Proceedings of the 6th International Conference on Computing and Data Science
DOI: 10.54254/2755-2721/86/20241594

Federated learning: Impact of different algorithms and models on prediction results based on fashion-MNIST data set

Yuan Gao

School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China

u202110081@hust.edu.cn

Abstract. The realm of federated learning is rapidly advancing amid the era of big data. Therefore, how to select a suitable federated learning algorithm to achieve realistic tasks has become particularly critical. In this study, we explore the impact of different algorithms and models on the prediction results of Federated Learning (FL) using the Fashion-MNIST data set. Federated Learning enhances data privacy and reduces latency by training models directly on local devices since it is a decentralized machine learning approach. We analyze the performance of several FL algorithms including Federated Averaging (FedAvg), Federated Stochastic Gradient Descent (FedSGD), Federated Proximal (FedProx), and SCAFFOLD. Our experiments reveal significant differences in accuracy and stability among these algorithms, highlighting their strengths and weaknesses in handling non-IID (Non-Independent and Identically Distributed) data. FedProx demonstrate superior performance in terms of accuracy and robustness, making them suitable for complex federated learning environments. These discoveries offer crucial insights for choosing suitable FL algorithms and models in practical applications.

Keywords: Federated Learning, Fashion-MNIST, FedAvg

1. Introduction

Federated Learning represents a significant shift in trained machine learning models, particularly in environments prioritizing data privacy. Traditional centralized machine learning gather and process data on a central server, while FL enables model training directly on devices where data originates. This approach both strengthens data privacy and mitigates latency and bandwidth costs associated with data transmission. Federated Learning encompasses several essential stages: the global model is first initialized and then distributed to local devices. Each device conducts training using its local data. The local models are then aggregated to update the global model. This cycle repeats iteratively until the global model reaches convergence. The robustness of FL in handling diverse data distributions and its capacity to leverage distributed data sources make it an essential technique in the era of big data and IoT (Internet of Things). Therefore approaches that keep data on the device and share the model have become increasingly attractive.

Studying the impact of different algorithms and models on prediction results is crucial for improving the efficiency and effectiveness of federated learning. Firstly, different algorithms may exhibit significant differences in terms of data distribution, computational capability, and communication costs. Comparative analysis can help identify the optimal algorithm for specific scenarios, enhancing the overall system performance. Secondly, with the proliferation of IoT devices and edge computing, the performance of federated learning in handling various types of data is also a key research focus. Different types of data may require different processing methods and optimization strategies to achieve better model training outcomes.

Current researches primarily concentrate on enhancing the algorithm, and lack sufficient discussion on the application scenarios and effects of the improved algorithm. At the same time, few studies have paid attention to the selection of neural network model. In this paper, we study the impact of different algorithms and models on prediction results within FL. We analyze algorithms like FedSGD, FedAvg, FedProx, and SCAFFOLD on the Fashion-MNIST data set. Our goal is to understand their performance differences and identify which algorithms are best suited for more complicated (compared to MNIST) federated learning environments. This paper's research enhances our understanding of the strengths and weaknesses of various algorithms and models using the Fashion-MNIST dataset, providing valuable guidance for selecting algorithms and models in practical applications.

2. Related work

Currently, the main algorithms for federated learning include FedAvg, FedSGD, FedProx , and SCAFFOLD. Each of these algorithms has its unique features and application scenarios. FedAvg and FedSGD were proposed by McMahan et al. [1]. FedAvg is a basic and widely used algorithm of FL. Due to its simplicity and low communication overhead, it has been widely adopted in practical applications. FedProx was proposed by Li et al. [2]. It introduces a regularization term on the basis of FedAvg to alleviate the model convergence issues caused by different data distributions across nodes (i.e., non-IID data). SCAFFOLD was proposed by Karimireddy et al. SCAFFOLD reduces the variance in global model updates by introducing control variables, thereby accelerating convergence and improving model performance [3, 4]. This method utilizes the differences between local and global updates, correcting each node's model update direction to reduce error accumulation, making it particularly suitable for scenarios with highly heterogeneous data distributions.

