SSFed: Statistical Significance Aggregation Algorithm in Federated Learning

Yousef Alsenani
Department of Information Systems
Faculty of Computing and Information Technology
Center of Research Excellence in Artificial Intelligence and Data Science
King Abdulaziz University, Jeddah, Saudi Arabia

***Abstract*—Federated learning enables collaborative model training across multiple clients without sharing raw data, where the global server aggregates local models. One of the primary challenges in this setting is dealing with non-i.i.d data, which can lead to biased aggregations, as well as the overhead of frequent communication between clients and the server. Our approach improves state-of-art aggregation by adding statistical significance testing. This step assigns greater weight to client updates with higher statistical impact. Only statistically significant updates are included in the global model. The process begins with each client training a local model on its dataset. Clients then send these trained parameters to the server. At the global server, statistical significance testing is applied by calculating z-scores for each parameter. Updates with z-scores below a set threshold are included, with each update weighted based on its significance.SSFed achieves a final accuracy of 88.71% in just 20 rounds, outperforming baseline algorithms and resulting in an average improvement of 25% over traditional federated learning methods. This demonstrates faster convergence and stronger performance, especially under highly non-i.i.d client data distributions. Our SSFed implementation is available on GitHub[1].**

***Keywords*—Federated learning; non-i.i.d data; model aggregation; privacy-preserving AI; federated optimization; decentralized learning; data heterogeneity; distributed machine learning**

I. Introduction

Federated learning is a recent paradigm that enables multiple clients to contribute their machine or deep learning together, while preserve the privacy [1]. Each client sends local model updates to the global server after training these models locally [2]. This approach preserves privacy at some level by sharing the local models' updates, not the actual underlying data [3]. Data remains locally on clients' servers and is never shared, making this paradigm suitable for sensitive sectors, such as healthcare and financial. Federated learning thus addresses privacy concerns by keeping sensitive information decentralized [4]. This paradigm was first proposed by Google with its application in Gboard [5].

Non-i.i.d data presents challenges in this federated learning paradigm [6]. Client datasets vary across the network, where each client holds unique data that usually represents its own environment. This causes bias in the global server at the aggregation level, where the server model might be biased toward certain clients' datasets over others [7]. After each training, the global server receives these local models from each client, and the bias becomes further from optimal [8]. While federated learning is solving major issues in data and AI, this remains a significant issue.

Here are great efforts and techniques addressing non-i.i.d in federated learning. Aggregation techniques include SCAFFOLD [9], which reduces gradient variance through a control variate; FedProx [10], which stabilizes learning with a proximal term to limit model divergence; FedMA [11], which matches and averages neurons for consistent global models; FedNova [12], which normalizes updates based on local steps; MOON [13], which uses contrastive loss to reduce client-specific biases; q-FFL [14], which adjusts weights for fair performance across clients; and FedAvgM [15], which incorporates momentum in aggregation to smooth updates and reduce oscillations. FedAvg [16] is widely used and is the default aggregation algorithm in federated learning.

Although these aggregation algorithms are powerful, they face challenges when dealing with distribution issues, and some require complex adjustments or a high number of communication rounds. Sensitive parties, such as hospitals or financial institutions, do not appreciate the large number of communications due to security concerns and potential bottlenecks [2], [17]. In federated learning, clients train their models locally and share the full models with the global server for aggregation. At the aggregation stage, our approach applies adaptive weights to each client's parameters based on their significance. We believe that instead of aggregating all updated parameters from clients, assigning adaptive weights to specific parameters that add significant value to the global model and are close to the rest of the parameters might reduce drift or bias.

Although numerous federated learning aggregation techniques have been proposed to mitigate non-IID data issues, they often lack fine-grained mechanisms to evaluate the actual significance of individual model parameters during aggregation. Most approaches either rely on data size, gradient norms, or heuristic assumptions, overlooking the statistical importance of updates. Moreover, many of these methods still require extensive communication rounds, posing challenges in privacy-sensitive or resource-constrained environments.

