Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2023.040731

ARTICLE

Decentralized Heterogeneous Federal Distillation Learning Based on Blockchain

Hong Zhu*, Lisha Gao, Yitian Sha, Nan Xiang, Yue Wu and Shuo Han

Nanjing Power Supply Branch, State Grid Jiangsu Electric Power Co., Ltd., Nanjing, 210000, China

*Corresponding Author: Hong Zhu. Email: academic2023@163.com

Received: 29 March 2023 Accepted: 13 June 2023 Published: 08 October 2023

ABSTRACT

Load forecasting is a crucial aspect of intelligent Virtual Power Plant (VPP) management and a means of balancing the relationship between distributed power grids and traditional power grids. However, due to the continuous emergence of power consumption peaks, the power supply quality of the power grid cannot be guaranteed. Therefore, an intelligent calculation method is required to effectively predict the load, enabling better power grid dispatching and ensuring the stable operation of the power grid. This paper proposes a decentralized heterogeneous federated distillation learning algorithm (DHFDL) to promote trusted federated learning (FL) between different federates in the blockchain. The algorithm comprises two stages: common knowledge accumulation and personalized training. In the first stage, each federate on the blockchain is treated as a meta-distribution. After aggregating the knowledge of each federate circularly, the model is uploaded to the blockchain. In the second stage, other federates on the blockchain download the trained model for personalized training, both of which are based on knowledge distillation. Experimental results demonstrate that the DHFDL algorithm proposed in this paper can resist a higher proportion of malicious code compared to FedAvg and a Blockchain-based Federated Learning framework with Committee consensus (BFLC). Additionally, by combining asynchronous consensus with the FL model training process, the DHFDL training time is the shortest, and the training efficiency of decentralized FL is improved.

KEYWORDS

Load forecasting; blockchain; distillation learning; federated learning; DHFDL algorithm

1 Introduction

With the increasingly prominent natural environmental problems, the constraint of fossil energy becomes tighter. It is imperative to develop renewable energy and promote national energy transformation vigorously. However, due to its small capacity, decentralized layout, and strong randomness of output, distributed renewable energy will have an impact on the security and reliability of the grid when it is connected to the grid alone, so it is difficult to participate in the power market competition as an independent individual. The large-scale integration of distributed power grid requires intelligent centralized management to coordinate the power grid effectively. As an important form of intelligent management, VPP [1] not only fosters enhanced interaction among users but also bolsters the stability of the power grid. However, the continuous emergence of power consumption peaks increases the

load of virtual power plants, affecting the grid's power supply quality. Consequently, the VPP requires a sophisticated computational approach to accurately predict load demands, thereby enabling more efficient power grid dispatching and management.

In the load forecasting business, each VPP platform precipitates large enterprise power consumption data. The accuracy of power load forecasting can be improved through cross-domain collaborative computing of data. Traditional machine learning requires centralized training of data. However, in the cross-domain transmission link of data within the power grid, there are problems such as data theft, data tampering, data power, responsibility separation, and low transmission efficiency. The power grid has the right to access the power data of the enterprise, which can be viewed internally, but does not have the right to disclose the data. Once the power data of the enterprise is stolen, sold, and disclosed in the transmission link, it will cause a great blow to the credibility of the power grid. At the same time, the power and responsibility caused by the cross-domain data need to be clarified, and different branches pay different attention to the same data, which may further increase the risk of data leakage.

FL can protect data security and ensure the consistency of data rights and responsibilities as a kind of privacy-protected distributed machine learning. It is a secure collaborative computing for data confirmation. On the other hand, it can eliminate data transmission links and reduce the energy consumption of collaborative computing. It is a kind of green collaborative computing. FL can ensure that the local data owned by the participants stay within the control of the participants and conduct joint model training. FL can better solve the problems of data islands, data privacy, etc. At present, FL has been widely used in various fields.

However, the existing FL primarily relies on the parameter server to generate or update the global model parameters, which is a typical centralized architecture. There are problems such as single-point failure, privacy disclosure, performance bottlenecks, etc. The credibility of the global model depends on the parameter server and is subject to the centralized credit model. Traditional FL relies on a trusted centralized parameter server, in which multiple participants cooperate to train a global model under the condition that their data does not go out of the local area. The server collects local model updates, performs update aggregation, maintains global model updates, and other centralized operations. The entire training process is vulnerable to server failures. Malicious parameter servers can even poison the model, generate inaccurate global updates, and then distort all local updates, thus making the entire collaborative training process error. In addition, some studies have shown that unencrypted intermediate parameters can be used to infer important information in training data, and the private data of participants are exposed. Therefore, in the process of model training, it is particularly important to adopt appropriate encryption schemes for local model updates and maintain the global model on distributed nodes. As a distributed shared general ledger jointly maintained by multiple parties, the blockchain realizes the establishment of the trust relationship between participants without relying on the credit endorsement of a trusted third party through the combined innovation of multiple technologies such as distributed ledger technology, cryptographic algorithms, peer-to-peer communication, consensus mechanism, smart contracts, etc. It can be used to replace the parameter server in FL and store relevant information in the model training process.

In the peer-to-peer cooperative computing scenario, the traditional centralized FL has the disadvantages of low communication efficiency, slow aggregation speed, insecure aggregation, and untrustworthy aggregation. First, the aggregation node needs to consume a large amount of computing and communication resources. However, in the peer entities, the benefits are equal, and the entities are unwilling to take responsibility for aggregation tasks and bear redundant responsibilities and resource consumption. Secondly, in the process of aggregation, there are malicious attacks. On the one hand,

aggregation nodes can maliciously reduce the aggregation weight of a cooperative subject so that the global model deviates from its local model, and targeted attacks can be achieved. On the other hand, the aggregation node can retain the correct model and distribute the tampered model to achieve a global attack. Finally, the global model trained by the aggregation node has a weak prediction effect for a single agent and cannot be personalized. In practical applications, due to data heterogeneity and distrust/nonexistence of the central server, different federations can only work together sometimes.

