GFL: A Decentralized Federated Learning Framework Based On Blockchain

Yifan Hu [1] **Wei Xia** [2] **Jun Xiao** [3] **Chao Wu** [4]

Abstract

Due to people's emerging concern about data privacy, federated learning(FL) is currently being widely used. Conventional federated learning uses a highly centralized architecture, but in a real federated learning scenario, due to the highly distributed of data nodes and the existence of malicious data nodes, It is of great challenges for conventional federated learning to improve the utilization of network bandwidth and maintained the security and robustness of federated learning under malicious node attacks. In this paper, we propose an innovative Ring decentralized federated learning algorithm(RDFL) that not only makes full use of the bandwidth of the network but also improves the security and robustness of federated learning under malicious node attacks. At the same time, we encapsulated RDFL into a blochain-based federated learning framework called Galaxy Federated Learning framework(GFL) and used real data to perform experiments on the GFL to verify the effectiveness of the GFL.

1. Introduction

Nowadays, data privacy is more valued(Horvitz & Mulligan, 2015), in order to solve the data privacy problem in the process of deep learning, Google has proposed Federated Learning 2016(Konečný et al., 2016). It enables training models without data being pushed to the server, so federated learning can effectively protect data privacy.

In the conventional federated learning topology, a centralized server node is used to collect updates uploaded by numerous data providers and perform aggregation. The update can be model parameters or gradients. When the aggregation is completed, a new global model will be obtained. Then, the server needs to broadcast the global model to all data providers participating in the federated learning. We can know that when the model is larger, the update will also become larger, which will put a huge pressure on the network bandwidth and there will be a lot of time waiting for the data providers after uploading the data, so the bandwidth utilization is very low. On the other hand, from the perspective of internal attacks and external attacks, when it is attacked by internal malicious nodes, conventional federated learning cannot distinguish updates from malicious nodes. These malicious updates will be aggregated as normal updates, making the global model unable to converge. When the centralized server receives an external attack, the entire federated learning process will be terminated. Therefore, the conventional federated learning system lacks robustness and security.

A direct way to solve the robustness problem caused by the centralization problem is to remove the centralized server node and transfer the aggregation work to the data providers. There are also many studies on decentralized federated learning algorithms in academia. Hu *et al.*(Hu C, 2019) proposed a model segmentation method and a decentralized federated learning algorithm based on model segmentation and gossip protocol to maximize the substitution between the two and improve the convergence performance. Li *et al.*(Li Y, 2020) proposed a blockchain-based decentralized federated learning algorithm framework to reduce the impact of internal malicious node attacks on federated learning. Roy *et al.*(Roy A G, 2019) proposed a peer-to-peer decentralized federated learning framework for medical scenarios and showed a high degree of dynamics in a peer-to-peer environment.

Through the literature listed above, we can find that the introduction of blockchain into the decentralized federated learning algorithm can naturally solve the problem of decentralized storage. At the same time, the consensus algorithm of the blockchain can also help us solve the problem of external attacks. However, the improving network bandwidth utilization, reducing network transmission overhead and reducing internal malicious node attacks on the convergence of federated learning is still a key challenge in blockchain-based decentralized federated learning.

In this paper, we propose an innovative blockchain-based decentralized federated learning framework GFL[1]. In GFL, we use IPFS and blockchain to form a decentralized federated learning storage system and introduce an encryption mechanism to improve transmission efficiency and ensure

[1] https://github.com/GalaxyLearning/GFL

the security of transmitted information. In addition, we introduce Ring-allreduce algorithm in distributed machine learning, consistent hash algorithm(Lamping J, 2014) and knowledge distillation(Hinton G, 2015) to propose Ring decentralized federated learning algorithm(RDFL) which improves the network bandwidth while making the decentralized federated learning more effective in internal malicious node attacks and non-iid data.

