A New Privacy Protection Method for Federated Learning in Smart Grids

Jianguo Wei

North China Electric Power University, School of Control and Computer Engineering
No. 2 Beinong Road, Changping District, Beijing, China
Email: *2582686754[at]qq.com*

Abstract: ***With the development of grid technology, smart meters have become part of people's lives. Compared to conventional meters, smart meters collect more abundant data and provide more intelligent monitoring information. However, in the power grid, data is often scattered across different locations, making it difficult to fully utilize its value. To better utilize the data, we adopt federated learning to aggregate training data and improve the model's performance, in order to plan local electricity scheduling more effectively. Due to concerns about data privacy, differential privacy with added noise is commonly used, but this approach can significantly impact model accuracy. To address these challenges, we propose a Federated Learning method based on Wiener Filtering for Adaptive Differential Privacy (WADP-FL). WADP-FL adaptively adds noise based on the importance of each layer and utilizes Wiener Filtering to maximize data privacy while preserving model accuracy. Through simulation experiments, we demonstrate that WADP-FL can effectively preserve data privacy in testing neural network models using the MNIST, FMNIST, and CIFAR-10 datasets. Compared to common differential privacy-based federated learning approaches, WADP-FL achieves a significantly improved model accuracy of 4.3%, 2.07%, and 1.86% on different datasets, respectively.***

Keywords: Federated Learning; Differential Privacy; Smart Grid; WADP-FL

1. Introduction

Since the 21st century, the power grid [1-5] has been continuously developing towards intelligence and digitization. Sensors distributed widely in various places continuously return power-related data from all aspects, and these sensors are based on smart meters and are equipped with GPS systems. The new generation of 5G communication technology ensures efficient information transmission. However, due to the fact that different power companies have consumer's personal information in their local data, using this data can pose a serious threat to their privacy. Therefore, we hope to adopt a way to achieve the final model effect without uploading local data, so we use federated learning to solve such problems.

Federated learning was first proposed by Google in 2016 to solve the problem of updating models on Android mobile terminals locally [6]. The ultimate goal is to achieve collaborative modeling while ensuring data privacy, security, and compliance, thereby achieving the effect of AI models. Introducing federated learning into the power system can effectively solve the problem of data silos within the power system. However, unfortunately, there are not many studies on the application of federated learning in the field of smart grids.

Federated learning can play a good role in updating models, but privacy leaks cannot be completely avoided. Attackers can launch attacks from uploaded gradients. After gradient reversal, they can obtain the original data and thus obtain sensitive information. In such solutions, homomorphic encryption or differential privacy is usually used to address the issue. Homomorphic encryption allows plaintext to be transformed into matching ciphertext operations. In federated learning, the Paillier additive homomorphic encryption algorithm is widely used, which allows additional operations on ciphertext. This can solve the problem of data fusion calculation and data privacy protection to some extent, but due to the large computational burden, using homomorphic encryption algorithms will bring a huge burden to smart meters. Differential privacy can solve the problem of large computation, but conventional differential privacy algorithms are usually achieved by adding noise, which can affect the final model accuracy.

In light of the above issues, we have proposed a novel differential privacy scheme and combined it with a Wiener filter to ensure the final model accuracy.

The contributions of this paper are as follows:
1) We propose a novel scheme (WADP-FL) based on differential privacy and Wiener filtering in the smart grid. The multi-party collaborative model for protecting data privacy is proposed to better ensure model accuracy while preserving privacy.
2) We propose a differential privacy hierarchical perturbation algorithm that adaptively adds noise according to the importance of different layers, thereby more reasonably protecting user privacy and security.
3) We propose a Wiener filtering method to reduce model noise, which reduces the noise introduced by the differential privacy mechanism and further improves the accuracy of the model.
4) Comparative experiments on relevant datasets show that compared with traditional federated learning algorithms based on differential privacy, the WADP-FL algorithm maximizes model accuracy.