To sum up, this paper proposes a decentralized asynchronous federated distillation learning algorithm. Through circular knowledge distillation, the personalized model of each federation is obtained without a central server. Then the trained model is uploaded to the blockchain for other federations on the chain to download to the local for training.

Our contributions are as follows:

a) We propose an asynchronous FL distillation algorithm that integrates blockchain and federated learning, which can accumulate public information from different federations without violating privacy and implement personalized models for each federation through adaptive knowledge distillation.
b) Asynchronous consensus is combined with FL to improve the efficiency of model uplink.
c) By comparing the FedAvg algorithm, BFLC [2] algorithm, and the DHFDL algorithm proposed in this paper, it can be seen that the DHFDL training of asynchronous uplink aggregation of models through asynchronous consensus takes the shortest time and has the highest efficiency.

2 Related Work

2.1 Federated Learning

FL [3] was launched by Google in 2016 to solve the problems of data privacy and data islands in AI. Its essence is that the central server pushes the global model to multiple data parties participating in FL and trains the model in multiple data parties. The data party transmits the updates of local training to the central server, which aggregates these updates to generate a new global model and then pushes it to the data party. The architecture of FL is shown in Fig. 1.

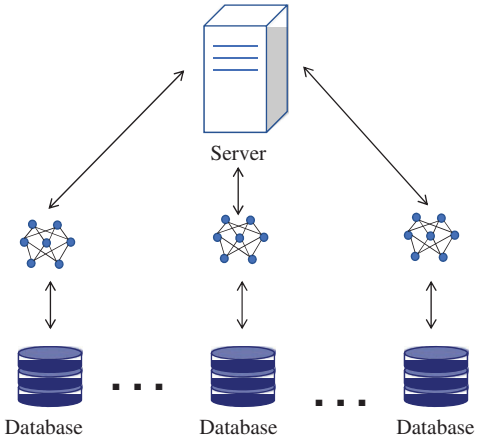


Figure 1: General FL architecture

To fully use the data of different independent clients while protecting data privacy and security, Google has proposed the first FL algorithm FedAvg to summarize customer information. FedAvg trains the machine learning model by aggregating data from distributed mobile phones and exchanging model parameters rather than directly exchanging data. FedAvg can well solve the problem of data islands in many applications. However, simple FedAvg cannot meet complex reality scenarios, and when meeting the statistical data heterogeneity, FedAvg may converge slowly and generate many communication costs. In addition, because only the shared global model is obtained, the model may degenerate when making predictions in the personalized client. Reference [4] combined three traditional adaptive technologies into the federated model: fine-tuning, multi-task learning, and knowledge distillation. Reference [5] attempted to deal with feature changes between clients by retaining local batch processing normalization parameters, which can represent some specific data distribution. Reference [6] proposed introducing the knowledge distillation method into FL so that FL can achieve better results on the local data distribution might be not Independent and Identically Distributed (Non-IID). Reference [7] evaluated FL's model accuracy and stability under the Non-IID dataset. Reference [8] proposed an open research library that allows researchers to compare the performance of FL algorithms fairly. In addition, the research library also promotes the research of various FL algorithms through flexible and general Application Programming Interface (API) design. Reference [9] proposed a sustainable user incentive mechanism in FL, which dynamically distributes the given budget among the data owners in the federation, including the received revenue and the waiting time for receiving revenue, by maximizing the collective effect and the way perceived below, to minimize the inequality between data owners. Reference [10] proposed a new problem called federated unsupervised representational learning to use unlabeled data distributed in different data parties. The meaning of this problem is to use unsupervised methods to learn data distributed in each node while protecting user data privacy. At the same time, a new method based on dictionary and alignment is proposed to realize unsupervised representation learning.

The purpose of FL is to train in-depth learning models on the premise of ensuring user privacy. However, the transmission of model updates involved in general FL has been proved in Reference [11] that gradients can disclose data, so we can see that there is still a risk of data privacy disclosure in general FL. Therefore, the research on FL security is also a valuable direction. The following summarizes some research results on FL security. Reference [12] proposed introducing a differential privacy algorithm into FL to construct false data sets with similar data distribution to real data sets to improve the security of real data privacy. Reference [13] proposed to apply secure multi-party computing (SMC) and differential privacy at the same time and achieve a balance between them so that FL can achieve better reasoning performance while achieving the security brought by differential privacy. Reference [14] proposed an algorithm combining secret sharing and Tok-K gradient selection, which balances the protection of user privacy and the reduction of user communication overhead, reduces communication overhead while ensuring user privacy and data security, and improves model training efficiency.

2.2 Knowledge Distillation

Knowledge distillation is a technique that extracts valuable insights from complex models and condenses them into a singular, streamlined model, thereby enabling its deployment in real-world applications. Knowledge distillation [15] is a knowledge transfer and model compression algorithm proposed by Geoffrey Hinton et al. in 2015. For a specific character, through the use of a knowledge distillation algorithm, the information of an ideally trained teacher network containing more knowledge can be transferred to a smaller untrained student network.

In this paper, the loss function $L_{student}$ of the student network can be defined as:

$$L_{student} = L_{CE} + L_{KL}\left(p_{student} || p_{teacher}\right) \tag{1}$$

$$p = \frac{\exp\left(\frac{z_i}{T}\right)}{\sum_j \exp\left(\frac{z_j}{T}\right)} \tag{2}$$

LCE is the cross entropy loss function, LKL is the Kullback Leibler (KL) divergence, $p_{student}$ and $p_{teacher}$ are the outputs of the network after the softmax activation function, z is the output logits of the neural network, and T is the temperature, which is generally set as 1. The primary purpose of temperature is to reduce the loss of knowledge contained in the small probability results caused by excessive probability differences. KL divergence can measure the difference between the two models. The larger the KL divergence, the more significant the distribution difference between the models, and the smaller the KL divergence, the smaller the distribution difference between the two models. The formula of KL divergence is:

$$L_{KL}(P||Q) = \sum P(x) \log \frac{P(x)}{Q(x)} \tag{3}$$

where P(x) and Q(x) respectively represent the output of different networks after the softmax activation function.