The main contributions of the paper:

- We introduce IPFS and blockchain to build our decentralized storage system to fix the overhead of each network communication in federated learning at 46 bytes. Therefore, the efficiency of network communication can be greatly improved.
- We borrow the Ring-allreduce algorithm in distributed training and modified it to improve the utilization of network bandwidth in decentralized federated learning.
- Based on the above two points, we proposed an innovative decentralized federated learning algorithm RDFL and encapsulated it into a decentralized federated learning framework GFL.

2. Related Work

There are many decentralized distributed machine learning algorithms in conventional distributed machine learning. Baidu first proposed the decentralized machine learning algorithm Ring-allreduce[2]. In Ring-allreduce, each worker transmits its own gradient to next worker in a ring so that each worker finally has the gradient of the rest of the workers. The Ring-allreduce algorithm improves the utilization of network bandwidth, but as the size of the worker node increases, the number of hops required for Ring-allreduce synchronization will also increase linearly, so the Ring-allreduce algorithm will consume a lot of time on the synchronization gradient. There is also a gossip mechanism different from the hop mechanism adopted by Ring-allreduce. Blot *et al.* (Blot M, 2016) first introduce gossip protocol in deep learning. In gossip deep learning, each worker sends gradients to some workers and the workers that receive these gradients send these gradients to other workers. The distributed machine learning using the gossip mechanism also makes full use of bandwidth and has lower synchronization time than Ring-allreduce mechanism.

In federated learning, decentralized federated learning algorithms can draw inspiration from decentralized distributed machine learning algorithms. Hu *et al.*(Hu C, 2019) took inspiration from the distributed machine learning mechanism of gossip protocol and modified it. Each worker uses the gossip protocol to transmit the model segmentation so that decentralized federated learning can improve the utilization of network bandwidth. However, each node in gossip protocol will send a large number of the same request at the same time which will cause additional communication overhead and even block the communication channel.

It is worth noting that due to the lack of consensus among workers in the decentralized federated learning algorithm, it is easy to be attacked by external malicious nodes through forged gradient. Blockchain is a decentralized storage system. Due to the existence of consensus mechanism between blockchain nodes, using blockchain as a storage system for decentralized federated learning is one of the ideal choices. In fact, there are many decentralized federated learning algorithms based on blockchain. Li *et al.*(Li Y, 2020) proposed a blockchain-based decentralized federated learning algorithm called BFLC which reduces the impact of internal malicious nodes forging gradient attacks by selecting some trusted nodes to form a committee to validate gradients. Shayan(Shayan M, 2018) proposed a fully decentralized peer to peer (P2P) approach to multi-party ML based on blockchain called Biscotti. Biscott has a relatively good performance on scalable, fault toler- ant, and defends against known attacks.

Blockchain-based decentralized federated learning has indeed made effective progress in recent years, but there are still some problems need to be solved.

- *Communication pressure* When the federated learning model is relatively large, if the model parameters or gradients are directly transmitted, it will bring huge network communication pressure.
- *Communication efficiency*At present, the bandwidth utilization of worker nodes in decentralized federated learning is not high, and only upstream or downstream bandwidth is used in each federated learning stage. On the other hand, workers have to wait for each other after completing the transmission, resulting in a waste of bandwidth.
- *Model security* The processing of internal malicious node attacks is not complete enough and there is no processing for the aggregation abnormality caused by the different gradient structure in the case of using heterogeneous models for malicious nodes.
- *Aggregation performance* In addition, for non-independent and identically distributed(non-IID) data, the existing work does not propose an effective way to aggregate updates from different data distribution nodes.

[2]https://andrew.gibiansky.com/blog/machine-learning/baidu-allreduce/

In the following section, we will explain how to solve this series of problems in GFL.

3. The Proposed Framework

The storage system of GFL and RDFL algorithm is the core of GFL's decentralized federated learning. Therefore, we first describe the details of the storage system and RDFL algorithm in this section. Then, we will describe some implementation details of GFL. It is worth noting that while describing the storage system and RDFL algorithm, some GFL components will be involved. We will briefly describe the functions of these components first and the details of the components will be described in detail in the implementation of GFL.

