Proceedings of the 2nd International Conference on Machine Learning and Automation
DOI: 10.54254/2755-2721/107/2024.18310

Improving Federated Learning Accuracy with the Incremental Averaging Method: A Comparative Analysis of Model Aggregation Techniques

Chenzhang Hu

School of Art and Science, Ohio State University, Columbus, Ohio, United States

hu.2385@buckeyemail.osu.edu

Abstract. Given the increasing use of federated learning in the market and its promising future development prospects, this paper will mainly provide a new model merging method for federated learning called The Incremental Average method. After completing the local model construction using this method, customers can upload the model without waiting. This paper will first introduce the background, applications, and existing advantages and disadvantages of remote federated learning. It will then carefully analyze the current federated learning (FL) frameworks, including the formulas and methods involved. Following this, a new Incremental Averaging method will be presented and compared with the previous methods. The results show that the accuracy of remote federated learning significantly improves with the use of the Incremental Averaging method.

Keywords: Federated Learning, Incremental Averaging, Federated Averaging.

1. Introduction

Federated learning is a distributed machine learning approach where model training process is decentralized. In conventional intelligent learning, data is centralized on a single server, which performs unified operations and scheduling to ultimately form a virtual model.[1] This has been changed in federated learning, compared to a single centralized data, federated learning train models across multiple devices. These devices do not need to be the same type as the main server; they can be any device used in daily life, including phones, computers, etc. Federated learning involves a series of processes that do not involve data interaction between devices, but only model interaction. Firstly, each device downloads the model from the main server, then trains the model using the data from the current device and sends the trained model back to the main server. The main server merges the new model received into the main model. At this point, due to the differences in the new models generated by each corresponding device, specific formulas are needed to merge them. Afterwards, the main server will continue to use the improved model and send the new model to the device requesting current model. The key idea here is to bring the code to the data instead of bringing the data to the code. Below is an example diagram showing how federated learning operates between different devices and the main server.

DOI: 10.54254/2755-2721/107/2024.18310

2. Related work

Recent research has mainly focused on a deeper exploration of FedAvg, aiming to change its computational methods.[2] For example, in the paper "FedAvg with Fine Turning: Local Updates Lead to Representation Learning", led by Lian Collins, a new approach is explicitly mentioned. Compared to an innovative algorithm, this paper is more closely aligned with a supplementary study on FedAvg. Due to the fact that the FedAvg algorithm mainly focuses on how to integrate models and does not take into account the level of refinement of individual user models. T This causes the traditional FedAvg model to struggle with non-IID (non-independent and identically distributed) data across clients, leading to suboptimal performance. In the experiment, researchers achieved efficiency improvement by fine-tuning the model on their local data even after updating the global model. The number of times this fine-tuning is determined by the corresponding formula. The paper demonstrates that the proposed method leads to significant improvements in model performance, particularly in scenarios with highly non-IID data.