Kairouz et al. summarizes the latest advancements and open challenges in federated learning [5]. The authors discuss the current state in data privacy, system architecture, and algorithm optimization, and identify future research directions. This paper highlights the crucial role of algorithm optimization in FL, which aligns with the goal of this study to explore optimal strategies by comparing the performance of four algorithms on the Fashion-MNIST data set. Li et al. delves into the challenges faced by federated learning, including non-IID data, communication inefficiencies, and system heterogeneity [6]. The authors propose a series of optimization methods, such as data augmentation, model compression, and asynchronous updates, to address these challenges. Inspired by [6], we performed experiments to verify Impact of different algorithms on prediction results.

Wang et al. proposes a method using reinforcement learning to optimize federated learning, addressing the challenges posed by non-IID data [7]. The authors introduce a policy gradient-based reinforcement learning framework that learns optimal client selection and parameter update strategies to improve the performance of FL. The experimental results of the paper demonstrate that the use of reinforcement learning can significantly improve model convergence speed and accuracy under non-IID data conditions. Paper [7] uses multiple data sets. Compared with MNIST data set and CIFAR-10, the optimization algorithm does not perform well for Fashion MNIST. This is the reason why the Fashion MNIST data set is selected in this paper.

Bonawitz et al. discusses Google's practical experience in designing large-scale federated learning systems [8]. The authors address technical challenges and solutions in implementing FL in large-scale distributed environments, including secure aggregation, model synchronization, and node fault handling. This study draws on the design ideas from this paper in building the experimental platform and system optimization, ensuring that the experimental process effectively simulates real-world system operations to obtain more valuable reference results.

3. Algorithms

This section offers a comprehensive description of the algorithms utilized in our federated learning experiments. Specifically, we focus on FedSGD, FedAvg, FedProx, and SCAFFOLD. Each of these algorithms addresses different aspects of the federated learning paradigm, such as handling non-IID data distributions, reducing communication costs, and improving convergence rates. We will outline the key mechanisms and mathematical formulations underlying these algorithms, highlighting their distinctive features and potential advantages in the context of our experiments on the Fashion MNIST data set.

3.1. FedSGD&FedAvg

FedAvg is one of the foundational algorithms in the realm of FL [1]. It is a simple yet effective method that involves averaging the model weights from multiple local devices. In each communication round, the global model is distributed to participating devices, where local training is conducted using their respective local data. Upon completion of training, the updated local models are transmitted back to the central server, which aggregates these models to update the global model.The strength of FedAvg lies in its simplicity and ease of implementation. It efficiently reduces the communication overhead by aggregating model weights instead of gradients, making it suitable for scenarios with limited communication bandwidth. However, its performance can be sensitive to the heterogeneity of data across devices, potentially leading to suboptimal convergence.

FedSGD is another pivotal algorithm in FL, focusing on the aggregation of gradients rather than model weights. It aims to harness the benefits of SGD in a federated setting, particularly its ability to handle large-scale data efficiently. The primary advantage of FedSGD is its potential for faster convergence compared to weight-based aggregation methods. However, it also entails higher communication costs, as gradients need to be transmitted frequently between devices and the central server. In FedAvg, given K clients indexed by k; B is the local minibatch size, E is the number of local epochs. we can take $B = \infty$ and $E = 1$ which corresponds exactly to FedSGD.

3.2. FedProx

FedProx enhances FedAvg by adding a proximal term, which helps tackle the issues of data heterogeneity and system heterogeneity in federated learning [2]. The proximal term acts as a regularizer that penalizes significant deviations from the global model during local training. This approach helps to mitigate the drift of local models, especially when the data distribution across devices is non-iid. FedProx enhances the stability and robustness of the training process, ensuring better convergence even in highly heterogeneous environments. While it introduces additional computational complexity due to the proximal term, its benefits in terms of convergence and performance in non-iid settings make it a valuable algorithm in FL.

3.3. SCAFFOLD

SCAFFOLD is designed to address the variance and bias issues arising from local updates in FL [3]. SCAFFOLD introduces control variates to correct the local updates, reducing the variance and ensuring more consistent updates to the global model. By aligning local gradients with the global direction, SCAFFOLD improves the convergence rate and accuracy of the global model. The key innovation in SCAFFOLD is the use of control variates to provide a more precise estimation of the global gradient, thereby enhancing the overall performance of the federated learning process. However, this approach involves additional overhead in maintaining and updating the control variates, which can be a trade-off for the improved convergence benefits.