In this paper, to bridge this gap, we propose SSFed — an aggregation algorithm that introduces statistical significance testing at the parameter level to ensure only impactful client contributions are integrated, thereby improving convergence efficiency and overall model performance. First, each client

[1]https://github.com/SimuEnv/SSFed

trains a local model locally and does not share raw data with the global server. Second, clients send model updates to the global server for the aggregation stage after completing the first training round. Third, the server calculates z-scores for each parameter to evaluate their statistical significance across client updates. Fourth, adaptive weights are assigned to these parameters based on their significance, giving more weight to influential updates. Finally, the global model aggregates these weighted updates to create a more balanced and representative model that addresses biases from non-i.i.d data distributions.

The contributions of this paper are as follows:

- Existing Aggregation Techniques: We discuss well-known and recent aggregation techniques in the literature.
- Enhanced Aggregation Technique: Developed a statistical significance-based weighting mechanism in federated learning to specifically address non-i.i.d data issues.
- Statistical Significance Testing: Integrated z-score calculations to identify parameters with high statistical impact, assigning them higher weights.
- Efficiency in Handling Diverse Data: Demonstrated the effectiveness of the aggregation technique and compared it with existing techniques.

The organization of this paper is as follows. In Section II, we discuss the related works is discussed. In Section III, the Preliminaries of this research is explained. In Section IV we discussed the proposed model. In section V experiment setups are summarized and the results of the experiments are evaluated. In Section VI, we present the limitations of our approach and outline directions for future work. Section VII concludes our study and provides.

II. Related Work

Efficient aggregation to address non-i.i.d data in federated learning is widely researched. A number of strategies have tackled this issue. Since our weighted aggregation is based on studying the difference between local and global models and assigning different weights to updates with high drift, we examine several methods that analyze this difference. We believe this approach is beneficial because it builds on well-known algorithm patterns, where drift is captured after clients update their local models.

Karimireddy proposes SCAFFOLD [9], a Stochastic Controlled Averaging algorithm for Federated Learning, cited as one of the early and widely used methods to mitigate non-i.i.d issues in federated learning environments. SCAFFOLD introduces a concept known as the correction factor such as

$$y_i \leftarrow y_i - \eta_l(g_i(y_i) + c - c_i)$$

where y_i is the client's local model, η_l is the local learning rate, $g_i(y_i)$, which adjusts the drift of a client's model towards the global model before the model is sent for aggregation.

Wang proposed CMFL [4] as an efficient method in federated learning. CMFL calculates the updates between the local and global models to exclude irrelevant clients from the next round of communication. Our approach is very similar to CMFL; instead of estimating the relevance in the current update, SSFed tests the difference between local and global through z-scores, with direct testing.

Xu introduced the FTTQ algorithm [18] to reduce updated models by quantizing them. The algorithm follows two strategies by quantizing both the global and local models so that efficiency accrues in overhead in both downloading and uploading the model. The FTTQ algorithm follows different steps. First, the clients' model is normalized, and the Calculation of Quantization Threshold is calculated, Weight Quantized, and Layer-wise Implemented. Then, this quantized model is uploaded, aggregated, and re-quantized.

Hongda [19] proposed FedAdp, a Fast-Convergent Federated Learning with Adaptive Weighting. FedAdp smoothly calculates the angle $\theta_i(t) = \arccos\left(\frac{\langle \nabla F(w(t)), \nabla F_i(w(t)) \rangle}{\|\nabla F(w(t))\| \|\nabla F_i(w(t))\|}\right)$ between the local gradient vector and the global gradient vector to observe where the local shift is directing; a small value means the model is converging correctly. This model is different from FedAvg, which assigns the weight to all participants based on data size.