3.1. Storage System

GFL's storage system consists of two parts, blockchain and IPFS. GFL uses Ethereum(G, 2014) as the implementation of the blockchain. The reason for using Ethereum is that Ethereum is currently one of the most popular open-source blockchain platforms, which provides good interface support and has an active community. The blockchain mentioned in the paper is Ethereum by default. Every participating node in GFL, whether it is a data provider or a model provider, is a node in the blockchain. The node will call the smart contract deployed in the blockchain to store the model parameters in the blockchain. Specifically how to store model parameters in the blockchain, Li *et al.*(Li Y, 2020) has explained in detail in the section *Blockchain storage*.

However, as we mentioned before, if the model parameters are stored directly in the blockchain, when the model parameter file is large, it will bring huge communication pressure, consensus and storage pressure between blockchain nodes. Therefore, we introduce IPFS in GFL to solve this problem. IPFS(J, 2014) is a decentralized file storage system based on blockchain. Uploading a file to IPFS will get a hash fixed to 46 bytes, which can be used to obtain the corresponding file from IPFS. In GFL, each node is a node of the blockchain and also a node of IPFS. When a node wants to transmit model parameters, it first uploads the model parameter file to IPFS to obtain the corresponding 46-byte IPFS hash and then uploads the IPFS hash to the blockchain for storage. In this way, each communication overhead and storage overhead can be reduced to 46 bytes, which greatly reduces communication pressure, blockchain consensus pressure and storage pressure.

3.2. Ring Decentralized Federated Learning(RDFL)

Now, we assume that there are six data providers, these nodes are respectively labeled as A, B, C, D, E, F, where B, C, E are untrusted nodes, A, F, D are trusted nodes.

According to the principle of the consistent hash algorithm(Lamping J, 2014), these six nodes calculate the consistent hash value according to their ip and surround them into a ring according to the hash value from small to large. The entire ring structure is shown in Fig.1 and the untrusted nodes B, C, E have been marked in gray.

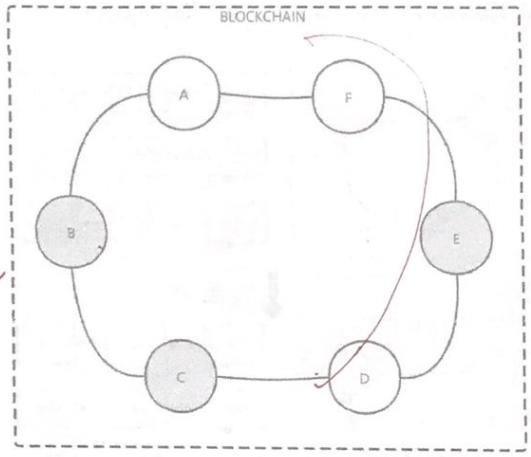

Figure 1: Ring topology

When each node completes local training, the node will start the synchronization operation. For the convenience of illustration, we assume that all nodes complete local training at the same time and default the model parameters described below to be the IPFS hash corresponding to the model parameter file. In addition, for the convenience of explanation, we focus on describing the RDFL algorithm itself and omit the encryption step and the call of the blockchain smart contract. These two parts will be supplemented later.

We mark the local model parameters of A, B, C, D, E, F as MA, MB, MC, MD, ME, MF. According to the principle of the consistent hash algorithm, the untrusted node will send its model parameters to the first trusted node encountered in a clockwise direction. For example, B, C will send the model parameters MB, MC to the trusted node D, and ME will send it to the trusted node F. If trusted node D is down, B and C will find trusted node F in a clockwise direction and then send their model parameters to F. By using the consistent hash algorithm, we transfer the model parameters of untrusted nodes to different trusted nodes, reducing the communication pressure and when a node is down, the node can automatically adjust the transmission object to improve decentralized federated learning Topological stability.

For trusted nodes A, D and F, we use an algorithm similar to Ring-allreduce[3], and each trusted node passes the model parameters to the next trusted node in a clockwise order. Therefore, combined with the above-mentioned consistent hash algorithm, the first round of model parameter transfer is as folows: model parameters MA, MB, MC are passed to node D, model parameters MD, ME are passed to node F, model parameters MF is passed to node A. At the end of the first cycle, model parameters owned by each trusted node hash shown in the first cycle diagram of Fig.2.