3.4. Comparative Evaluation of Algorithms

FedAvg's advantages include simplicity and low computational overhead. However, its performance may be suboptimal when dealing with non-IID data, leading to slow or unstable model convergence. FedSGD's main advantage is its ability to quickly respond to data changes, making it suitable for dynamic data environments. However, FedSGD has high communication overhead since gradients need to be transmitted with each update, which can become a bottleneck in large-scale systems.

FedProx introduces a regularization term on top of FedAvg. This term makes the local models closer to the global model, improving convergence under non-IID data conditions. While FedProx performs well with non-IID data, it also increases computational complexity and implementation difficulty.

SCAFFOLD enhances model stability and convergence speed, especially in non-IID data environments. However, SCAFFOLD is more complex to implement and requires additional storage and computational resources to maintain the control variates.

In summary, FedAvg is suitable for scenarios with relatively uniform data distribution and limited computational resources; FedSGD is ideal for dynamic data environments but requires high communication bandwidth; FedProx excels in handling non-IID data and is suitable for uneven data distribution scenarios; SCAFFOLD offers advantages in improving model stability and convergence speed but has higher implementation complexity and resource requirements. By comparing and evaluating these four algorithms, this study provides a reference for selecting the appropriate federated learning algorithm for practical applications.

4. Experiments

4.1. Setup

4.1.1. Fashion-MNIST data set The Fashion-MNIST dataset serves as a challenging benchmark for image classification tasks, replacing the traditional MNIST dataset [4]. It comprises 70,000 grayscale images depicting fashion items categorized into 10 classes: T-shirts, trousers, pullovers, dresses, coats, sandals, shirts, sneakers, bags, and ankle boots. Each image measures 28x28 pixels, providing standardized input for diverse machine learning models (as shown in Figure 1). The dataset is divided into 60,000 training images and 10,000 test images, facilitating comprehensive model training and evaluation. Fashion-MNIST offers increased complexity compared to MNIST, making it an ideal benchmark for assessing federated learning algorithm performance.

4.1.2. Implementation The implenmentation steps of different algorithms are similar. First, We implement FedAvg, FedSGD, FedProx and SCAFFOLD in Tensorflow . In order to Implementing data processing, we normalize the pixel values of the images to the [0, 1] range. And we need to add a new single-channel dimension at the end to fit the CNN model. Next, we split the training dataset into multiple subsets, each representing a client's data. This step simulates the federated learning environment, where each client possesses its own local data for training. Table 1 is the set of Hyper parameters

Table 1. Hyper parameters

	parameters	value
1	num_clients	5
2	rounds	10
3	epochs	1
4	optimizer	adam
5	loss	sparse_categorical_crossentropy
6	metrics	'accuracy'

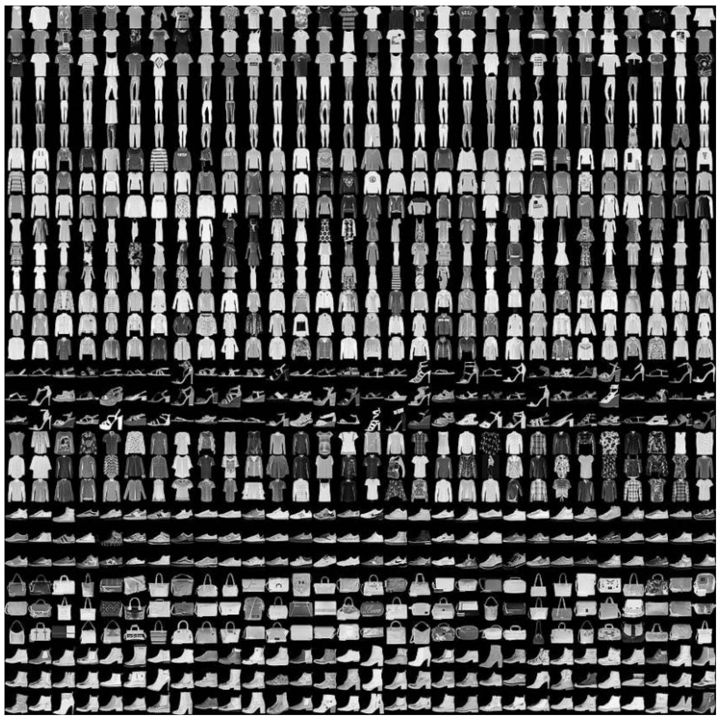


Figure 1. Fashion-MNIST data set [4]