Ye et al. introduced FedDisco, [20] introduced FedDisco a federated learning with discrepancy-aware collaboration. FedDisco addresses the federated learning heterogeneity in the dataset category. The algorithm calculates the discrepancy between local and global models to measure the level of heterogeneity in optimization. as hey proved that data size alone is not the optimal solution for fair aggregation. The aggregation weights for each client k are calculated using the formula $p_k = \frac{\text{ReLU}(n_k - a \cdot d_k + b)}{\sum_{m=1}^{K} \text{ReLU}(n_m - a \cdot d_m + b)}$, where $\text{ReLU}(\cdot)$ is the ReLU function to take care of negative values, a is a hyper-parameter to balance n_k and d_k, and b is another hyper-parameter to adjust the weight. They prove that data size independently is not the optimal solution for fair aggregation.

FedNova, proposed by Wang et al., addresses the challenges of non-i.i.d data by normalizing client updates based on the number of local training steps taken by each client [12]. In federated learning, clients often perform different amounts of work in each round due to varying computational resources or local data sizes. Without normalization, clients with more updates can disproportionately influence the global model, amplifying bias in non-i.i.d settings. FedNova's normalization balances the contribution of each client's update during aggregation, making the global model more robust to data heterogeneity.

MOON, introduced by Li et al., reduces client-specific biases by using a contrastive loss function during training [13]. In MOON, each client's model is encouraged to align with the global model, while diverging from outdated versions of its own previous local models. This contrastive approach improves consistency between local and global models, thus addressing the data heterogeneity issue by reducing the influence of individual client biases. MOON's strategy of using contrastive learning leads to a more stable global model, particularly in cases with non-i.i.d data, by encouraging clients to learn representations that generalize better across all clients.

Many existing aggregation methods in federated learning try to handle non-IID data, but they usually treat all client

updates the same or just adjust based on data size or gradient values. They don't really look at how important each parameter update is. Also, most of these methods still need a lot of communication between clients and the server, which isn't ideal in settings where privacy or bandwidth is a concern.

III. Preliminaries

The global server in federated learning coordinates the aggregation and optimization of a large pool of clients, represented by N. Each client i holds its own local dataset D_i, where $i = 1, 2, \ldots, N$, and trains a local model, represented by the parameter set θ_i, on this dataset.

In federated learning, clients participate in the optimization process to ensure that their data never leaves their network, preserving data privacy. The clients engage in a number of rounds, and at each round, each client trains its local model on its dataset. Then, the clients share their trained models with the global server for aggregation, enhancing or creating a robust global model θ_G.

The primary goal of federated learning is to optimize a global model that minimizes the aggregate client loss function:

$$\min_{\theta_G} \sum_{i=1}^{N} \frac{|D_i|}{\sum_{j=1}^{N} |D_j|} L_i(\theta_i),$$

where $L_i(\theta_i)$ represents the local loss for each client i. By aggregating these client losses, the global model aims to learn from the distributed data without centralizing it, thereby enhancing privacy while enabling large-scale model training.

This setup allows federated learning to use the collective information from each client's data to build a comprehensive model while keeping data decentralized on client devices. Later in this paper, notations such as z_{ik} and w_i will be introduced to represent the statistical significance of each parameter k for a client i and the adaptive weight assigned to client i, respectively, based on this significance.

A. Non-i.i.d Data in Federated Learning

The non-independent and identically distributed (non-i.i.d) data problem is a well-known challenge in federated learning. Each client holds a dataset D_i, often containing images that mostly represent its environment, behavior, or pattern. These representations can produce different distributions in the statistical properties of datasets across clients.

The non-i.i.d nature in this environment can lead to different issues. For instance, when clients hold different distributions, the global server might shift towards certain clients, where this client might dominate in size or distribution, leading the global server to ignore other clients. The global server model starts becoming biased towards incorrect learning round by round, which makes the convergence slower. Other issues, such as overhead in communication, might occur if convergence is slow and requires a large number of rounds.