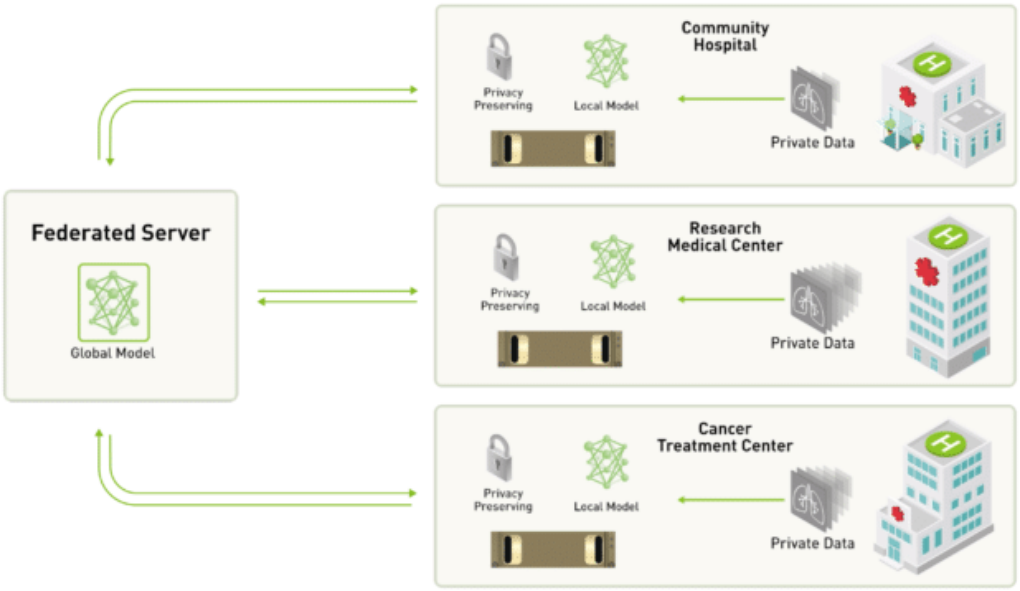

Figure 1. Federated learning process[3]

3. Benefit and deficiency of federated learning

Federated Learning has many benefits, and the following five points are particularly important. First, federated Learning ensures the security of user data. Due to the fact that only the model itself is involved in the transmission process not the data, the data only stays on the current device and will not be transferred. This results in the data not being intercepted or stolen during the transmission process, thus eliminating most of the possibility of data theft. Secondly federated Learning effectively reduces the demand for the main server equipment. During the training process, the main server does not need to face billions of data points, it only needs to allocate and merge models. Each device independently sends the current new model to the main model instead of a large number of data packets, reducing the requirement for network speed. The main server that receives these models does not require data to calculate the models, which reduces the need for computer CPUs and improves efficiency, allowing servers with the same configuration to calculate more complex and accurate models. Third point is that federated learning can be customized through privatization. Due to the nature of its decentralized model, the main model can decide which devices to use to calculate sub model, providing more options. Personalized Federated learning has become possible as a result, giving rise to many websites such as TensorFlow Federated, OpenMined PySyft.[4] People can join the federated learning sequence on these websites. The fourth point is data diversity and representativeness. For federated learning, although it

DOI: 10.54254/2755-2721/107/2024.18310

cannot directly read information from users, the source of its model is obtained from user information. The most mainstream learning method nowadays: supervised learning method obtain the final model by concentrating a large amount of data. The accuracy of the model largely depends on the quality and representativeness of the collected data. For example, in image recognition applications, traditional supervised learning typically collects a large amount of image data and stores these images as well as corresponding labels on a server for training. To train a model for recognizing cats and dogs, the data needs to include a large number of cat and dog images from different breeds, different shooting angles and different lighting conditions. In this case, the model performs well because it learns more representative features and is able to accurately identify cats and dogs. However, if the dataset lacks images of cats and dogs in certain specific scenarios, such as dark environments or specific breeds of dogs, the model will make errors when encountering these situations. Because the model has only seen limited types and environments of cat and dog images, it may perform poorly when encountering situations that have not been seen before. Federated learning reduces the likelihood of such situations occurring, with a wide range of data sources while avoiding unconscious screening that may occur during the data collection process. Last point is the stability. Due to the nature of federated Learning, the models it produces will be more randomized and realistic. When devices other than the main server fail, the efficiency of federated learning will not be affected. This makes federated Learning have a lot of scalabilities and cutting-edge applications.[5][6]

The drawbacks of federated learning mainly lie in its dependence on high-performance devices and lack of discriminative ability for Non-IID data. Federated learning requires sending the user's model to the main server, which will result in significant broadband consumption. Once the broadband of the main server is blocked or malfunctioning, the entire training process will be forced to stop. Federated learning also heavily relies on the local training capabilities of user devices, but the quality of user devices varies. Some devices, such as smartphones do not have enough memory to support complex calculations.[7]The biggest challenge faced by federated learning in terms of data is Non-IID data, which stands for Non-Independent and Identifiably Distributed data Non IID data means that the data in the database is not evenly distributed. There may be differences or connections among them. The differences in living regions and cultures can result in significant variations in the models obtained on the user side. The main model generated using similar models may lack typicality and be difficult to represent individual situations.[8]