4.2. *Evaluation metrics*

The Training Accuracy Curves compares the accuracy under different number of rounds, it can show how the accuracy of the models changes over training rounds for algorithms, comparing convergence speed and effectiveness.

4.3. *Model Selection (based on FedSGD algorithm)*

All experiments follow the principle of controlled variables, and the learning rate of its NET model is set to 0.01, the batch size of its NET model is 32.

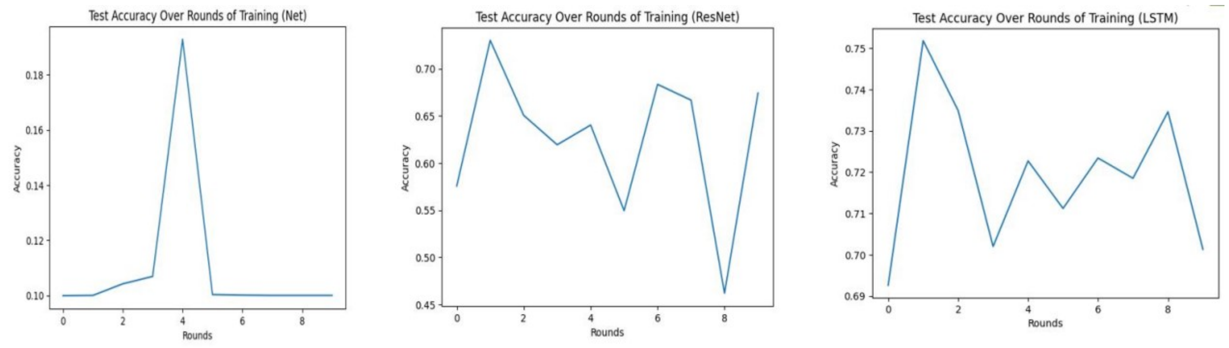
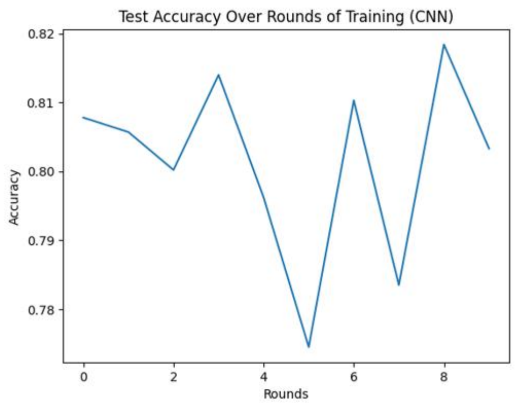


Figure 2. Net, ResNet, LSTM model accuracy (Picture credit : Original)



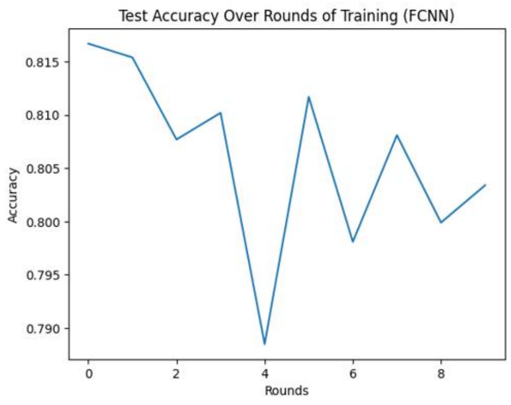


Figure 3. CNN&FCNN model accuracy (Picture credit : Original)

Table 2. Model accuracy

	Model	Accuracy
1	Net	0.1
2	ResNet	0.6742
3	LSTM	0.7013
4	CNN	0.8033
5	FCNN	0.8034

Figure 2 and figure 3 show how the test accuracy changes across rounds when training with different neural network models (Net, ResNet, LSTM, CNN, and FCNN). The horizontal axis represents the number of training rounds. The vertical axis represents accuracy. Overall, NET performs extremely poorly, ResNet and LSTM perform poorly and have large curve fluctuations, and CNN and FCNN have high accuracy and small curve fluctuations.