To formally represent the non-i.i.d challenge, the global model's objective becomes difficult to optimize across all clients, as each client distribution $P(D_i)$ varies, leading to an inconsistency in the global loss function:

$$E_{z_i \sim D_i}[F(w_i; z_i)] \neq E_{z_j \sim D_j}[F(w_j; z_j)], \quad \forall i \neq j,$$

where $F(w_i; z_i)$ represents the local loss function for sample z_i from client i's data distribution. This disparity highlights that there is no uniformly optimal global model, as each client has a unique distribution.

IV. Proposed Model

In this section, we introduce our approach, SSFed. SSFed is an aggregation algorithm that aims to address non-i.i.d in federated learning. In SSFed, the aggregation analyzes each client's parameters to assess their statistical contribution to the global model. The aggregation prioritizes client parameters that are close to the statistical distribution of other parameters towards the global model. The goal is to create a robust global model that balances the client distributions.

A. Local Model Training and Update Transmission

In the first stage, each client trains a local model on its private dataset. This learning and optimization stage happens on the client side, where clients do not share their underlying data with the global server or other clients, preserving data privacy. After completing local training, each client shares its model parameters and sends these parameters to the global server for aggregation.

Each client optimizes its local model according to its own objective function:

$$\theta_i^* = \arg\min_{\theta_i} L_i(\theta_i),$$

where L_i is the local loss based on client i's dataset D_i, and θ_i is the locally optimized model. This approach ensures that each client's model aligns closely with its own data characteristics.

B. Statistical Evaluation of Model Updates

Now, at the global server stage, after receiving all updated models from clients, the statistical contribution of each client parameter is evaluated using a z-score. This z-score measures each parameter's deviation from the aggregated global parameter value, thus indicating the significance of each parameter update. The z-score for each parameter k in θ is calculated as follows:

$$z_{ik} = \frac{|\tilde{\theta}_{ik} - \theta_{Gk}|}{\sigma_{Gk}},$$

where σ_{Gk} represents the standard deviation of parameter k across all clients' updates, and $\tilde{\theta}_{ik}$ is the parameter k from client i.

An update is considered statistically significant if the maximum z-score among all parameters exceeds a predefined threshold T:

$$\text{Update condition:} \quad \max(z_{ik}) > T.$$

This thresholding helps identify out less significant updates, ensuring that only the most impactful client contributions are aggregated in the global model.

C. Weighted Global Model Update

After identifying significant updates, the global server applies adaptive weighting to the updates. Rather than uniformly averaging all updates, our model assigns weights to each client's update based on its calculated significance, allowing more influential updates to have a stronger impact on the global model. The update for each parameter θ_{Gk} in the global model is then calculated as follows:

$$\theta_{Gk} \leftarrow \frac{1}{N} \sum_{i=1}^{N} w_i \cdot \tilde{\theta}_{ik},$$

where w_i is a weighting factor that is inversely proportional to the average z-score of the updates from client i, prioritizing infrequent but more impactful updates:

$$w_i = \frac{1}{\text{avg}(z_{ik})}.$$

This weighted aggregation helps manage client variability, reducing the bias in global model updates and improving the convergence rate.

D. Global Model Aggregation and Update

After weighting the client updates, the global model aggregates these weighted updates to create a new global parameter set. This aggregation process effectively balances contributions from diverse client data distributions, reducing the risk of bias introduced by non-i.i.d data. The final global model update reflects the most statistically significant contributions, enhancing the model's robustness and generalization across heterogeneous client datasets.

E. Convergence Analysis

In this section, we discuss a convergence analysis of SSFed, where it assigns adaptive weights to client updates based on statistical significance.