2.3 Federated Learning Based on Blockchain

Reference [16] proposed a trusted sharing mechanism that combines blockchain and FL to achieve data sharing, protecting private data and ensuring trust in the sharing process. Reference [2] proposed an FL framework based on blockchain, using committee consensus BFLC. This framework uses blockchain to store global and local models to ensure the security of the FL process and uses special committee consensus to reduce malicious attacks. Reference [17] designed a blockchain-based FL architecture that includes multiple mining machines, using blockchains to coordinate f FL tasks and store global models. The process is as follows: nodes download the global model from the associated mining machine, train it, and then upload the trained local model as a transaction to the associated mining machine. The miner confirms the validity of the uploaded transaction, verifies the accuracy of the model, and stores the confirmed transaction in the candidate block of the miner. Once the candidate block has collected enough transactions or waited for a while, all miners will enter the consensus stage together, and the winner of PoW will publish its candidate block on the blockchain. In addition, miners can allocate rewards to encourage devices to participate in FL when they publish blocks on the blockchain. The recently emerged directed acyclic graph-based FL framework [18] builds an asynchronous FL system based on asynchronous bookkeeping of directed acyclic graphs to solve the device asynchrony problem in FL.

Reference [19] proposed to enhance the verifiable and auditable of the FL training process through blockchain, but the simultaneous up-chaining of the models through the committee validation model is less efficient. Reference [20] proposed data sharing based on blockchain and zero-knowledge proof, However, it is not suitable for computing and data sharing of complex models. Reference [21] proposed a verifiable query layer for guaranteeing data trustworthiness, but this paper's multi-node model verification mechanism is more suitable for FL.

3 Method

3.1 Problem Analysis

In an ideal situation, the current decentralized FL solutions have been proven to work well. However, in the actual scene, problems such as model training speed and federated learning security still pose a huge challenge to the existing decentralized FL algorithm. For the model training speed, synchronous model aggregation slows down the model updating speed when the equipment performance of multiple participants is different. In terms of the security of federated learning, decentralized, federated learning faces not only the data poisoning of malicious nodes but also the problem of information tampering. Malicious nodes undermine the security of FL by tampering with the model parameters or gradient of communication. Different nodes have different FL computing and communication resources. In conventional centralized synchronous FL systems, a single node needs to wait for other nodes to finish their tasks before moving on to the next round. Only after completing their training tasks can they enter the next round together. However, if a node is turned off during training, it may completely invalidate a round of FL.

Blockchain is a decentralized asynchronous data storage system. All transactions verified in the blockchain will be permanently stored in the blockchain blocks and cannot be tampered with. In addition, the blockchain uses a consensus algorithm to verify transactions, which can effectively prevent malicious nodes from tampering with transaction information. In decentralized FL, blockchain asynchronous consensus can be used to accelerate the model aggregation efficiency, taking the model training speed as an example. Therefore, this paper introduces blockchain technology based on a decentralized FL framework so that network communication load, FL security, and other indicators can reach better standards.

3.2 Architecture Design

Considering the absence of a central server among different federal bodies, the key to enabling them to share knowledge without the involvement of other administrators and without directly exchanging data is crucial. The objective of a blockchain-based FL architecture is to accumulate public knowledge through knowledge distillation while preserving data privacy and security and storing personalized information. As shown in Fig. 2, decentralized heterogeneous federated distillation learning architecture is divided into the bottom algorithm application layer, the blockchain layer for trustworthy model broadcasting and endorsement, and the asynchronous federated distillation learning part for model training.

On the blockchain, we design two different types of blocks to store public models and local models. FL training relies only on the latest model blocks, and historical block storage is used for fault fallback and block validation. The data storage structure on the blockchain is shown in Fig. 3.

The public model block is created in the common knowledge accumulation phase. In the common knowledge accumulation phase, nodes use local data for model training and then access the blockchain to obtain the latest public model. The public model acts as a teacher to enhance the local model by knowledge distillation, and the local model that completes knowledge distillation is chained as a new public model block. When the public model blocks with the same TeacherID accumulate to a certain number, the model aggregation smart contract is triggered to generate a new public model block. The public model block includes the block header, TeacherID, this model ID, model evaluation score, IsPublic, IsAggreation, and model parameters.