Net model performs poorly on this task, we can see the final accuracy is about 0.1, which means the trained model cannot perform effective recognition (Table 2). The accuracy of ResNet [9] and LSTM [10] models are relatively close, around 0.7, which proves that these two models are not suitable for federated learning algorithms. The accuracy of CNN and FCNN is relatively close, around 0.8. Although this data is still not ideal, it is the disadvantage of FedSGD algorithm that leads to the low accuracy due to several factors. The primary reasons include non-IID data distribution across clients, leading to biased and divergent updates, high communication overhead from frequent server-client interactions, lack of extensive local training resulting in high variance gradients, and scalability issues when involving a large number of clients. These factors collectively hinder the stability and efficiency of the training process in federated learning environments.

The Net model has a weight update strategy problem: The weight update strategy used is to simply add the weight of the local model with the weight of the global model. This approach may not effectively integrate local updates across clients, especially when client data is unevenly distributed or the model is not sufficiently trained locally.

LSTM is a neural network specifically designed to process sequence data. Since Fashion-MNIST is an image classification task and image data does not have time dependence like sequence data, LSTM is not suitable for such image classification tasks.

Although ResNet is a very powerful image recognition model, it is designed to handle more complex and deeper image recognition tasks than Fashion-MNIST. For the relatively simple Fashion-MNIST data set, using ResNet may lead to unnecessary waste of computing resources, and the model may be too complex and prone to overfitting.

Generally FCNN is used for image segmentation tasks. However, in classification tasks, fully convolutional networks effectively learn the local features of images by leveraging their spatial hierarchical performance, thereby achieving high accuracy.

The CNN model is composed of two convolutional layers, two pooling layers, a flat layer and a fully connected layer, and the activation function is ReLU. CNN is more suitable than other models for processing the Fashion-MNIST data set when choosing a model architecture for several reasons:

Feature extraction capability: CNN automatically extracts important features from images through convolutional layers without manually designing feature extractors, which is very important for Fashion-MNIST images containing complex patterns and styles.

Model complexity and computational efficiency: CNN models can be designed to be relatively lightweight, fast in training, and easy to debug and optimize.

Local connection and weight sharing: These two characteristics of CNN reduce the number of parameters and improve its generalization ability, which means that CNN is better able to learn from limited training samples and reduces the risk of overfitting.

Spatial hierarchy: Through multi-layer convolution and pooling operations, CNN is able to capture the hierarchical structure in the image, from simple edges to complex shapes and patterns. This is very useful for clothing image classification in Fashion-MNIST because clothing images contain various shapes and textures. Thus, we choose CNN as the model.

4.4. Algorithm Comparison

Based on the CNN model, different algorithms are selected for experiments to compare the training accuracy of each algorithm on the Fashion MNIST data set (Table 3). Figure 4 and figure 5 show how the test accuracy changes across rounds when training with different algorithms. The horizontal axis represents the number of training rounds. The vertical axis represents accuracy. We can see that the FedSGD algorithm not only has a low accuracy, but also has a large fluctuation of the accuracy curve. However, the accuracy of the other three algorithms is relatively close, and the accuracy curve has an obvious upward trend with the increase of the number of rounds.

The FedSGD algorithm has the lowest final accuracy. The accuracy of 0.8025 proves that the algorithm has difficulties in identifying the data set. At the same time, through the accuracy curve, it can be seen that the method is not stable in recognition. Since the FedSGD algorithm works by completing a gradient update independently on each client and then sending all clients' updates to a central server for averaging. A significant drawback of this approach is that it does not account for the non-IID nature of the data across clients. In this project, each client may have data with different characteristics, which will cause the features learned by the model during local updates to be non-universal, consequently impacting the overall model's performance and generalization capability.

