Training Aggregation in Federated Learning

Li Hu, Hongyang Yan(✉), and Zulong Zhang

School of Artificial Intelligence and Blockchain, Guangzhou University, Guangzhou, People's Republic of China

Abstract. Federated learning is a new machine learning paradigm for distributed data. It enables multi-party cooperation to train global models without sharing their private data. In the classic federated learning protocol, the model parameters are the interaction information between the client and the server. The client can update the local model according to the global model parameters, and the server can aggregate the updated model parameters of each client to obtain a new aggregation model. However, in the actual federated learning scenario, there are still privacy problems caused by model stealing attack in collaborative learning using model parameters as interactive information. Therefore, we use knowledge distillation technology to avoid the model stealing attack in federated learning, and on this basis, we propose a novel aggregation scheme, which can make the output distribution of each customer refine the aggregation results through model training. Experiments show that the scheme can achieve normal convergence while ensuring privacy security, and reduce the number of interactions between client and server, thus reducing the resource consumption of each client participating in federated learning.

Keywords: Collaborative learning · Knowledge distillation · Heterogeneous · Communication

1 Introduction

Federated learning [1–3] is a collaborative machine learning algorithm for Non-IID data [4,5], which enables multiple parties to collaboratively train a shared global model without sharing data. At the beginning of each round of training, the central server will transmit the current global model to all parties, and all parties will train the model on local data. Then the server will collect model update information from all parties and update the central global model.

The main purpose of participating in federated learning is to obtain better models without sharing local data. The prototypical federated learning architecture [2,6] uses model parameters as the interactive information between the clients and the server, and then the server performs weighted average aggregation to update the global model. However, there are three problems with the model parameters as the interactive information:

J. Cheng et al. (Eds.): CSS 2020, LNCS 12653, pp. 87–94, 2021.
https://doi.org/10.1007/978-3-030-73671-2_9

1. When model parameters are used as interactive information, malicious clients may steal model parameters from victims, which may lead to privacy leakage.
2. The model structure trained by each client must be the same to be aggregated. However, in real scenarios, the agents participating in federated learning may have different training model structures due to different requirements [7,8].
3. When there are many users participating in federated training, the required communication will increase sharply, because each user has many model parameters, which will affect the efficiency of federated aggregation.

In response to the above problems, existing related articles [9–11] propose to use the output of the model as interactive information. However, articles [9,10] both directly aggregate the logit value by weighted average, while the article [11] designs an robust mean estimation algorithm to achieve aggregation, but none of these methods of aggregation explains whether the final aggregation result is the optimal result. Inspired by these solutions, this paper proposes to use knowledge distillation technology [12–14] and let the logit value output by the client aggregate by itself, that is, train the model on the server to aggregate the updated value uploaded by each client. Experiments have proved that this aggregation method can also achieve high convergence accuracy. At the same time, the number of interactions with the client is greatly reduced for its convergence, which will significantly reduce the amount of communication in the federated learning scenario. And it can explain from the perspective of knowledge distillation that each client participating in the training can obtain better results.

Figure 1 gives a high-level overview of this scheme. The model of each client is different. What is uploaded is the prediction distribution of the public data set. A model is trained on the server to aggregate the prediction distribution uploaded by the client. The generated model then predicts the public data set, and the prediction results broadcast to each client updates its local model, and so on, until the average accuracy of each client model converges, and then stops federated learning.

The main contributions of this work are summarized as follows:

- We design a collaborative learning approach with training for aggregation to achieve federated learning with heterogeneous model architectures.
- We further verify federated learning aggregation by training (FLAT) can improve the computation efficiency and reduce communication overhead.

Organization. The remainder of this paper is organized as follows. In Sect. 2, we present our detailed designs of mutual federated learning framework. The system evaluation and experimental results are presented in Sect. 3. Section 4 concludes this paper.

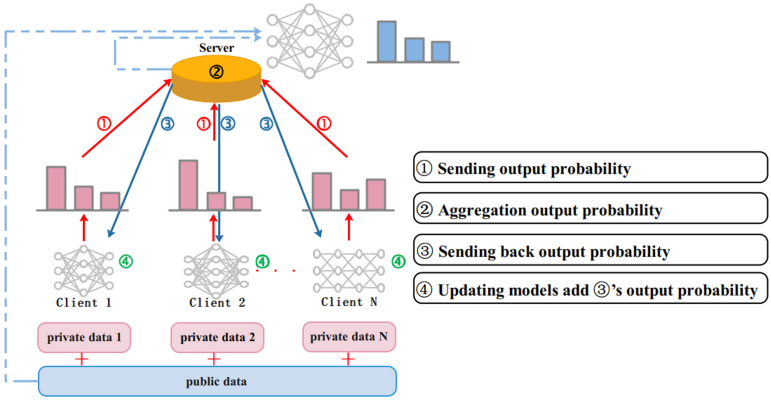


Fig. 1. The illustration of FLAT framework.