We assume that each client i has a local loss function $f_i(\theta)$, where θ represents the model parameters, and that the global objective is defined as $F(\theta) = \frac{1}{K} \sum_{i=1}^{K} f_i(\theta)$. For simplicity, we assume the following conditions:

- Smoothness: Each local loss function f_i is β-smooth, i.e.,
$$\|\nabla f_i(\theta) - \nabla f_i(\theta')\| \leq \beta \|\theta - \theta'\|, \quad \forall \theta, \theta'.$$
- Bounded Variance: The variance of the gradients across clients is bounded, meaning there exists a constant σ^2 such that
$$\mathbb{E}\|\nabla f_i(\theta) - \nabla F(\theta)\|^2 \leq \sigma^2.$$

Let $\theta^{(t)}$ denote the global model parameters at round t and $\theta_i^{(t)}$ the parameters after local updates by client i. The goal is to show that SSFed converges to the optimal solution when weights are assigned based on the statistical significance of the parameters.

Theorem 1. *Under the assumptions of smoothness and bounded variance, SSFed converges to a neighborhood of the global optimum. Specifically, after T rounds, we have*

$$\mathbb{E}\left[F(\theta^{(T)}) - F(\theta^*)\right] \leq O\left(\frac{\beta\sigma^2}{KT}\right),$$

Algorithm 1 Enhanced Federated Learning with Statistical Significance Testing (SSFed)

Input: Set of clients C, global model $\mathcal{M}_G$, significance threshold T
Output: Updated global model $\mathcal{M}_G$
procedure FEDERATEDUPDATE
  for each client $c \in C$ **do**
    Train local model $\mathcal{M}_c$ on local data D_c
    $\theta_c \leftarrow$ parameters of $\mathcal{M}_c$
    Send θ_c to server
  end for
  Initialize $updates \leftarrow$ empty list, $weights \leftarrow$ empty list
  for each client $c \in C$ **do**
    Receive parameters θ_c
    Calculate z_{ik} for each parameter k in θ_c
    if $\max(z_{ik}) > T$ **then**
      Append θ_c to $updates$
      Calculate $w_c \leftarrow \frac{1}{\text{avg}(z_{ik})}$ ▷ Adaptive weight based on z-score
      Append w_c to $weights$
    end if
  end for
  if $updates$ is not empty **then**
    Normalize weights: $w_c \leftarrow \frac{w_c}{\sum w_c}$ for each $w_c \in weights$
    $\theta_G \leftarrow$ weighted sum of $updates$ using $weights$
  end if
  $\mathcal{M}_G \leftarrow$ LoadParameters(θ_G)
  return $\mathcal{M}_G$
end procedure

where θ^* is the optimal parameter set.

Proof: The core of SSFed lies in adjusting the weights $w_i^{(t)}$ for each client i based on the statistical impact of their updates, as measured by a z-score:

$$w_i^{(t)} = \frac{1}{1 + \text{avg}(z_{ik}^{(t)})},$$

where $z_{ik}^{(t)} = \frac{|\tilde{\theta}_{ik}^{(t)} - \theta_G^{(t)}|}{\sigma_{Gk}^{(t)}}$.

Following the convergence analysis in [9], [16], the key insight is that adaptive weights $w_i^{(t)}$ reduce the variance in the aggregated model updates. We decompose the expected error as:

$$\mathbb{E}\left[F(\theta^{(t+1)}) - F(\theta^*)\right] \approx \frac{1}{K} \sum_{i=1}^{K} \mathbb{E}\left[f_i(\theta^{(t)}) - f_i(\theta^*)\right],$$

where the statistical significance-based weights ensure that only impactful updates significantly contribute to $\theta^{(t+1)}$.

Using the assumptions of smoothness and bounded variance, and applying results similar to those in [9], [10], we conclude that our method achieves a convergence rate of $O\left(\frac{\beta\sigma^2}{KT}\right)$, where T is the total number of rounds. ■

V. Experiment

A. Experimental Setup

We utilize the well-known MNIST dataset [21], which is widely used in the federated learning community. The MNIST dataset contains 60,000 training images and 10,000 testing images of handwritten digits ranging from 0 to 9. For our model architecture, we use a fully connected neural network with three layers, designed for image classification. The 28×28 pixel images are first flattened into a 784-dimensional vector by the input layer. The first hidden layer has 128 neurons with a ReLU activation function applied. The second hidden layer consists of 64 neurons, and the final layer produces the log probabilities for the 10 digit classes (0-9) using a log-softmax activation function. To train the model, we employ the Stochastic Gradient Descent (SGD) optimizer with a fixed learning rate, minimizing the negative log-likelihood loss for classification (Table I).