2. Model and Algorithm

2.1 Method Overview

The training process of the system includes the following steps: First, the server sends the global model to each local aggregation center[7]. The local aggregation centers train the model using collected smart meter data. After the trained gradient is processed by an adaptive differential privacy algorithm, the gradient parameter is uploaded to the central server. Then, a filtering algorithm is applied to the aggregated result. Finally, the central server sends the aggregated result back to the users, and each local aggregation center updates their own model and the process is iterated continuously to complete the entire training process. The final model parameters will be shared by all local aggregation centers.

```
Algorithm 1: Overall Steps of Federated Learning
Initialize $\theta_t$ on the server side
Local aggregation centers collect local smart meter data
Add output layer and softmax activation layer for each training
round t = 1,2,3... do
  Select m = c*k clients ,c∈ (0,1) clients
    Download $\theta_t$ to each client k
  Differential privacy disturbance parameters
  for each client k∈m do
      wait for Client to synchronize
      Noise reduction using Wiener filter algorithm
      Aggregate local model
  end
end
```

2.2 Detailed description of the Algorithm

In this section, we will provide a detailed description of our method. Table 1 defines the important symbols that we frequently use.

Symbol	Definition
w	Local model parameters on the client side
w_t	Global model parameters in the t-th round of global training
$w_{t,c}$	Model parameters of the C-th client in the t-th round of global training
T	Total number of global training rounds
∇g	Local gradient
b	Training round
D	Datasets
ε	Privacy budget
AIM	Layer importance
J	Sample loss function
Δf	Sensitivity
L	Total number of layers
C	Gradient clipping value
α	Learning rate

2.3 Adaptive differential privacy algorithm

2.3.1 General Description

The adaptive differential privacy scheme based on different layers mainly includes the following steps. Unlike traditional Federated Learning algorithms, firstly, each client uses local data samples to pretrain the adopted model, and calculates the importance of each layer in the trained model[8]. Then, the central server distributes the current initial model to the selected clients according to the range needed by clients. The selected clients train their local data and calculate the local gradients using the gradient descent algorithm. Gaussian noise is added to the gradient parameters of different layers according to the importance of the model's different layers. Afterwards, the noisy gradients are uploaded to the central server. Then, the central server uses relevant algorithms to filter and fine-tune the global model using a public dataset, further improving the model's accuracy.

The algorithm steps are shown in the figure below:

```
Algorithm 2: Client-side adaptive noise addition algorithm process
Update local model parameter $w_t \rightarrow w$
for round t = 1,2,3... do
  for each batch $b \in D_k$
    Calculate gradient $\nabla g = \nabla_w J(w,b)$
    if i = E：
      for each layer，j = 1,2,...L do
        Calculate gradient clipping value for each layer $c^{(j)} = c[j]$
      Clip each layer $\nabla g = \nabla g / \max(1, \|\nabla g\|_2 / c^{(j)})$
    Update parameters $w - \alpha \nabla g \rightarrow w$
  end for
end for
Calculate model difference: $\Delta w_t = \Delta w_t - w$
for Parameters of each layer：j = 1,2,...L do
  Calculate sensitivity $\Delta f_i = 2ac^{(j)}$
  Calculate privacy budget: $\varepsilon_i = (AIM^{(j)} / \sum_{j=1}^{L} AIM^{(j)}) * \varepsilon$
 Add noise:
$\Delta w_t^j = \Delta w_t^j + Guass(\Delta f_i / \varepsilon_i)$
end for
return $\Delta w_t$
```

2.3.2 Details of the noise addition algorithm

Firstly, the client obtains the initial model and uses local data to train the model. Then, we use a small portion of the data samples to calculate the importance of each layer. Inspired by Hu et al.'s [9] APoZ method for calculating the importance of neurons, we propose a method called AIM for calculating the importance of layers. The calculation formula is shown in the figure as follows:

$$AIM^{(i)} = \frac{\sum_{h=1}^{H} \sum_{j=1}^{K} f(o_j^{(i)}(m) > -2)}{H * K} \tag{4}$$

In the formula, represents the value of the jth neuron in the ith layer of the model before inputting the Sigmoid activation function when the mth validation sample is input to the model. H is the number of validation samples and K is the number of neurons in the ith layer of the model. Due to the characteristic of the Sigmoid activation function having a value range between 0 and 1, when approaches negative infinity, the value of the neuron after the activation function tends to be 0. Due to the calculation process of multiplication, addition, and activation unique to neural network models, neurons with output values close to 0 have little contribution to the output of subsequent layers and the final results. Since Sigmoid(-2) ≈ 0.1, the threshold for AIM is set to -2. When the input value of the function in the formula is greater than -2, the result of the function f() is 1; otherwise, the result of the function f() is 0.