4. Applications of federated learning

Federated learning has a wide range of applications and a high level of depth. The application of several industries is particularly important and has produced a certain degree of feedback to society. This includes the healthcare industry, where people can access corresponding self-generated healthcare plans through the use of federated learning. Multiple hospitals can work together to complete this project. Federated learning allows hospitals to collaboratively train models for detecting diseases in medical images like X-rays, MRIs, and CT scans. By leveraging data from multiple sources, these models can improve diagnostic accuracy and robustness without sharing raw patient data. Predictive models for diseases such as cancer, diabetes, and COVID-19 can be enhanced using federated learning by training on data from different geographic locations and institutions. This helps in creating more generalized models that are less prone to institutional bias. Federated learning can also help them avoid potential risk investments and trading traps, and better protect their corresponding assets.

Federated Learning has also been well utilized in the field of Autonomous Vehicles. Autonomous vehicles typically require the ability to quickly identify and respond to unexpected situations. But in the real world, the situation is complex. For example, the weather may be sunny or blizzard; The road conditions are sometimes flat and sometimes bumpy; The one lying in the middle of the road could be a plastic bag or a baby stroller. In this case, the breadth and accuracy of the model are particularly important.[9]

Federated learning also has prominent applications in the field of education. Thanks to good privacy protection, federated learning can help different students customize their learning plans and improve

DOI: 10.54254/2755-2721/107/2024.18310

their learning outcomes. This is helpful for non-native students studying in different cultures or regions. Some foreign students may not be accustomed to the local education system, but thanks to the intervention of federated learning, they can quickly find a suitable study plan for themselves. Federated learning can conduct periodic evaluations of these students' grades and provide guidance. The model originates from different students around the world, and there will always be cases that fit the current situation in such a large amount of data.[10]

5. Algorithm of federated learning

The most commonly used algorithm when training models is FedAvg. It is a relatively basic algorithm. The actual process of this algorithm will be described in detail below. Suppose there is a central server and M clients. Each client i has a local training dataset $\widehat{D}_i$ consisting of n_i labeled samples drawn from a distribution $\widehat{D}_i$ over input space X and label space Y. The learning model is denoted by$h_\theta: X \to Y$, parameterized by $\theta \in \mathbb{R}^D$. The loss on a sample (X, Y) is given by $\ell(h_\theta(X), Y)$, which could be squared loss, cross-entropy loss, etc. The average loss of the model h_θ on the samples in $\widehat{D}_i$ for client i is:

$$fi(\theta) = \frac{1}{n_i}\sum_{i=1}^{n_i} \ell\left(h_\theta(X_i, j), Y_i, j\right) \tag{1}$$

The server aims to minimize the average of the client losses, weighted by the number of samples. Since the goal is to find a single model θ that minimizes the average loss, the following calculation formula can be derived:

$$\min_\theta \frac{1}{N}\sum_{i=1}^{M} n_i f_i(\theta) = \frac{1}{N}\sum_{i=1}^{M}\sum_{j\in\widehat{D}_i} \ell\left(h_\theta(X_i, j), Y_i, j\right) \tag{2}$$

In equation (2) $N = \sum_{i=1}^{M} n_i$ the total number of samples across all clients. Clients cannot share their local data $\widehat{D}_i$ directly due to privacy and communication constraints, so the optimization must be performed in a federated manner. During the client selection part, in each round t of FedAvg, the server selects a subset $\mathfrak{T}_i$ of $m \le M$ clients. Starting with the local training each selected client receives the current global model parameters θ_t, each client performs local updates by running multiple steps of stochastic gradient descent (SGD) on their local data starting from θ_t. To be more specific, the local update rule is:

$$\theta_{t,i,s} = \theta_{t,i,s-1} - \alpha g_{t,i,s}\left(\theta_{t,i,s-1}\right) \tag{3}$$

In equation (3)$g_{t,i,s}\left(\theta_{t,i,s-1}\right)$ is the stochastic gradient computed using a mini-batch of local data, and $s = 0, \ldots, \tau - 1s$ with $\theta_{t,i,0} = \theta_t$. After completing local updates, each client sends the updated model parameters $\theta_{t,i,\tau}$ back to the server. The server aggregates these updates by computing a weighted average to form the new global model:

$$\theta_{t+1} = \frac{1}{N_t}\sum_{i\in\mathfrak{T}_t} n_i \theta_{t,i,\tau} \tag{4}$$