2 The Detail for Federated Learning Aggregation by Training

In this section we will introduce the details of our proposal. Algorithm 1 describes the process (called FLAT). We learn from the idea of knowledge distillation [12,15] and use the output of the model as the interactive information between the client and the server to achieve federated learning.

Each client initializes its own model parameter $\theta_k^0 (k \in \{1, 2, ...n\})$, and the models are heterogeneous between clients, that is, they have different model structures and sizes. User k has a dataset D_k containing N_k samples $X_k = \{x_{ki}\}_{i=1}^{N_k}$, of the M class, with the corresponding label set of $Y_k = \{y_{ki}\}_{i=1}^{N_k}, y_{ki} \in \{1, 2, ..., M\}$, at the same time, each client has a dataset D containing N samples $X = \{x_i\}_{i=1}^{N}$, of the M class, with the corresponding label set of $Y = \{y_i\}_{i=1}^{N}, y_i \in \{1, 2, ..., M\}$. In the first round, each client updates the model parameters based on the local datasets D_k and D, and obtains $\theta_k^1 = \theta_k^0 - \bigtriangledown L(D_k \bigcup D; \theta_k^0)$, where $L(D_k \bigcup D; \theta_k^0)$ is about θ_k^0 loss function on the dataset $D_k \bigcup D$, and α is the learning rate. Each party then predicts dataset D based on locally updated parameters and uploads the prediction $\overline{Y}_k^1 = f(\theta_k^1, X)$ to the server, $f(\theta_k^1, X)$ represents the probability distribution of input X passing through the model with parameters θ_k^1. The server aggregates by training a model (model parameter is θ^0) and gets a new predictive distribution $\widetilde{Y}^1$ of public dataset D. Each client obtains $\widetilde{Y}^1$ through the server broadcast, and updates themself model through the dataset $D_k \bigcup D \bigcup (X, \widetilde{Y}^1)$.

3 Experiment Evaluation

The framework FLAT is evaluated based on popular image datasets: MNIST [16], that is a widely used in the FL [17,18]. In the MNIST training dataset with 60,000 samples, and in the MNIST testing dataset with 10,000 samples.

Algorithm 1. Federated Learning Aggregation by Training(FLAT)

Input:

D_k : private dataset for each client k ,$(k \in \{1, 2, ..., n\})$

D : public dataset for all clients and server

α : the learning rate of training

Output:

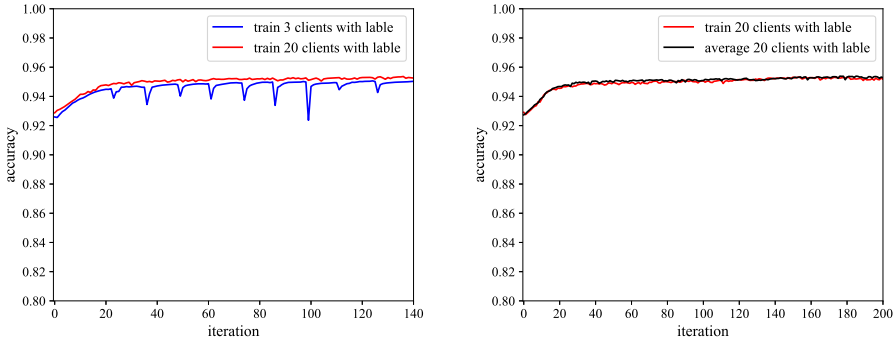
θ_k^t : model parameters for each client k ,$(k \in \{1, 2, ..., n\})$

Initialize :
each client initialize θ_k^0 by themself
server initialize θ^0 by all clients
t = 0
repeat
  Client:
  $\overline{D} = D$
  for each client $k \in \{1, 2, ..., n\}$ **do**
    if $t == 0$ **then**
      $\widetilde{D}_k = D_k \bigcup D$
      $\theta_k^{t+1} = \theta_k^t$ - $\alpha \bigtriangledown L(\widetilde{D}_k; \theta_k^t)$
    else
      $\theta_k^{t+1} = \theta_k^t$ - $\alpha \bigtriangledown L(\widetilde{D}_k; \theta_k^t)$
    end if
    $\overline{Y}_k^t = f(\theta_k^{t+1}, X)$
    send $\overline{Y}_k^t$ to server
    server get dataset $\overline{D} = \overline{D} \bigcup (X, \overline{Y}_k^t)$
  end for