Based on the calculated importance of each layer, we add different amounts of noise to the model. We add more noise to the layers with higher importance and less noise to the layers with lower importance. In the overall training process, we selectively add noise to ensure the accuracy of the final data model. The amount of noise added is controlled by the privacy budget, and the calculation formula for the privacy budget is as follows:

$$\varepsilon_i = \frac{AIM^i}{\sum_{i=1}^{N} AIM^i} \times \varepsilon \qquad (5)$$

where N represents the total number of layers in the model, i represents the ith layer of the model, represents the privacy budget of the model. In the scenario where the privacy budget is not allocated based on layer importance, the privacy budget for each layer is calculated as $\varepsilon_i = \varepsilon / N$.

2.4 Noise filtering method based on Wiener filtering

During the aggregation process, Wiener filtering is used to process the gradient model parameters provided by each local node, denoise the parameters, and aggregate them to obtain new global model parameters for the next round of iteration until the model converges.

Since the noise we add follows a Gaussian distribution with a mean value of 0, we can use Wiener filtering to denoise the model parameters. The specific process is as follows:

1) Update of model parameters

After model training, the parameter update equation is:

$$\begin{cases} w_{t+1} = w_t - \lambda \sum_{c \in C} g'_{t,c} \\ w_{t+1,1} = w_{t,1} - \lambda g_{t,1} + N(0, \sigma^2 M^2 I) \\ \dots \\ w_{t+1,c} = w_{t,c} - \lambda g_{t,c} + N(0, \sigma^2 M^2 I) \end{cases} \qquad (8)$$

Where w_t represents the global model parameters of global training in the t-th round, w_{t+1} represents the parameters for the next round of iteration, $N(0, \sigma^2 M^2 I)$ represents the noise mechanism that satisfies the Gaussian distribution. We add Gaussian noise in each round of iteration.

To transform the parameter equation for each round of model training into matrix form, The parameter vector formed by the node models is: $w_t = [w_t, w_{t,1}, w_{t,2} \dots w_{t,N}]$,The gradient vector formed by all gradient values is: $G'_t = [0, g'_{t,1}, g'_{t,2} \dots g'_{t,c}]$.

2) Parameter update correction

In Wiener filtering, we use the frequency-domain Wiener filtering approach, where $H(w_k)$ is the coefficient of the Wiener filter and $Y(w_k)$ is the noisy signal parameters that are transmitted.

Firstly, we perform a Fourier transform on the noise signal $x = h(n) * y(n)$ in the time domain to transform it to the frequency domain. The result is:

$$X' = H(w) * Y(w_k) \qquad (9)$$

The resulting error is:

$$E(w_k) = X(w_k) - H(w_k) Y(w_k) \qquad (10)$$

According to the principle of minimum mean error, we can obtain:

$$\begin{aligned} \mathrm{E}[|E(w_k)|^2] &= E\{[X(w_k) - H(w_k)Y(w_k)] \\ &*[X(w_k) - H(w_k)Y(w_k)]\} \end{aligned} \qquad (11)$$

Expanding and setting :

$$P_{yy}(w_k) = E[|Y(w_k)|^2] \qquad (12)$$

$$P_{dy}(w_k) = E[Y(w_k) D^*(w_k)] \qquad (13)$$

Taking the derivative of H yields:

$$\frac{\partial E}{\partial H(w_k)} = [H(w_k) P_{yy}(w_k) - P_{dy}(w_k)] = 0 \qquad (14)$$