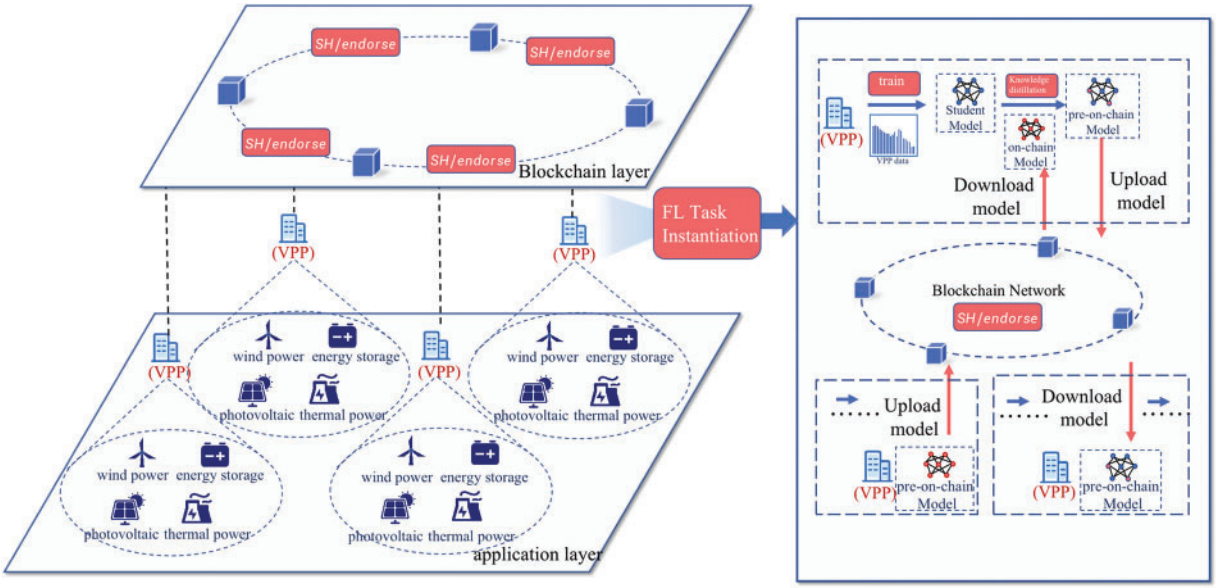


Figure 2: Blockchain-based FL architecture

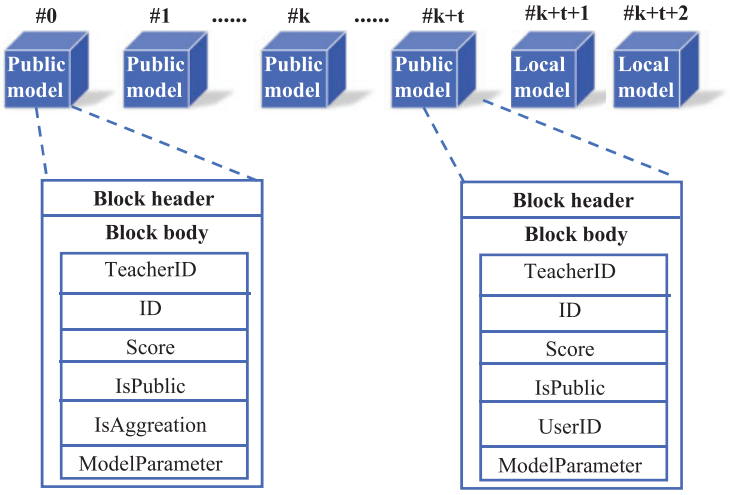


Figure 3: Data storage structure on blockchain

Local model blocks are created in the personalization phase. When the accuracy of the public model reaches a threshold, the subsequent participating nodes trigger the personalization training smart contract and enter the personalization training phase. The public model we obtained in the previous phase is used as a pre-trained model input in the personalization phase to improve the node's local model, and no new public model blocks are generated. The participating nodes in this phase first perform local model training, then download the public model and use the public model as a teacher to fine-tune the local model by distillation learning. The fine-tuned local model is up-linked as a new personalized model block. Nodes download each new personalized model block, verify the personalized model uploaded by other nodes and try to use it for fine-tuning the local model, and the fine-tuned personalized model is also uploaded to the blockchain for knowledge sharing. The

personalized model block includes the block header, TeacherID, ID of this model, IsPublic, UserID of the user to which the model belongs, model evaluation score, and model parameters.

3.3 Model Asynchronous Uplink Mechanism Based on Honeybadger Consensus

The model uplink consensus mechanism based on honeybadger consensus is shown in Fig. 4:

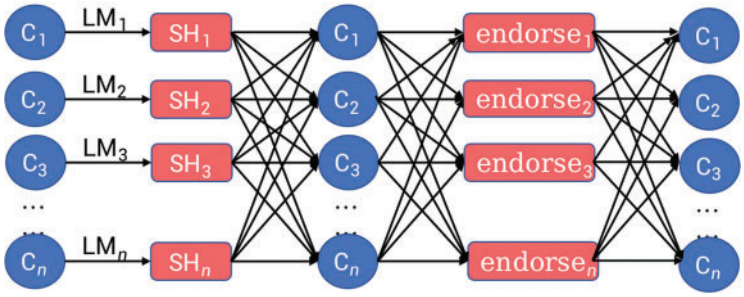


Figure 4: Flow chart of model chain consensus mechanism

Where C_n participates in the FL training, that is, the load forecasting business of the virtual power plant; LM_n is a local model for participants to train based on local data, namely personalized training. The SH_n module is used for the trusted broadcast of each local model. The $endorse_n$ module is used for model endorsement. When a local model collects a certain number of endorsements, it performs the model up chain step through the smart contract.

3.3.1 SH_n Module

As an anonymous transmission channel based on secret sharing, the SH_n module can obfuscate the uploader's address of the model, preventing malicious nodes from reproducing the data and launching targeted attacks against the model owner by knowing the model's source in advance.

After construction, the SH_n module is used for on-chain updates of the FL local update block. The local update block serves as a black box input, and the on-chain verification nodes cannot access or modify the information inside the block before completing the anonymous transmission.

The SH_n module satisfies the following properties:

(Validity) If an honest node outputs integrity verification set V against the received locally updated model integrity, then $|V| \geq N - f$ and v contain the verification of at least $N - 2f$ honest nodes.

(Consensus) If one honest node outputs integrity verification set V, then other nodes should also output V.

(Integrity) If $N - f$ correct nodes receive input, then all nodes generate an output.

N means the number of participating nodes is N, and f means the number of malicious nodes is f.

The anonymous transmission channel SH_n is implemented based on secret sharing. Firstly, the node needs to generate a public key and N private keys SK_i according to the node ID. Then, the public key is used to encrypt the model and distribute the encrypted model and public key to other nodes. For the encrypted model, multiple nodes need to cooperate to complete decryption. Once the $f+1$ honest node decrypts the ciphertext, the encrypted model will be restored to a usable model. Unless an honest node leaks the model after decryption, the attacker cannot complete the decryption of the model ciphertext. The SH_n process is as follows:

SH.setup()->PK,{SK_i}, generates the cryptographic public key PK for the local model update and SK_i, a set of private keys for decrypting the cryptographic model.

SH.Enc(PK,m)->C, encrypts the local model update m using the public key and generates the encrypted model C.

SH.DecShare(SK_i, C) distributes the encrypted model and key to each node.