In equation (4) $N_t = \sum_{i\in\mathfrak{T}_t} n_i$ is the total number of samples across the selected clients.[11][12]

6. New method

It is not difficult to find in the above Fedavg formula that when the main server receives client models to update the main model, it determines the weight of these model to the main model by judging the number of data points for each client model. Although Fedavg does not assume that all selected clients' models will arrive at the same time, it will wait until the model of the selected clients in that round all

DOI: 10.54254/2755-2721/107/2024.18310

arrived before proceeding to the next step of computation. This results in resource waste as the main server does not perform any work during the waiting period. To address this issue, I propose using the Incremental Averaging method. The specific principle is as follows, assuming that the main server receives IID data. At first, ignore the impact of the main model and focus on merging each client model, after merging all client model finally merge it with the main model. The specific method is as follows: first, assume that there is an average value during the merging process of the two models, denoted as $\bar{x}$. The same method as Fedavg is used here to calculate this average value. Assuming that the average model after receiving the update from one of the select client is $\bar{x_{new}}$, before receiving the update from one of the select client is $\bar{x}_{old}$. There are a total of i clients participating in the experiment, α_i is a constant that varies with the change of i , $\alpha_i = \frac{1}{i}$. For example, when the second client transmits data $\alpha_2 = \frac{1}{2}$. x_i is the model of the i-th client. When receiving data, the following formula can be achieved.

$$\bar{x}_{new} = \bar{x}_{old}(1 - \alpha_i) + \alpha_i x_i \tag{5}$$

By using equation (5), data processing can be performed simultaneously with receiving the model.

In addition, upload time, download time, and computation time all affect the time when the model reaches the parameter server from a local client. Therefore, these data can be recorded during the process for further research.

7. Evaluation of incremental averaging method

The MNIST dataset will be used in the experiment to test federated learning. MNIST is a benchmark dataset for image classification that includes handwritten digit (0-9) images and is widely used for benchmark testing of image classification. Each image is a 28x28 pixel grayscale image with a total of 70000 samples (60000 training samples and 10000 testing samples). [13] The main purpose of choosing the MNIST dataset is to focus on simulating Non-IID data, with a moderate sample size and low processing costs. The most commonly used federated learning method FedAvg will be used as the baseline in the experiment. Apply the baseline method and the improved method to the same set of experimental conditions to more accurately compare performance. Use lightweight Convolutional Neural Networks (CNNs) to ensure effective training under limited device computing resources. [14][15] The evaluation of new models created using incremental averaging calculating method mainly focuses on the following three points. The first point is the accuracy of the model. Record the accuracy of the global model on the test set in each round of experiments, and evaluate whether the incremental averaging method improves the accuracy of the model by comparing it with the main model generated by the FedAvg method. The second point is the convergence speed of the model. It measures the number of training iterations required to achieve the same level of accuracy. The fewer times, the less resources the model needs to complete the task. The third point is communication cost. When using different federated learning methods for training, communication costs will be incurred to varying degrees. This expense comes from the model interaction between the main server and user devices. In the experiment, the number of communications and data transmission during each training round can be recorded to evaluate the performance of the incremental averaging calculating method in reducing communication costs.[16][17]