Simplifying the derivation results in:

$$H(w_k) = \frac{P_{yy}(w_k)}{P_{dy}(w_k)} \qquad (15)$$

Where ,

$$\begin{aligned} P_{dy}(w_k) &= E[X(w_k)X^*(w_k)] + E[X(w_k)N^*(w_k)] \\ &= P_{xx}(w_k) \end{aligned} \qquad (16)$$

$$P_{yy}(w_k) = P_{xx}(w_k) + P_{nn}(w_k) \qquad (17)$$

Substituting the given values yields the result:

$$H(w_k) = \frac{P_{xx}(w_k)}{P_{xx}(w_k) + P_{nn}(w_k)} \qquad (18)$$

Define prior signal-to-noise ratio:

$$\xi_k \square \frac{P_{xx}(w_k)}{P_{nn}(w_k)} \qquad (19)$$

Substituting the prior signal-to-noise ratio into H (w_k) yields:

$$H(w_k) = \frac{\xi_k}{\xi_{k+1}} \qquad (20)$$

3. Experimental Procedure

3.1 Description of the experiment

(1) Model structure
The CNN model [10]structure used in this experiment consists of 3 convolutional layers and 1 fully connected layer, as shown in the diagram below:

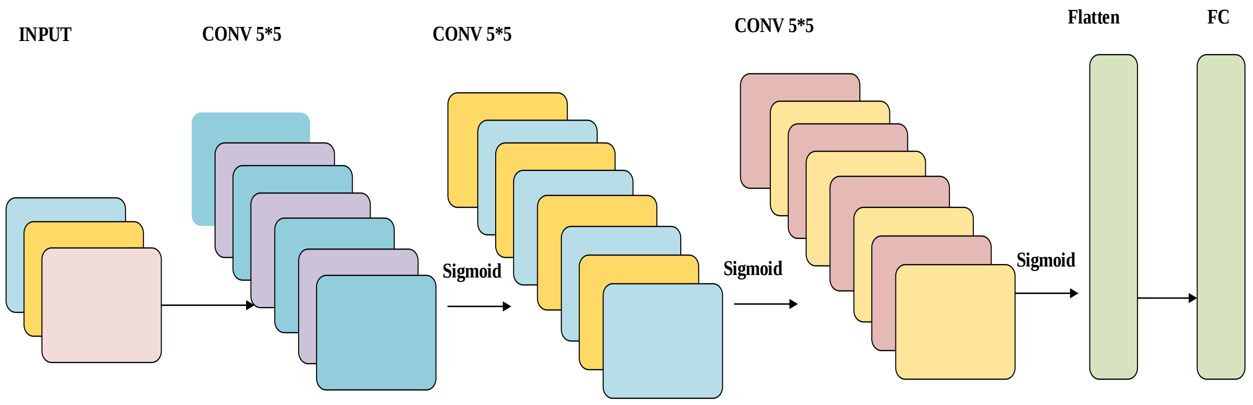

Figure 2: Model accuracy on different Federated Learning aggregation methods

(2) Specific parameter settings
In the Federated Learning experiment, a Federated Learning framework with 20 clients and one central server was used. The local clients in the framework used the same parameters, with a local iteration epoch set to 100, learning rate set to 0.01, and SGD momentum set to 0.9. The model on the central server was trained for 50 times, and the probability of selecting a client was set to 0.7.

For the MNIST and Fashion-MNIST experiments, 1000 randomly sampled test samples were selected as the public dataset for fine-tuning the global model. For the CIFAR10 experiment, 3000 randomly selected test samples were used as the public dataset for fine-tuning the model.

3.2 Experimental results

(1)Traditional differential privacy versus adaptive differential privacy
We first trained a model on the CIFAR10 dataset with an accuracy of 97.56% and used it as a benchmark for comparison. We then trained the model under two conditions: normal noise addition and noise addition using adaptive privacy budget. To control variables, we used filtering for both algorithms and compared the model accuracies, as shown in Table 1 below:

Table 1: Comparison of training accuracy between traditional differential privacy and adaptive differential privacy models

Privacy budget	Baseline (ACC)	Adaptive-DP (ACC)	DP (ACC)	Growth rate
$\varepsilon = 0.5$	97.56%	26.98%	22.43%	19.88%
$\varepsilon = 1.0$	97.56%	85.65%	43.66%	96.17%
$\varepsilon = 5.0$	97.56%	92.67%	86.65%	6.9%
$\varepsilon = 10.0$	97.56%	95.81%	90.53%	5.8%

As shown in Table 1:
a) Adding noise will affect the training accuracy of the model to a certain extent, especially when the privacy budget is small. Adding noise in this case will have a significant impact on the final model accuracy
b) Compared to traditional methods, the stratified additive adaptive differential privacy method greatly improves the accuracy of model training.
c) As the privacy budget increases, the accuracy of the adaptive differential privacy model approaches the accuracy of the baseline model without noise.