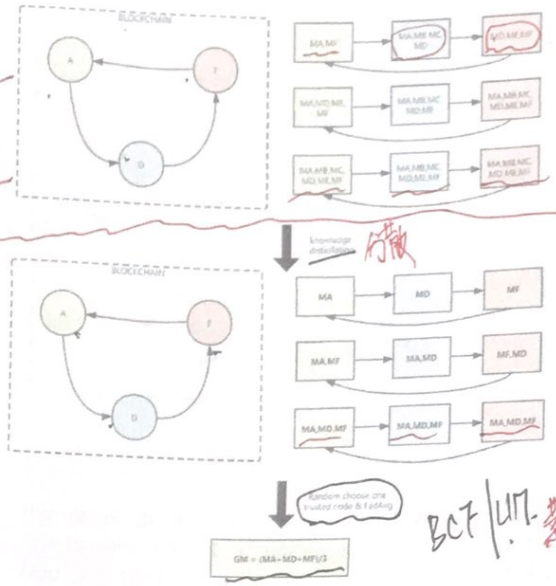

Figure 2: Ring Decentralized Federated Learning

In the second round, the trusted nodes A, D and F transfer their own model parameters to the next trusted node in a clockwise direction. The next trusted node obtains its missing model parameters from the obtained model parameters. The second round of model parameter transfer is as follows: the model parameters MA and ME owned by node A transferred to node D, the model parameters MA, MB, MC and MD owned by node D are transferred to node F, model parameter MD, ME and MF owned by node F are passed to node A. At the end of the second cycle, the model parameters owned by each trusted node are shown in the second cycle diagram of Fig.2.

Similar to the second round, the trusted nodes transfer their own model parameters clockwise. At the end of the third cycle, the model parameters owned by each trusted node are shown in the third cycle diagram of Fig.2. Each trusted node has the model parameters of all nodes and reaches a stable state. By introducing the Ring-allreduce algorithm into decentralized federated learning, the bandwidth utilization of synchronization model parameters between nodes is maximized and thanks to IPFS, the pressure of each communication between nodes is also reduced to a minimum.

When all trusted nodes reach a stable state, we use knowledge distillation to transfer the knowledge contained in the model parameters received by each trusted node to its locally trained model parameters. Knowledge distillation(Hinton G, 2015) is a method of knowledge transfer proposed by Hinton in 2015. Through knowledge distillation, the dark knowledge in a powerful teacher model can be transferred to the student model with little knowledge. The loss function $L_{student}$ of the student model is shown in formula (1).

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

$$p_{\text{teacher}} = \frac{\exp(z/T)}{\sum_i \exp\left(z_i/T\right)} \tag{2}$$

L_{CE} and D_{KL} respectively represent cross entropy and Kullback Leibler(KL) divergence. $P_{student}$ and $P_{teacher}$ represent the output of the model after the softmax activation function. z represents the logits output by the model and T is a hyper-parameter representing the temperature of distillation. Knowledge distillation minimizes $L_{student}$ to allow the student model to learn dark knowledge. In RDFL, we use Euclidean distance instead of KL divergence to measure the difference in distribution between models. Therefore, $L_{student}$ of the student model in RDFL is shown in formula (3). D_{ED} represents Euclidean distance(ED). $z_{student}$ and $z_{teacher}$ represent the logits of the model.

$$L_{\text{student}} = L_{CE} + D_{ED}\left(z_{\text{teacher}} \| z_{\text{student}}\right) \tag{3}$$

In RDFL, when all trusted nodes reach a stable state, all trusted nodes use their data to start distillation. But in order to prevent malicious nodes from using malicious data to forge model parameters before distillation, we select model parameters with a smaller Euclidean distance from the model parameters of trusted nodes for distillation. For example, in trusted node A, A will use MA as the student model, then select the remaining MB, MC, MD, ME, MF to calculate the Euclidean distance, and then select the 30% model parameter with the smallest Euclidean distance as the teacher model. Then, The knowledge of this 30% teacher model is transferred to the MA model through knowledge distillation. At the same time, as the number of aggregation rounds increases,

[3]https://andrew.gibiansky.com/blog/machine-learning/baidu-allreduce

the proportion of selected teacher models will be dynamically increased to improve the generalization performance of the final aggregation model.