DOI: 10.54254/2755-2721/107/2024.18310

8. Result

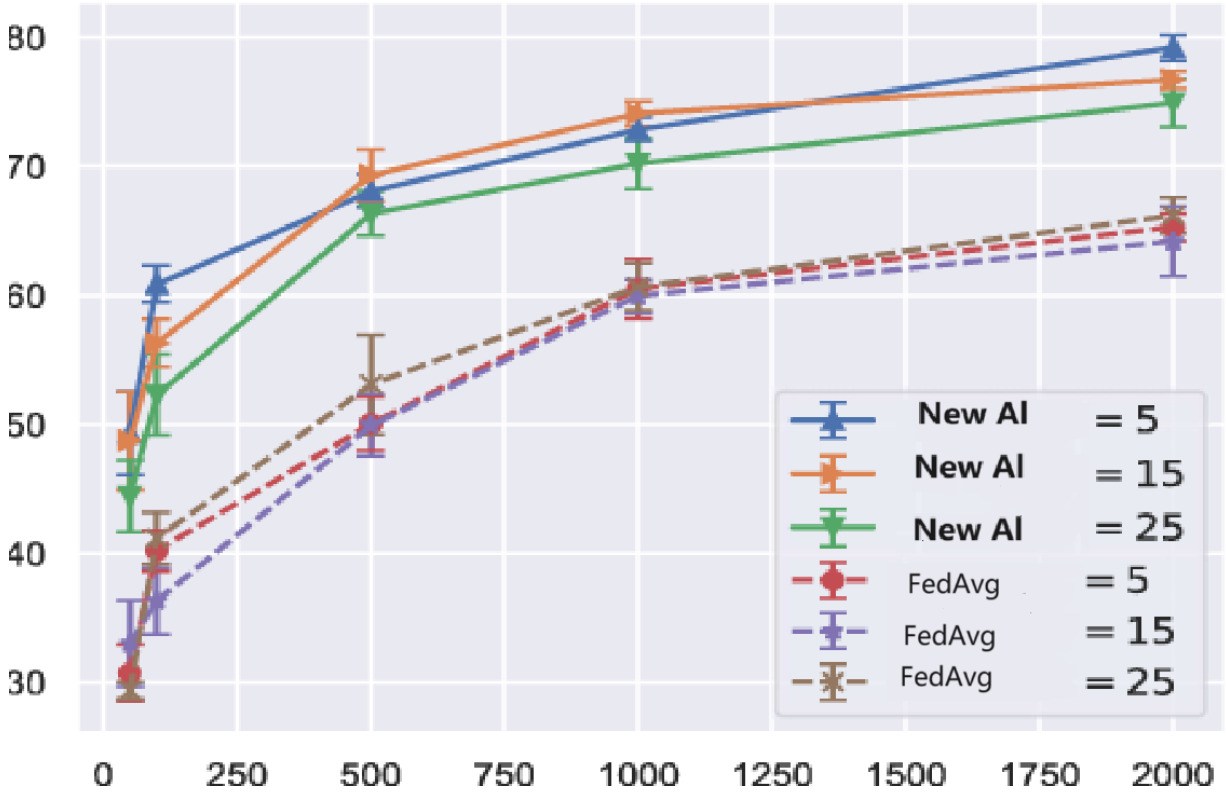

Figure 2. Result after applying incremental averaging[18]

By applying the new incremental averaging calculating method, we should obtain results similar to those in Figure 2. The experiments are conducted in the context of federated learning. It aims to improving the main model by incorporating updates from multiple clients. Each client work on their local portion of the data without sharing raw data just as explained in Introduction. The main difference compared to the traditional federated learning model is that in the standard FedAvg algorithm, the server posts model aggregation until updates from all chosen clients are received. However, in the new model, the incremental averaging technique is used, aggregation will happen as soon as each client transmit his own model. In Figure 2, the horizontal axis represents the number of samples, and the vertical axis represents accuracy. FedAvg represents federated learning using traditional batch averaging. New Al represents federated learning using incremental averaging as explained in section VI. This leads to a new formula that can improve the accuracy of the model. Under the same number of samples. The new calculation formula using the incremental averaging method can obtain more detailed and clear information, thereby achieving higher efficiency. For future improvement, an autonomous experiment should be conducted to obtain more accurate data. Simultaneously using more devices to monitor convergence speed and communication cost

9. Conclusion

This study proposes a new incremental averaging method to improve the aggregation technique of traditional federated learning models. Theoretical analysis shows that this method is expected to outperform traditional FedAvg methods in terms of accuracy, convergence speed and communication cost. The incremental averaging method reduces the time for servers to wait for updates from all clients by gradually updating the model, thereby achieving more efficient model aggregation. Especially in Non-IID data environments where device performance differences are significant, this method can more

DOI: 10.54254/2755-2721/107/2024.18310

effectively integrate diverse data from multiple clients while maintaining privacy. Although the experimental part has not been completed yet, the feasibility of theoretical derivation and model design algorithms provides a foundation for subsequent research. Future work can validate the performance of this method through detailed experiments on MNIST or larger datasets, and evaluate its actual effectiveness in more complex scenarios. Moreover, future research on federated learning can explore its scalability by applying incremental averaging methods to practical federated learning applications in different industries with prominent data heterogeneity and privacy issues, such as healthcare and finance. For these industries, profitability is necessary, and the incremental averaging method can help them better save communication costs to achieve profitability.