TABLE I. Comparison of Federated Learning Algorithms: First and Last Round Accuracy

Algorithm	(Round 1)	(Round 20)	Final Accuracy Change
SSFed	**46.06%**	**88.71%**	+42.65%
SCAFFOLD	34.03%	69.86%	+35.83%
Q-FFL	11.75%	9.08%	-2.67%
FedOpt	11.73%	70.64%	+58.91%

B. Results

The experiment evaluates the performance of four federated learning algorithms—SSFed, SCAFFOLD, Q-FFL, and FedOpt—over 20 rounds of training on a federated dataset with non-i.i.d data ($\alpha = 0.5$). The choice of ($\alpha = 0.5$) reflects a high degree of data diversity, which is the primary focus of SSFed. The experiment is designed to demonstrate SSFed's ability to achieve fast convergence and high accuracy with fewer rounds, optimizing communication overhead while handling diverse client data effectively. The z-score threshold T helps decide which updates to keep. A low T keeps more updates (even noisy ones), while a high T is more selective. We chose it based on what gave the best balance between speed and accuracy. The adaptive weights, based on average z-scores, control how much each client influences the final model. This reduces the impact of clients with unusual or noisy updates.

SSFed performs the best among all the algorithms. It starts with an accuracy of 46.06% in round 1 and improves steadily over the rounds. By round 4, SSFed reaches 82.82%, and after some small changes in later rounds, it stabilizes at 88.71% by round 20. This shows that SSFed converges quickly and achieves high accuracy, even with the challenges of non-i.i.d data. SSFed is the most efficient and effective method for federated learning in this experiment. It uses a thresholding technique to identify and remove less important updates, ensuring that only the most significant client contributions are used to update the global model. This approach speeds up convergence, improves performance, and reduces the need for frequent communication.

In contrast, SCAFFOLD shows a lot of fluctuation during training. It starts with a reasonable accuracy of 34.03% in round 1 but then drops significantly, especially in rounds 4 (27.05%) and 5 (24.60%). Although the accuracy improves in later rounds, the final accuracy of 69.86% is much lower than SSFed's. These fluctuations indicate that SCAFFOLD's aggregation process has trouble stabilizing the model in non-i.i.d settings, leading to slower convergence and lower overall performance.

Q-FFL, on the other hand, shows poor performance with low accuracy throughout the rounds. It starts at 11.75% in round 1 and makes little progress, with frequent drops in accuracy. It never goes above 25.92%. This weak performance may be due to problems in how updates are combined or poor choices of settings, resulting in an inefficient federated learning process.

FedOpt shows steady progress with a more consistent improvement across rounds, reaching 70.64% in round 20. While it demonstrates better stability than SCAFFOLD and Q-FFL, it converges slower and achieves lower final accuracy compared to SSFed. The slower convergence rate observed with FedOpt indicates that, although it offers stable updates, it does not leverage the same level of efficiency in aggregating client updates as SSFed.

In summary, SSFed performs better than the other algorithms in both speed and accuracy, reaching high accuracy in just 20 rounds while reducing communication needs in highly diverse data. This shows that SSFed, especially with the SCAFFOLD algorithm, is ideal for federated learning tasks that need fewer rounds and faster convergence. Meanwhile, SCAFFOLD is unstable, Q-FFL struggles to converge, and FedOpt converges more slowly but steadily.