The FedAvg algorithm improves upon FedSGD by running multiple local iterations on each client and then averaging these updates. This approach can reduce communication costs and allow more complex local model training, thereby improving model performance.

The FedProx algorithm is a variant of FedAvg. Compared with FedAvg, the algorithm has higher accuracy, which proves that FedProx has advantages for processing fashion-mnist data set. FedProx handles heterogeneous data, that is, differences in data distribution on different clients, by adding a regularization term to local updates.This regularization term helps stabilize the training process and ensures smoother and more consistent model updates for each client, even in the presence of large differences in data distribution. This approach improves the model's robustness and ultimate performance on diverse data sources, especially when faced with Fashion-MNIST data that is common in the real world. Moreover, by limiting the deviation degree of the local model, FedProx can make the global model converge to a better state faster in a small number of training rounds, which may be the reason for FedProx's excellent performance in fewer rounds.

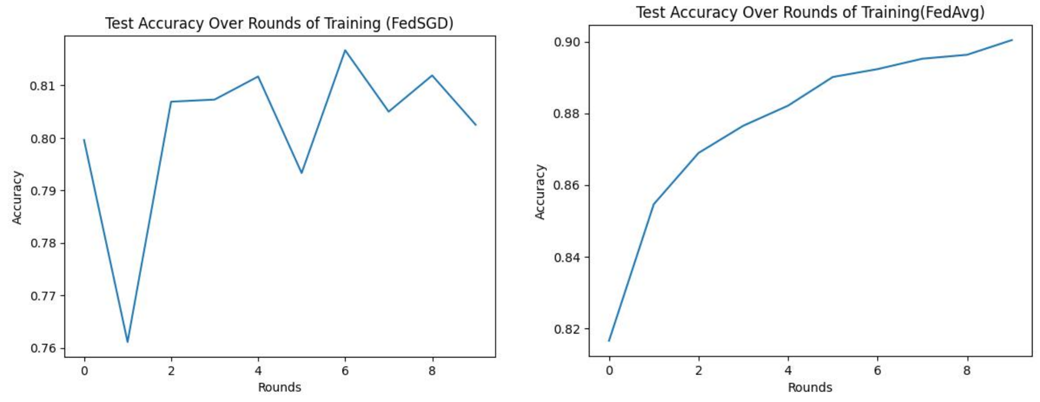


Figure 4. FedSGD accuracy and FedAvg accuracy (Picture credit : Original)

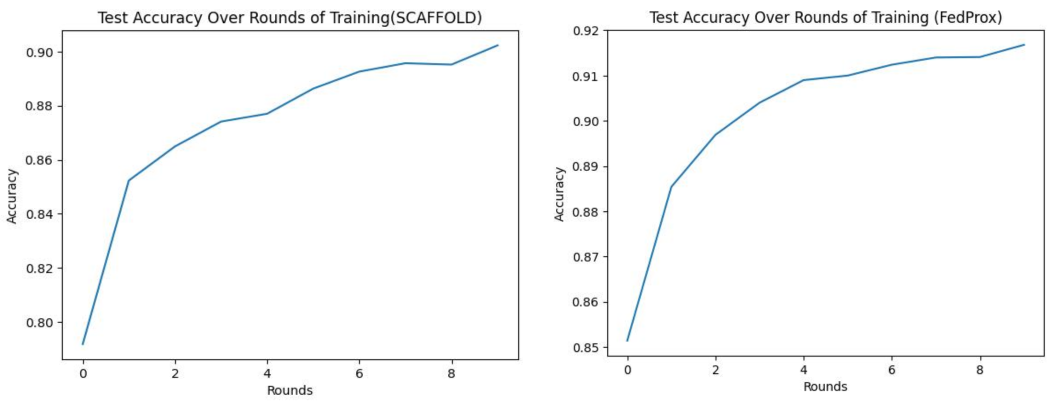


Figure 5. SCAFFOLD accuracy and FedProx accuracy (Picture credit : Original)