SH.Dec(PK, C, {i, SK_i})->m, aggregate {i, SK_i} from at least $f+1$ nodes to obtain the private key SK corresponding to PK, and use SK to decrypt each node to obtain a usable local update model.

3.3.2 Endorse$_n$ Module

The *endorse$_n$* module is used to verify the update of the FL local model. All nodes verify the model passed by SH_n and give the verification vote. The N concurrent instances of binary Byzantine are used for the counterpoint vector where $b = 1$ indicates that the node agrees to chain the model.

The *endorse$_n$* module satisfies the following properties:

(Consensus) If any honest node endorses the model output with agreement endorsement b, then every honest node outputs agreement endorsement b.

(Termination) If all honest nodes receive the input model, then every honest node outputs a 1-bit value indicating whether it agrees to endorse the model or not.

(Validity) If any honest node outputs b, then at least one honest node accepts b as input.

The validity property implies consistency: if all correct nodes receive the same input value b, then b must be a deterministic value. On the other hand, if two nodes receive different inputs at any point, the adversary may force the decision of one of the values before the remaining nodes receive the input.

3.4 Asynchronous Federated Distillation Learning

The training is divided into a common knowledge accumulation stage and a personalization stage. Specifically, in the common knowledge accumulation stage, each federation on the blockchain is regarded as a meta-distribution, and the knowledge of each federation is aggregated cyclically. After the knowledge accumulation is completed, the model is uploaded to the blockchain so that other federations on the blockchain can perform personalized training. The common knowledge accumulation stage lasts for several rounds to ensure that the public knowledge of each federation is fully extracted. In the personalization stage, the federation in the blockchain downloads the trained model from the chain to the local for guidance training, and the personalization stage can also be trained in the same way. Since the public knowledge has been accumulated, the local training Before sending the public knowledge model to the next federation. Both stages are based on knowledge distillation.

In the first stage, we divide the specific steps into four steps. In short, the four steps are operated as follows:

a) Train: Using the local dataset to train the local model as a student model
b) Download: Download the on-chain model for distillation
c) Distill: Using knowledge distillation to enhance the local model to get the pre-on chain model
d) Upload: Upload the pre-on chain model to the blockchain

The detailed procedures are as follows:

In the following, let each client k = 1, 2, ..., K, have a local dataset $\left(d_{i,k}^{p}, t_{i,k}\right)_{i=1}^{\mathrm{I_k}}$, where $\left(d_{i,k}^{p}\right)_{i=1}^{\mathrm{I_k}}$ represents the vectorized input samples in the dataset, and $\mathrm{I_k}$ represents the number of samples in the local dataset. $\mathrm{N_L}$ denotes the number of objective classes, and $\mathrm{N_S}$ denotes the dimension of the input samples. The label attached to the sample $d_{i,k}^{p}$ is represented by $t_{i,k} = [t_{i,k,1}, t_{i,k,1}, \ldots, t_{i,k,\mathrm{N_L}}]\mathrm{T}$ in the one-hot representation, where the element $t_{i,k,n}$ equals 1 if the $\mathrm{n^{th}}$ label is the ground-truth and 0 otherwise. To simplify notation, let D_k^p be an $\mathrm{N_S} \times \mathrm{I_k}$ matrix that represents the concatenation of $\left(d_{i,k}^{p}\right)_{i=1}^{\mathrm{I_k}}$ for client k, and let T_k be an $\mathrm{N_L} \times \mathrm{I_k}$ matrix that represents the concatenation of $\left(t_{i,k}\right)_{i=1}^{\mathrm{I_k}}$ for client k.

1. Train. In this step, each client updates its model with its local dataset. The model parameters are updated as follows:

$$w^k \leftarrow w^0 - \eta_1 \nabla \varnothing \left(D_k^p, T_k | w^0\right) \tag{4}$$

where η_1 represents the learning rate, $\varnothing \left(D_k^p, T_k | w^0\right)$ denotes the loss function that is minimized in this step. The loss function is exemplified in classification problems by the cross-entropy. In this case, $\varnothing \left(D_k^p, T_k | w^0\right)$ is given as follows:

$$\varnothing \left(D_k^p, T_k | w^0\right) = -\sum_{i \in \rho} \sum_{n \in \sigma} t_{i,k,n} \log F_n \left(d_{i,k}^{p} | w^0\right) \tag{5}$$

where $F_n \left(d_{i,k}^{p} | w^0\right)$ denotes the $\mathrm{n^{th}}$ element of $F \left(d_{i,k}^{p} | w^0\right)$, $\sigma = \{1, 2, 3 \ldots\ldots \mathrm{N_L}\}$ and $\rho = \{1, 2, 3 \ldots\ldots \mathrm{I_k}\}$. The update procedure is an iterative process until a terminating condition, such as convergence or a predefined number of iteration times, is satisfied.

2. Download. In this step, each client downloads an on-chain model w^g for distillation.

3. Distill.

Based on the model learned in the previous step, each client predicts the local logit, which is the label for data samples in the local dataset. More specifically, given the model parameter, each client predicts local logits $\hat{\mathrm{t}}_{\mathrm{j,k}}$ for $\mathrm{j} \in \rho$ as follows:

$$\hat{\mathrm{t}}_{\mathrm{j,k}} = \mathrm{F}\left(\mathrm{d}_{\mathrm{i,k}}^{\mathrm{p}} | w^g\right) \tag{6}$$

For shorthand notation, the $\mathrm{N_L} \times \mathrm{I^k}$ matrix $\hat{\mathrm{T}}_{\mathrm{k}}$ denotes the concatenation of $\hat{\mathrm{t}}_{\mathrm{j,k}}$

The clients enhance their local model based on the logits $\hat{\mathrm{T}}_{\mathrm{k}}$ and local dataset $\mathrm{D_k^P}$. More concretely, the model parameters are updated as follows:

$$w^k \leftarrow w^k - \eta_2 \nabla \varnothing \left(D_k^p, \widehat{\mathrm{T}}_{\mathrm{k}} | w^k\right) \tag{7}$$

where η_2 is the learning rate in the proposed distillation procedure.