  Server:
  get dataset $\overline{D}$
  $\theta^{t+1} = \theta^t$ - $\alpha \bigtriangledown L(\overline{D}; \theta^t)$
  $\widetilde{Y}^{t+1} = f(\theta^{t+1}, X)$
  send $\widetilde{Y}^{t+1}$ to each client
  client k get dataset $\widetilde{D}_k = D_k \bigcup D \bigcup (X, \widetilde{Y}^{t+1})(k \in \{1, 2, ..., n\})$
until t>100

First, we verify FLAT algrithm in same neural network with two hidden layers. After determining the optimal hyperparameter, we then verify FLAT algrithm in different neural network, which is taken from the tensorflow tutorial [19].

We tested the test accuracy of 3 clients and 20 clients with tagged public data, and their models are the same. As shown in Fig. 2(a), the results were consistent with the original federated learning effect, and the increase in the number of clients would improve the performance of federated learning. In order to verify the effectiveness of our scheme, we compared the results of federated distillation (FD) with 20 clients. In the training process of federated distillation,



(a) Average Accuracy. (b) Average Accuracy.

Fig. 2. The variation trend of Average Accuracy. On the left is the FLAT scheme using tagged public data to compare 3 clients and 20 clients. On the right, 20 clients are trained with labeled public data to compare the FLAT scheme and the FD scheme.

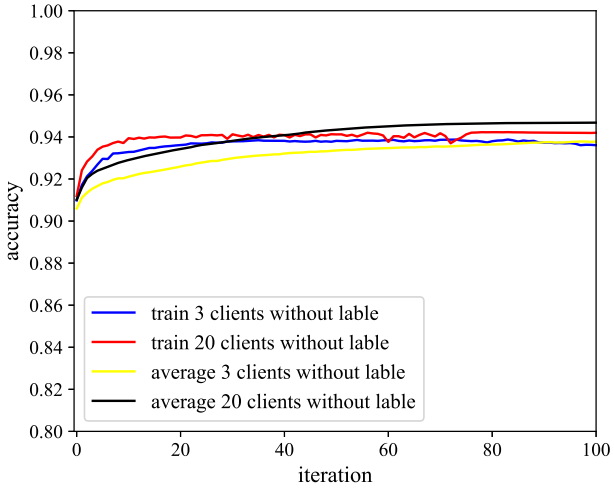


Fig. 3. Convergence behavior about the FLAT scheme and the FD scheme using unlabeled public data to train 3 and 20 clients.

the probability distribution uploaded by each client was weighted averaged, as shown in Fig. 2(b), and we found that we could also achieve the optimal accuracy. And we tested the test accuracy of 3 clients and 20 clients with untagged public data. Each client with 2,000 private data and 10,000 untagged public data, server just with 10,000 untagged public data. We can see Fig. 3, public data with or without tags will have the same effect.

At the same time, we found that when our scheme achieves the optimal accuracy, the number of rounds required for interaction between the client and the server is much less than that of the federated distillation solution. Table 1 lists the specific differences. It can be seen that when the number of clients is 3 and the public data has tags, our solution only needs 62 rounds of interaction to achieve the best test accuracy of 95.19%, while the federated distillation scheme requires 188 rounds of interaction to achieve the best test accuracy of 95.29%. That is to say, at this time, within the range of less accuracy loss, the number of interactions required by our solution is 1/3 of the FD solution, which can greatly reduce the resource consumption of the client participating in federated learning. We also compared the unlabeled cases of the public data sets. The experimental results show that the number of interactions required by our solution is 1/2 of the FD solution when the public data set is not labeled, although the accuracy is more lost when 20 clients participate in federated learning, but the accuracy loss is within an acceptable range.

Table 1. The number of interactions between clients and server when the model reaches the Top-1 accuracy (%) on the mnist dataset. FD - FLAT measures the difference in the number of interactions and accuracy between the FD and FLAT.