When the trusted nodes complete the knowledge distillation, they will directly execute the Ring-allreduce algorithm again to synchronize the model parameters after distillation. As shown in Fig.2, after synchronization is completed, all trusted nodes have the model parameters of each trusted node after this round of distillation. Then we randomly select a trusted node to execute the FedAvg algorithm to obtain the final aggregation parameter GM of this round. In Fig.2, we randomly select the trusted node A to perform the FedAvg operation. Finally, we send the GM to all trusted and untrusted nodes to start the next round of the cycle.

It should be noted again that in the above description process, in order to describe the algorithm itself more clearly, we use model parameters to replace the IPFS hash of model parameters. As for the encryption process and the invocation process of the blockchain smart contract, we will explain below.

3.3. Encryption Mechanism and Smart Contract

For the convenience of description, we also use model parameters to replace the IPFS hash of model parameters.

GFL uses a combination of asymmetric encryption and symmetric encryption to encrypt every piece of information stored on the blockchain. Every trusted node and untrusted node owns the public key and private key of the asymmetric encryption algorithm. The trusted node will additionally possess the secret key of the symmetric encryption algorithm. When the untrusted node finds the first trusted node it encounters in a clockwise direction, it tells the trusted node its own public key. The trusted node encrypts the symmetrically encrypted secret key with the public key of the untrusted node and sends it to the untrusted node. Then the untrusted node and the trusted node can use symmetric encryption to encrypt the transmitted information and can boldly save it on the blockchain. The communication encryption between trusted nodes is the same principle as the communication encryption between trusted nodes and untrusted nodes.

Since GFL is a decentralized federated learning framework based on the blockchain, in fact, almost every step of the RDFL algorithm is implemented by calling smart contracts on the Ethereum. For example: the untrusted node selects the first trusted node encountered by clockwise, the transmission of the secret key and the transmission of the model parameters, etc. In Ethereum, as long as the smart contract is called to modify the current state of the blockchain, a new transaction information will be generated. This transaction information needs to pass the consensus between the nodes to be packaged into the block and permanent save in the blockchain. Since Ethereum uses the PoW consensus algorithm, the computing power of malicious nodes needs to reach 51% of the total computing power of GFL nodes to successfully attack which is almost non-existent in reality.

Through the analysis of the above modules, we can know that GFL has solved the following problems:

- *Communication pressure* GFL uses IPFS to transmit the 46-byte IPFS hash of the model parameters instead of the model parameters, which greatly reduces the communication pressure. In addition, RDFL uses a model parameter synchronization method similar to Ring-allreduce, which reduces the communication congestion problem that often occurs in the gossip method.

- *Communication efficiency*The RDFL decentralized federated learning algorithm used by GFL uses an algorithm similar to Ring-allreduce, which is a bandwidth-optimal way to do an allreduce.

- *Model security* The RDFL decentralized federated learning algorithm used by GFL reduces the influence of malicious data on malicious nodes on the premise of improving the generalization of the aggregate model as much as possible. In addition, RDFL will not allow untrusted nodes to access model parameters from other nodes. More importantly, RDFL encrypts the IPFS hash of the model parameters to avoid exposure on the information blockchain and protect the information security as much as possible.

- *Aggregation performance* GFL uses the method of knowledge distillation and dynamic polymerization ratio to improve the generalization of the model while ensuring the safety of the model as much as possible. For non-iid data, GFL uses knowledge distillation as an aggregation method to have better accuracy than FedAvg in most cases and supports aggregation between heterogeneous models.

3.4. Implementation

The main actors in the system are shown in Figure 1, Each parameter in the picture is as follows, n represents the number of Model Providers, m represents the number of data providers, and k is the total number of jobs.