Table 3. Algorithm accuracy

	Algorithm	Accuracy
1	FedSGD	0.8025
2	FedAvg	0.9004
3	FedProx	0.9168
4	SCAFFOLD	0.9023

The SCAFFOLD algorithm further optimizes the federated learning process by introducing control variables to correct deviations in client updates, which helps to solve the problem of non-independent and identically distributed data.However, although SCAFFOLD provides better bias correction in theory, its performance in this project is similar to FedAvg. Fewer training rounds or smaller local data sets were used in the experiments, and the advantages of SCAFFOLD may not be fully reflected. The complexity and correction mechanism of SCAFFOLD requires more iterations and data to show its effect. The effect may not be apparent if the training time is short or the amount of local data is insufficient.

5. Conclusion

This paper addresses the critical challenge of understanding the impact of various federated learning algorithms and models on prediction accuracy, specifically using the Fashion-MNIST data set. The primary issue we tackle is the performance variability of FL systems when exposed to non-IID data, which is a common scenario in real-world applications. Our study aims to provide a comprehensive comparative analysis of four prominent federated learning algorithms: FedSGD, FedAvg, FedProx, and SCAFFOLD.

To this end, we conducted extensive experiments to evaluate the performance of these algorithms. Our research reveals several key findings. First, FedSGD, while straightforward in implementation, exhibits significant stability and accuracy issues due to its inability to effectively handle data heterogeneity. This limitation is particularly pronounced in non-IID settings, where the model's performance deteriorates noticeably. In contrast, FedAvg shows improved performance by allowing multiple local iterations before aggregating the model updates, thereby reducing communication overhead and enhancing overall accuracy. FedProx further refines this approach by introducing a regularization term that mitigates the adverse effects of data heterogeneity, resulting in better accuracy and stability. SCAFFOLD, which employs control variates to correct the bias in local updates, demonstrates superior performance in our experiments. Its benefits are especially evident with larger data sets and longer training periods, making it a robust choice for complex federated learning environments. Our experimental conclusions indicate that FedProx is the most effective algorithms among those studied, showcasing their potential for practical federated learning applications. These algorithms not only achieve higher accuracy but also offer greater robustness against data distribution challenges.

Looking forward, future research should aim to optimize these algorithms for various data types and extend their applicability across different federated learning scenarios. Additionally, exploring hybrid approaches that combine the strengths of multiple algorithms could further improve the performance and reliability of FL systems. Continued advancements in this field will be crucial for leveraging federated learning in diverse and dynamic real-world environments.

References

[1] McMahan B, Moore E, Ramage D, et al. 2017, Communication-efficient learning of deep networks from decentralized data, Artificial intelligence and statistics. PMLR: 1273-1282.
[2] Li T, Sahu A K, Zaheer M, et al. 2018 Federated Optimization in Heterogeneous Networks, arXiv preprint arXiv:1812.06127.
[3] Karimireddy S P, Kale S, Mohri M, et al. 2020, Scaffold: Stochastic controlled averaging for federated learning, International conference on machine learning. PMLR: 5132-5143.
[4] Xiao H, Rasul K, Vollgraf R. 2017, Fashion-mnist: a novel image data set for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747.
[5] Kairouz P, McMahan H B, Avent B, et al. 2021 Advances and open problems in federated learning. Foundations and trends in machine learning, 14(1–2): 1-210.
[6] Li T, Sahu A K, Talwalkar A, et al. 2020, Federated learning: Challenges, methods, and future directions. IEEE signal processing magazine, 37(3): 50-60.
[7] Wang H, Kaplan Z, Niu D, et al. 2020 Optimizing federated learning on non-iid data with reinforcement learning, IEEE INFOCOM 2020-IEEE conference on computer communications. IEEE: 1698-1707.
[8] Bonawitz K, Eichner H, Grieskamp W, et al.2019, Towards federated learning at scale: System design. Proceedings of machine learning and systems, 1: 374-388.
[9] He K, Zhang X, Ren S, et al. 2016 Deep residual learning for image recognition, Proceedings of the IEEE conference on computer vision and pattern recognition: 770-778.
[10] Hochreiter S, Schmidhuber J. 1997 Long short-term memory. Neural computation, 9(8): 1735-1780.