Number of clients	With Lable			Without Lable		
	FLAT	FD	FD - FLAT	FLAT	FD	FD - FLAT
3	62(95.19)	188(95.29)	126(0.01)	83(93.88)	170(93.89)	87(0.01)
20	150(95.32)	176(95.38)	26(0.06)	83(94.23)	169(94.72)	86(0.49)

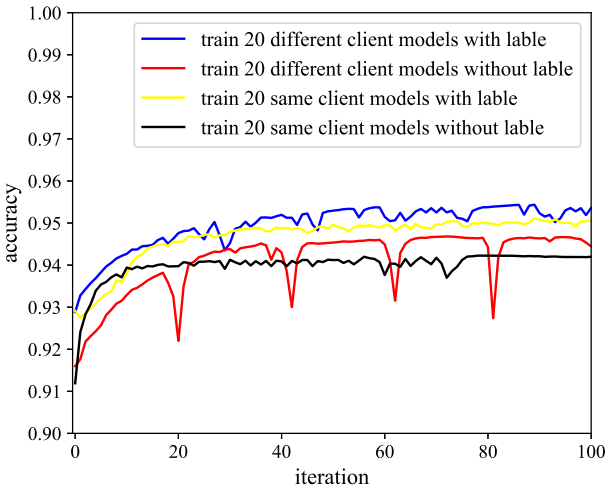


Fig. 4. Use different models to train 3 and 20 clients to compare the convergence behavior of the FLAT scheme with labeled public data and unlabeled public data.

Then we test on different models, so that the number of layers in the model is different (maybe 2–3 layers), and the number of neurons in each layer is different (there may be 32, 64, 128, 256, 512, 1024). After that we use the labeled public data set and the unlabeled public data set to train 20 clients for testing. We found that changing the model structure can still converge normally. As shown in Fig. 4, when the model becomes larger, the accuracy of the model obtained by convergence is also higher.

4 Conclusion

In this paper, we have proposed a novel while effective federated learning framework, targeting to achieve that the user model can be heterogeneous and the privacy of user can be protected, while reducing the user's resource consumption during the federated learning process. Here we learned from the knowledge distillation method of interacting information with model output as knowledge, and further improved the existing federated learning aggregation scheme. Through experiments, we found that FLAT can realize good convergence while greatly reducing the resource consumption of the client.

Acknowledgement. The research leading to these results has received funding from China Postdoctoral Science Foundation (2020M682658).

References

1. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial Intelligence and Statistics, pp. 1273–1282 (2017)
2. McMahan, H.B., Moore, E., Ramage, D., y Arcas, B.A.: Federated learning of deep networks using model averaging
3. Yang, Q., Liu, Y., Chen, T., Tong, Y.: Federated machine learning: concept and applications. ACM Trans. Intell. Syst. Technol. **10**(2), 1–19 (2019)
4. Zhao, Y., Li, M., Lai, L., Suda, N., Chandra, V.: Federated learning with Non-IID data
5. Sattler, F., Wiedemann, S., Müller, K.-R., Samek, W.: Robust and communication-efficient federated learning from Non-IID data
6. Mohri, M., Sivek, G., Suresh, A.T.: Agnostic federated learning. arXiv preprint arXiv:1902.00146
7. Liu, Y., et al.: Fedvision: an online visual object detection platform powered by federated learning. arXiv: Learning
8. Choudhury, O., et al.: Differential privacy-enabled federated learning for sensitive health data. arXiv: Learning
9. Li, D., Wang, J.: FedMD: Heterogenous federated learning via model distillation. arXiv preprint arXiv:1910.03581
10. Jeong, E., Oh, S., Kim, H., Park, J., Bennis, M., Kim, S.-L.: Communication-efficient on-device machine learning: federated distillation and augmentation under Non-IID private data. arXiv preprint arXiv:1811.11479

11. Chang, H., Shejwalkar, V., Shokri, R., Houmansadr, A.: Cronus: robust and heterogeneous collaborative learning with black-box knowledge transfer. arXiv preprint arXiv:1912.11279
12. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531
13. Zhang, Y., Xiang, T., Hospedales, T.M., Lu, H.: Deep mutual learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4320–4328 (2018)
14. Wang, J., Bao, W., Sun, L., Zhu, X., Cao, B., Philip, S.Y.: Private model compression via knowledge distillation. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 1190–1197 (2019)
15. Chen, H.: Data-free learning of student networks. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3514–3522 (2019)
16. Lecun, Y., Bottou, L.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
17. Caldas, S., et al.: Leaf: a benchmark for federated settings
18. Kairouz, P., et al.: Advances and open problems in federated learning. arXiv preprint arXiv:1912.04977
19. Team, T.: Tensorflow convolutional neural networks tutorial. http://www.tensorflow.org/tutorials/deepcnn