References

[1] Konečný, Jakub, et al. "Federated Learning: Strategies for Improving Communication Efficiency." arXiv preprint arXiv:1610.05492 (2016). PDF file.

[2] Collins, Lian, et al. "FedAvg with Fine Tuning: Local Updates Lead to Representation Learning." Proceedings of the Neural Information Processing Systems Conference, 2022. Figure 2 Adapted and modified model name.

[3] "Federated Learning: Collaborative Machine Learning without Centralized Training Data." NVIDIA Blog, NVIDIA, 26 May 2021, blogs.nvidia.com/blog/what-is-federated-learning/. Accessed 2 June 2024.

[4] Nishio, Takayuki, and Ryo Yonetani. "Client Selection for Federated Learning with Heterogeneous Resources in Mobile Edge." IEEE International Conference on Communications (ICC), 2019. PDF file.

[5] Collins, Lian, et al. "FedAvg with Fine Tuning: Local Updates Lead to Representation Learning." Proceedings of the Neural Information Processing Systems Conference, 2022. Figure 2 Adapted and modified model name.

[6] Liu, Bingyan, et al. "Recent Advances on Federated Learning: A Systematic Survey." arXiv, 2023, arxiv.org/abs/2301.01299.

[7] Sattler, Felix, et al. "Robust and Communication-Efficient Federated Learning from Non-IID Data." IEEE Transactions on Neural Networks and Learning Systems, vol. 31, no. 9, 2020, pp. 3400–3413.

[8] Li, Qinbin, et al. "A Survey on Federated Learning Systems: Vision, Hype and Reality for Data Privacy and Protection." IEEE Transactions on Knowledge and Data Engineering, vol. 34, no. 12, 2022, pp. 1–18.

[9] McMahan, H. Brendan, et al. "Communication-Efficient Learning of Deep Networks from Decentralized Data." arXiv preprint arXiv:1602.05629 (2016). PDF file.

[10] "FedAvg with Fine Tuning: Local Updates Lead to Representation Learning." Proceedings of the Neural Information Processing Systems Conference, 2022. Figure 2 Adapted and modified model name.

[11] Smith, Virginia, et al. "Federated Multi-Task Learning." Advances in Neural Information Processing Systems, vol. 30, 2017, pp. 4424–4434.

[12] Kairouz, Peter, et al. "Advances and Open Problems in Federated Learning." Foundations and Trends® in Machine Learning, vol. 14, no. 1–2, 2021, pp. 1–210.

[13] Li, Tian, et al. "Federated Learning: Challenges, Methods, and Future Directions." IEEE Signal Processing Magazine, vol. 37, no. 3, 2020, pp. 50–60.

[14] Zhang, Chao, et al. "Federated Learning for Healthcare Informatics." Knowledge-Based Systems, vol. 213, 2021, 106684.

[15] Mohri, Mehryar, et al. "Agnostic Federated Learning." Proceedings of the 36th International Conference on Machine Learning, vol. 97, 2019, pp. 4615–4625.

[16] Wang, Hongyi, et al. "Federated Learning with Matched Averaging." arXiv, 2020, arxiv.org/abs/2002.06440.

DOI: 10.54254/2755-2721/107/2024.18310

[17] Li, Tian, et al. "Federated Optimization in Heterogeneous Networks." *Proceedings of Machine Learning and Systems*, vol. 2, 2020, pp. 429–450

[18] Collins, Lian, et al. "FedAvg with Fine Tuning: Local Updates Lead to Representation Learning." Proceedings of the Neural Information Processing Systems Conference, 2022. Figure 2 Adapted and modified model name.