4. Upload. In this step, each client uploads the pre-on-chain model to the blockchain.

The second stage is the personalized training stage. Since there is no central server for the entire model, we must obtain the personalized model in the same order as the common knowledge accumulation stage. In the first stage, we obtain the public model f, which contains enough common knowledge. To prevent the common knowledge from being lost, the public model f is transferred to the next federation before local, personalized training. Other federations on the blockchain can download other nodes with trained models for local training. Since public knowledge has been accumulated, local training is optional. The process of the second stage is shown in Fig. 5. When the public model performs poorly on the local validation data, the personalization phase modifies it very little; when the public model's performance on the local validation data is acceptable, the personalization phase modifies it. Mostly modified for better performance.

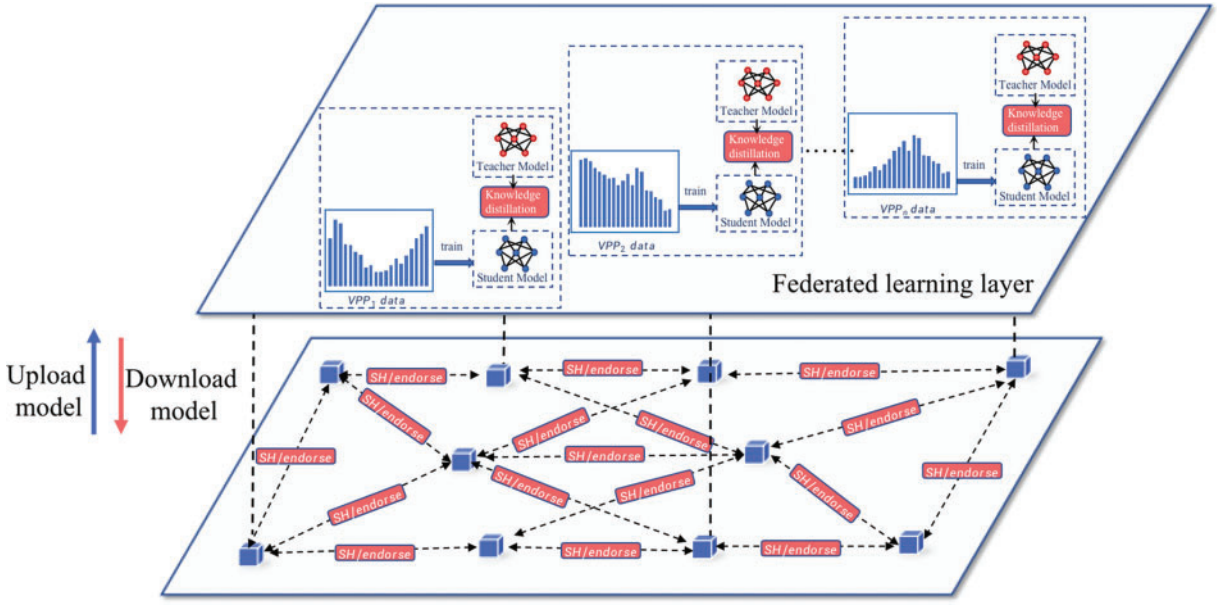


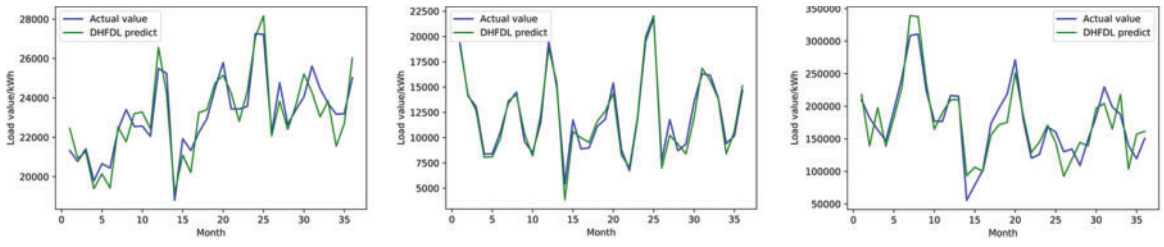
Figure 5: Asynchronous distillation learning aggregation process

4 Experiment

Based on the algorithm proposed in this paper, the load forecasting and analysis simulation experiments on the demand side of the VPP show that the forecasting model can realize the accurate prediction of the demand side compliance and support the VPP to achieve precise regulation and control of layers and partitions. The models are written in Python 3.9.10 and Pytorch 1.11.0 and executed on a Geforce RTX 3080Ti GPU.

In the load forecasting experiment, the dataset contains three types of enterprises the real estate industry, the manufacturing industry, and the catering industry. Each industry includes sample data from 100 companies for 36 consecutive months. The characteristics of each sample are enterprise water consumption, enterprise gas consumption, daily maximum temperature, daily minimum temperature, daily average temperature, daily rainfall, and humidity, and the label is enterprise energy used.

Based on the DHFDL proposed in this paper, the federated load forecasting model is constructed and trained for the data of the three industries. Fig. 6 is the comparison between the prediction effect and the actual value. It can be seen from the figure that the algorithm proposed in this paper has a better prediction effect.



(a) Real estate enterprise load forecast (b) Manufacturing enterprise load forecast (c) Catering enterprise load forecast

Figure 6: Comparison of predicted load values

To prove the effectiveness of blockchain-based decentralized heterogeneous federated distillation learning, this paper conducts experiments on the real-world federated dataset FEMNIST. The dataset contains 805,263 samples, made by 3,550 handwritten character image classification task users, including 62 different categories (10 digits, 26 lowercase letters, 26 uppercase letters), and the number of local datasets is unbalanced. The distributions are not independent. After randomly selecting active nodes, we perform local training and aggregation via memory.

