922–2927, Feb. 2019.

[23] A. Taik and S. Cherkaoui, "Electrical load forecasting using edge computing and federated learning," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2020, pp. 1–6.

[24] J. Li, Y. Ren, S. Fang, K. Li, and M. Sun, "Federated learning-based ultra-short term load forecasting in power Internet of Things," in *Proc. IEEE Int. Conf. Energy Internet (ICEI)*, Aug. 2020, pp. 63–68.

[25] Y. He, F. Luo, G. Ranzi, and W. Kong, "Short-term residential load forecasting based on federated learning and load clustering," in *Proc. IEEE Int. Conf. Commun., Control, Comput. Technol. Smart Grids (SmartGridComm)*, Oct. 2021, pp. 77–82.

[26] M. Savi and F. Olivadese, "Short-term energy consumption forecasting at the edge: A federated learning approach," *IEEE Access*, vol. 9, pp. 95949–95969, 2021.

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

[28] M. N. Fekri, K. Grolinger, and S. Mir, "Distributed load forecasting using smart meter data: Federated learning with recurrent neural networks," *Int. J. Electr. Power Energy Syst.*, vol. 137, May 2022, Art. no. 107669.

[29] N. Gholizadeh and P. Musilek, "Federated learning with hyperparameter-based clustering for electrical load forecasting," *Internet Things*, vol. 17, Mar. 2022, Art. no. 100470.

[30] M. Schlüter, M. Dörpinghaus, and G. P. Fettweis, "Bounds on phase, frequency, and timing synchronization in fully digital receivers with 1-bit quantization and oversampling," *IEEE Trans. Commun.*, vol. 68, no. 10, pp. 6499–6513, Oct. 2020.

[31] T. Ryffel, A. Trask, M. Dahl, B. Wagner, J. Mancuso, D. Rueckert, and J. Passerat-Palmbach, "A generic framework for privacy preserving deep learning," 2018, *arXiv:1811.04017*.

[32] A. Reisizadeh, A. Mokhtari, H. Hassani, A. Jadbabaie, and R. Pedarsani, "Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2020, pp. 2021–2031.

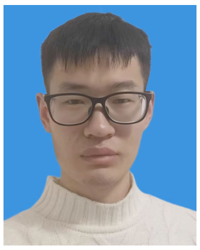
ZHENGXIONG MAO was born in November 1972. He received the B.S. degree. He is currently a Senior Engineer with the Information Center of Yunnan Power Grid Company. His main research interests include power grid informatization, information operation and maintenance, information security, and data mining and analysis.

HUI LI was born in February 1991. He received the master's degree. He is currently a mid-level Engineer with the Information Center of Yunnan Power Grid Company. His main research interests include application technology, data mining and analysis, and data asset operations and other research work.

ZUYUAN HUANG received the M.S. degree from Yunnan University, in 2021. He is currently with the Information Center of Yunnan Power Grid Company, Data Resource Center, as the Manager. His main research interests include big data and AI.

CHUANXU YANG was born in November 1981. He received the B.S. degree from North China Electric Power University. He is currently a Senior Engineer with the Information Center of Yunnan Power Grid Company. His main research interests include information and communication, power Digitalization Construction Management.

YANAN LI (Member, IEEE) received the master's degree from Henan Normal University, China, in 2007, and the Ph.D. degree from Xi'an Jiaotong University, China, in 2022. He is currently with Henan Polytechnic University. His research interests include differential privacy, machine learning, and federated learning.



ZIHAO ZHOU (Member, IEEE) received the bachelor's degree from the School of Mathematics and Statistics, Xi'an Jiaotong University, China, in 2020, where he is currently pursuing the Ph.D. degree with the School of Mathematics and Statistics. His research interests include federated learning and edge-cloud intelligence.