Data Provider provides data for federated learning and participates in training.

Model Provider is responsible for building models and generating federated learning job files. Since each model

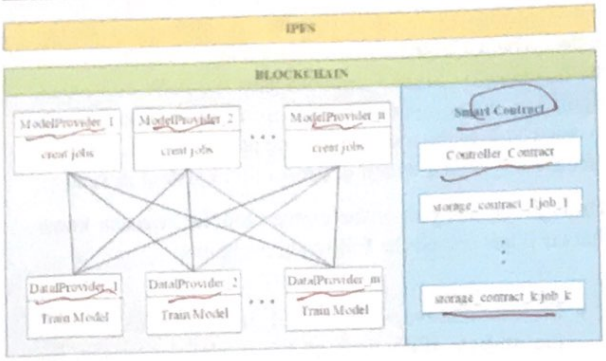

Figure 3: Architecture of GFL

provider may have more than one model, a model provider can initiate multiple federated learning jobs.

Storage Smart Contract is responsible for storing the IPFS hash of temporary files in the federated learning task, such as the IPFS hash of the model parameter file, the IPFS hash of the global model parameter file, etc. In addition, the storage contract is also responsible for some functions such as finding trusted nodes in the consistent hash algorithm.

Controller Smart Contract is used to manage federated learning , such as managing the life cycle of federated learning jobs, checking the current status of federated learning jobs, managing nodes in GFL and selecting aggregation nodes, etc.

In GFL, both the data provider and the model provider are the nodes that constitute the blockchain, and the data provider is the node that constitutes the IPFS. When the model provider creates a GFL federated learning job, GFL will first upload the job file to IPFS to obtain the corresponding 46-byte IPFS Hash and then record the IPFS Hash, jobId, and description of the job in controller smart contract. In addition, GFL will create a storage contract corresponding to the job. After that, different data providers will select the IPFS Hash of the federated learning job from controller smart contract according to the description information of the job, then download the corresponding job file from IPFS and execute the RDFL algorithm. All temporary files generated in the process of executing the RDFL algorithm, such as temporary model parameter files, will be upload to IPFS first and by calling the function in the storage contract corresponding to this federated learning job, the IPFS hash of the temporary model parameter file is stored in the blockchain.

In a real situation, some data providers as one of the nodes in the blockchain are likely to have many unexpected situations due to certain factors, such as power outages, unexpected exits, and so on. In a conventional centralized environment, the central server may also have a similar situation.

For this reason, in GFL, each node on the blockchain needs to run a monitoring mechanism to monitor the information of the surviving nodes on the smart contract. Since GFL uses a consistent hash algorithm, all nodes are arranged in a ring topology, so each node only needs to periodically check the status of the next node clockwise, which effectively reduces the communication overhead required for monitoring. If the next node goes down, the controller contract only needs to remove the node from the ring. At the specific implementation level,this ring topology is a list of consistent hashes of all nodes in the controller contract. The controller contract only needs to remove the consistent hash of the down node from the list. In addition, when the trusted node executing FedAvg goes down, thanks to the RDFL algorithm, the remaining trusted nodes also have the same model parameters, so the controller contract only needs to randomly select a trusted node to execute FedAvg again.

Guaranteed by the RDFL, consistent hash algorithm, the monitoring mechanism and the blockchain smart contract, GFL minimize the impact of power outages, unexpected exits and malicious node attacks. Because of the information required for decentralized federated learning is stored in the blockchain, so even if a large number of nodes are down. The remaining nodes can still continue federated learning. The only effect may be that the smaller the size of the data will affect the accuracy of the final model.

4. Experiments

The GFL github official website provides demos of handwriting recognition and image classification. The data set used for handwriting recognition is MNIST (Deng, 2012) and the model is a double-layer convolution network. The data set for image classification is CIFAR-10 (Li et al., 2017) and the model is also a double-layer convolution network. In both demos, the accuracy of using GFL for decentralized federated learning has reached the baseline.