Malicious blockchain nodes participating in FL training will generate harmful local models for malicious attacks. If they participate in model aggregation, the performance of the global model will be significantly reduced. In this section, we simulate malicious node attacks and set different malicious node ratios to demonstrate the impact of different malicious node ratios among participating nodes on the performance of FedAvg, BFLC, and the DHFDL model proposed in this paper. This paper assumes that the malicious attack mode is to randomly perturb the local training model to generate an unusable model. FedAvg performs no defenses and aggregates all local model updates. BFLC relies on committee consensus to resist malicious attacks. During the training process, each model update will get a score from the committee, and the model with a higher score will be selected for aggregation. In the experiment, we assume that malicious nodes are colluding, that is, members of the malicious committee will give random high scores to malicious updates and select nodes with model evaluation scores in the top 20% of each round of training as the committee for the next round. The participating nodes of DHFDL train the local model and select the on-chain model with a model accuracy rate to carry out knowledge distillation in each round of updates to improve the effectiveness of the local model. As shown in Fig. 7, DHFDL can resist a higher proportion of malicious codes than the comparative methods. This shows the effectiveness of DHFDL with the help of knowledge distillation.

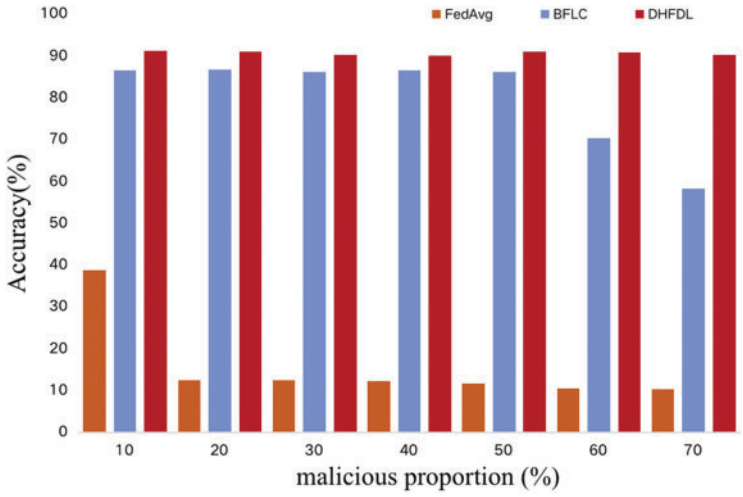


Figure 7: Performance of algorithms under malicious attacks

By combining the asynchronous consensus with the training process of the FL model, the training efficiency of decentralized FL can be improved. As shown in Fig. 8, this paper conducts FL training by setting the number of training nodes participating in FL and counts FedAvg and BFLC. Compared with the training time of each round of the three algorithms of DHFDL, it can be seen that the time required for each round of training of DHFDL, which realizes model asynchronous uplink aggregation through asynchronous consensus, is the lowest. At the same time, as the number

of participating nodes increases, the training of the three algorithms. The time required increases accordingly, but the time required for each training round of the DHFDL algorithm is still the lowest, which shows that DHFDL is efficient with the help of the asynchronous model on-chain.

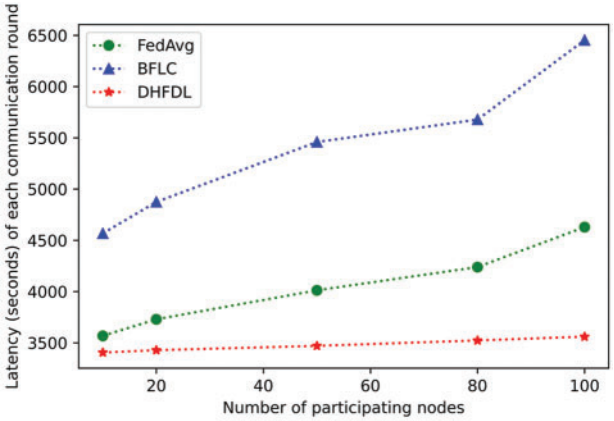


Figure 8: Performance of algorithms in different numbers of participating nodes

Fig. 9 describes the on-chain storage cost comparison of the algorithm. Ten training nodes are simulated to conduct FEMINIST classification model training based on the Convolutional Neural Network (CNN) model, and the on-chain storage cost of the model based on the BFLC and DHFDL algorithms is recorded. DHFDL algorithm that realizes the model chain aggregation through asynchronous consensus requires less on-chain storage overhead to achieve the same accuracy. Meanwhile, with the improvement of accuracy, the two algorithms require more on-chain storage space, but the DHFDL algorithm still has lower storage overhead than the BFLC algorithm.

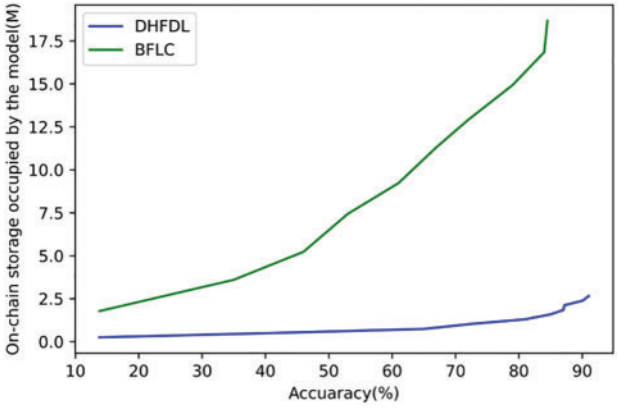


Figure 9: Storage performance of the algorithm

5 Conclusion

In this paper, we propose the DHFDL algorithm, Decentralized Heterogeneous Federated Distillation Learning, to effectively predict the load of virtual power plants for better grid scheduling. DHFDL does not need a central server to organize the federation for training. The public model is extracted through distillation learning, and the model is uploaded to the blockchain. The federation

nodes on the blockchain can download the trained models of other federation nodes to guide personalization training to get a better model. The introduction of blockchain technology enables indicators such as network communication load and FL security to reach better standards. By simulating malicious node attacks and comparing the FedAvg algorithm and the BFLC algorithm, it can be seen that the DHFDL algorithm proposed in this paper can resist a higher proportion of malicious codes. From the comparative experimental results, it can be seen that the combination of asynchronous consensus and FL model training process improves the training efficiency of decentralized FL.