(2) Validation on different datasets
We validate our approach using the Fashion-MNIST and CIFAR10 datasets, with privacy budgets of 1.0 and 5.0 for WADP-FL on these two datasets, respectively. After training for a certain number of epochs, the experimental results shown in Table 2 indicate that this model is applicable to other datasets and has better compatibility.

In addition, tests were performed on different neural network models, ResNet [11] and AlexNet [12], and WADP-FL showed improved accuracy compared to the DP-FedAvg method, indicating that this approach is compatible with different neural network structures.

Table 2: Comparison of model training accuracy on different datasets

	F-MNIST(ACC)		CIFAR10 (ACC)	
solution	ε= 1.0	ε= 5.0	ε= 1.0	ε= 5.0
FedAvg	85.66%	87.65%	84.66%	87.37%
DP-FedAvg	83.56%	85.46%	82.68%	85.35%
WADP-FL-Fed	84.29%	87.23%	83.89%	86.94%

4. Conclusion

This article proposes an adaptive differential privacy Federated Learning scheme based on Wiener filtering. The method calculates the importance of different layers in the model structure, adds different amounts of noise, and combines Wiener filtering algorithm to filter the noise after aggregation. Through experiments on relevant datasets such as MNIST, F-MNIST, CIFAR10, and experiments on various neural network models, the method's ability to ensure model accuracy and improve model precision is validated by comparison with other popular methods, thus effectively solving the problem of differential privacy in Federated Learning, which affects the final model accuracy due to blind noise addition.

However, since it is difficult to obtain real-world data from smart grids, the experimental results may not be fully applicable to real-world applications. Therefore, this article still has many shortcomings that need to be further supplemented and strengthened. The proposed method requires a certain level of computational power on the terminal, and future work should continue to consider the diversity of scenarios to better address the application of real-world scenarios.

References

[1] M. L. Tuballa and M. L. Abundo, “A review of the development of Smart Grid technologies,” Renew. Sustain. Energy Rev., vol. 59, pp. 710 - 725, Jun. 2016.
[2] Hasan M K, Habib A K M A, Shukur Z, et al. Review on cyber-physical and cyber-security system in smart grid: Standards, protocols, constraints, and recommendations [J]. Journal of Network and Computer Applications, 2023, 209: 103540.
[3] Dewangan F, Abdelaziz A Y, Biswal M. Load Forecasting Models in Smart Grid Using Smart Meter Information: A Review[J]. Energies, 2023, 16(3): 1404.
[4] Jafari M, Kavousi-Fard A, Chen T, et al. A review on digital twin technology in smart grid, transportation system and smart city: Challenges and future[J]. IEEE Access, 2023.
[5] Li X, Wu F, Kumari S, Xu L, Sangaiah AK, Choo K-KR. A provably secure and anonymous message authentication scheme for smart grids. J Parallel Distrib Comput 2019; 132:242-9.
[6] C. Dwork, “Differential privacy: A survey of results,” in International conference on theory and applications of models of computation, 2008.
[7] Jia Feixuan. Research on privacy-preserving methods of federal learning based on differential privacy [D]. Xidian University, 2022. DOI:10.27389/d.cnki.gxadu.2022.001637.
[8] HU H, PENG R, TAI Y W, et al. Network Trimming: A Data-Driven Neuron Pruning Approach towards Efficient Deep Architectures [J]. CoRR, 2016, abs/1607.03250.
[9] Zhang J, He T, Sra S, et al. Why gradient clipping accelerates training: A theoretical justification for adaptivity[J]. arXiv preprint arXiv:1905.11881, 2019.
[10] Kavitha R, Jothi D K, Saravanan K, et al. Ant colony optimization-enabled CNN deep learning technique for accurate detection of cervical cancer[J]. BioMed Research International, 2023.
[11] Targ S, Almeida D, Lyman K. Resnet in resnet: Generalizing residual architectures [J]. arXiv preprint arXiv:1603.08029, 2016.
[12] Alom M Z, Taha T M, Yakopcic C, et al. The history began from alexnet: A comprehensive survey on deep learning approaches [J]. arXiv preprint arXiv:1803.01164, 2018.