4.1. MNIST

The MNIST data set comes from the National Institute of Standards and Technology (NIST). The training set is composed of 250 handwritten numbers from different people, 50% of which are high school students and 50% are from the Census Bureau staff. The test set is also the same proportion of Handwritten digital data.

4.2. CIFAR-10

CIFAR-10 is a small data set for identifying universal objects. A total of 10 categories of RGB color pictures: aircraft ,automobile, bird, cat, deer, dog, frog, horse, ship, and

Figure 4: MNIST experiment result accuracy

truck. The size of the picture is 32×32. There are 50,000 training pictures and 10,000 test pictures in the data set.

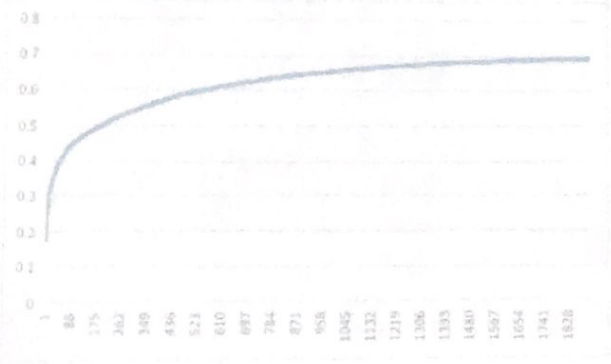

Figure 5: CIFAR-10 experiment result accuracy

4.3. Implementation Details and Result Analysis

The experiments are conducted on a machine with 1.3 GHz Intel Core i5 with memory 4 GB 1600 MHz DDR3. We implement each private network architecture with convolutional neural network (CNN) model.For CIFAR-10 data set we us model(with 2 convolutional layers with 3 filters each followed by a max pooling layer,then 2 more convolutional layers with 3 filters each followed by another max pooling layer and a dense layer with 512 units, ReLU activation is used in dense layer) to validate the effectiveness of our framework, and for MNIST data set we use model(with a convolutional layer with 5 filters each followed by a max pooling layer,then another convolutional layers with 5 filters each followed by max pooling layer and a dense layer with 500 units, ReLU activation is used in dense layer).A mini-batch size of 128 and learning rate 0.01 are used in these experiments. We adopt our framework for the Horizontal Federated Learning (HFL), and the experimental results are shown in Figures 4 and Figures 5. Experiments show that our framework has excellent performance on both data sets.

5. Conclusion

In this paper, we propose and implement a decentralized federated learning framework based on blockchain called GFL to solve the problems of centralization of federated learning. GFL uses consistent hash algorithm, RDFL algorithm, IPFS and blockchain to solve the problems of poor robustness, high communication pressure, low bandwidth utilization and cannot protect model security and communication security in conventional blockchain-based decentralized federated learning. In addition, thanks to the knowledge distillation algorithm, GFL can get better results in the case of non-iid data and model heterogeneity between nodes.

6. Future Work

Vertical Federated Learning The decentralized federated learning mechanism proposed by GFL can theoretically be applied to vertical federated learning. In the future, we will test vertical federated learning and provide APIs related to vertical federated learning.

Selection of aggregation nodes For the selection of data providers in the GFL's Selection Decentralization Federated Learning Mechanism, GFL currently use the amount of data as the only indicator for selection. But in the future, GFL will add factors such as the number of computing resources of the data provider, current power, network conditions and other factors and comprehensively score according to a certain weight and preferentially select the data provider with the highest score as the node for model aggregation node.

Model compression At present, for some models with many model parameters, the process of uploading large model parameter files to IPFS will take a lot of time, so it will make the GFL decentralized federated learning mechanism spend a lot of time on uploading and downloading files. Therefore, GFL will use methods such as model quantization and model distillation implemented in GFL in the future to compress some model parameters within an acceptable range to greatly improve the time of GFL decentralized federated learning mechanism.

References

Galaxy federated learning. https://github.com GalaxyLearning/GFL.

Blot M, Picard D, C. M. e. a. Gossip training for de learning. *arXiv preprint arXiv:1611.09726*, 2016.

Deng, L. The mnist database of handwritten digit imag