Acknowledgement: I would first like to thank Jiangsu Provincial Electric Power Corporation for providing the experimental environment and necessary equipment and conditions for this research. The strong support from the institution made the experiment possible. I would particularly like to acknowledge my team members, for their wonderful collaboration and patient support. Finally, I could not have completed this dissertation without the support of my friends, who provided stimulating discussions as well as happy distractions to rest my mind outside of my research.

Funding Statement: This work was supported by the Research and application of Power Business Data Security and Trusted Collaborative Sharing Technology Based on Blockchain and Multi-Party Security Computing (J2022057).

Author Contributions: Study conception and design: Hong Zhu; data collection: Lisha Gao; analysis and interpretation of results: Yitian Sha, Shuo Han; draft manuscript preparation: Nan Xiang, Yue Wu. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: Data not available due to ethical restrictions. Due to the nature of this research, participants of this study did not agree for their data to be shared publicly, so supporting data is not available.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] Z. Wei, S. Yu, G. Sun, Y. Sun, Y. Yuan *et al.,* "The concept and development of virtual power plant," *Electric Power System Automation*, vol. 37, no. 13, pp. 1–9, 2013.

[2] Y. Li, C. Chen, N. Liu, H. Huang, Z. Zheng *et al.,* "A blockchain-based decentralized federated learning framework with committee consensus," *IEEE Network*, vol. 35, no. 1, pp. 234–241, 2021.

[3] B. McMahan, E. Moore, D. Ramage, S. Hampson, B. Agüera *et al.,* "Communication-efficient learning of deep networks from decentralized data," in *Proc. of AISTATS*, Washington, USA, pp. 1273–1282, 2017.

[4] T. Yu, E. Bagdasaryan and V. Shmatikov, "Salvaging federated learning by local adaptation," arXiv:2002.04758, vol. 2020, no. 4758, pp. 1–11, 2020.

[5] M. Idriss, I. Berrada and G. Noubir, "FEDBS: Learning on Non-IID data in federated learning using batch normalization," in *Proc. of ICTAI*, Washington, USA, pp. 861–867, 2021.

[6] S. Liu, X. Feng and H. Zheng, "Overcoming forgetting in local adaptation of federated learning model," in *Proc. of PAKDD*, Chengdu, China, pp. 613–625, 2022.

[7] F. Sattler, S. Wiedemann, K. R. Müller and W. Samek, "Robust and communication-efficient federated learning from non-i.i.d. data," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 9, pp. 3400–3413, 2019.

[8] C. He, S. Li, J. So, X. Zeng, M. Zhang *et al.,* "FedML: A research library and benchmark for federated machine learning," arXiv:2007.13518, pp. 1–8, 2020. [Online]. Available: https://doi.org/10.48550/arXiv.2007.13518

[9] H. Yu, Z. Liu, Y. Liu, T. Chen, M. Cong *et al.,* "A sustainable incentive scheme for federated learning," *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 58–69, 2020.

[10] B. Berlo, A. Saeed and T. Ozcelebi, "Towards federated unsupervised representation learning," in *Proc. of the Third ACM Int. Workshop on Edge Systems, Analytics and Networking*, New York, USA, pp. 31–36, 2020.

[11] L. Zhu, Z. Liu and S. Han, "Deep leakage from gradients," in *Proc. of NIPS*, Vancouver, Canada, pp. 14774–14784, 2019.

[12] T. U. Islam, R. Ghasemi and N. Mohammed, "Privacy-preserving federated learning model for healthcare data," in *Proc. of CCWC*, Las Vegas, USA, pp. 281–287, 2022.

[13] O. Choudhury, A. Gkoulalas, T. Salonidis, I. Sylla, Y. Park *et al.,* "Differential privacy-enabled federated learning for sensitive health data," arXiv:1910.02578, pp. 1–6, 2019. [Online]. Available: https://doi.org/10.48550/arXiv.1910.02578

[14] Y. Dong, W. Hou, X. Chen and S. Zeng, "Efficient and secure federated learning based on secret sharing and gradient selection," *Computer Research and Development*, vol. 57, no. 10, pp. 2241–2250, 2020.

[15] P. Chen, S. Liu, H. Zhao and J. Jia, "Distilling knowledge via knowledge review," in *Proc. of CVPR*, pp. 5008–5017, 2021.

[16] C. Liu, S. Guo, S. Guo, Y. Yan, X. Qiu *et al.,* "LTSM: Lightweight and trusted sharing mechanism of IoT data in smart city," *IEEE Internet of Things Journal*, vol. 9, no. 7, pp. 5080–5093, 2022.

[17] H. Kim, J. Park, M. Bennis and S. L. Kim, "Blockchained on-device federated learning," *IEEE Communications Letters*, vol. 24, no. 6, pp. 1279–1283, 2020.

[18] M. Cao, L. Zhang and B. Cao, "Towards on-device federated learning: A direct acyclic graph-based blockchain approach," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 4, pp. 2028–2042, 2021.

[19] Z. Peng, J. Xu, X. Chu, S. Gao, Y. Yao *et al.,* "VFChain: Enabling verifiable and auditable federated learning via blockchain systems," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 1, pp. 173–186, 2021.

[20] Z. Peng, J. Xu, H. Hu and L. Chen, "BlockShare: A blockchain empowered system for privacy-preserving verifiable data sharing," *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, vol. 1, pp. 14–24, 2022.

[21] H. Wu, Z. Peng, S. Guo, Y. Yang and B. Xiao, "VQL: Efficient and verifiable cloud query services for blockchain systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 6, pp. 1393–1406, 2021.