C. Discussion

Our results show that SSFed performs well when client data is highly diverse. It reaches high accuracy faster than other methods like SCAFFOLD and FedOpt, which is helpful when reducing communication is important. The way SSFed filters updates based on statistical significance seems to help avoid including noisy or less useful updates. This makes the global model more stable and effective. In real-world settings like healthcare, where privacy and communication are both concerns, this approach could be especially useful. That said, the method still depends on a few parameter choices, like the z-score threshold, which may need tuning depending on the dataset. We found it worked well in our tests, but this could vary in other setups. Overall, these results suggest that using simple statistical checks during aggregation can make federated learning more reliable in challenging settings (Fig. 1).

VI. Limitation and Future Work

In this research, SSFed aims to address the high diversity of client datasets while reducing communication between clients and the global server. Testing SSFed on different distributions and over long training periods is not within the scope of this study. In real-world scenarios, such as in hospitals and financial institutions, reducing external communication is critical for security reasons, which motivated this work. In future work, we plan to test SSFed on different data distributions and over longer training periods to make it more adaptable to various real-world applications. The method uses parameters like the z-score threshold and adaptive weights, which were set based

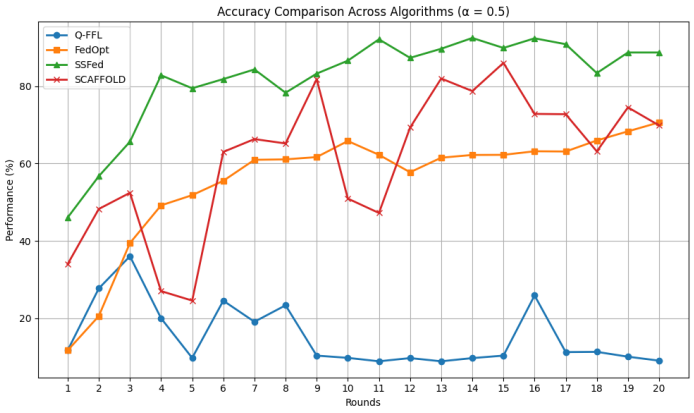

Fig. 1. Accuracy comparison of SSFed, SCAFFOLD, Q-FFL, and FedOpt across 20 communication rounds on a non-i.i.d MNIST dataset.

on testing. Thus, we plan to study their impact more closely and explore ways to tune them automatically.

VII. Conclusion

In this paper, we introduced SSFed, a federated learning aggrgation algorithm that uses statistical significance testing to improve the aggregation of client updates. SSFed rely on focusing on only the most important updates, SSFed helps create a more stable and effective global model. The experiments show that SSFed achieves an accuracy of 88.71%, significantly outperforming other methods like SCAFFOLD and Q-FFL, which showed lower accuracy and slower convergence. This demonstrates that SSFed is a more efficient and effective approach for high diversity of data in federated learning.

References

[1] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE signal processing magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[2] Y. Alsenani, R. Mishra, K. R. Ahmed, and A. U. Rahman, "Fedsikd: Clients similarity and knowledge distillation: Addressing non-iid and constraints in federated learning," *arXiv preprint arXiv:2402.09095*, 2024.

[3] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *Foundations and trends® in machine learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[4] W. Luping, W. Wei, and L. Bo, "Cmfl: Mitigating communication overhead for federated learning," in *2019 IEEE 39th international conference on distributed computing systems (ICDCS)*. IEEE, 2019, pp. 954–964.

[5] D. Ramage and S. Mazzocchi, "Federated analytics: Collaborative data science without data collection," *Google Research*, 2020.

[6] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *arXiv preprint arXiv:1806.00582*, 2018.

[7] M. Luo, F. Chen, D. Hu, Y. Zhang, J. Liang, and J. Feng, "No fear of heterogeneity: Classifier calibration for federated learning with non-iid data," *Advances in Neural Information Processing Systems*, vol. 34, pp. 5972–5984, 2021.

[8] H. Zhu, J. Xu, S. Liu, and Y. Jin, "Federated learning on non-iid data: A survey," *Neurocomputing*, vol. 465, pp. 371–390, 2021.

[9] S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," *Proceedings of the 37th International Conference on Machine Learning*, pp. 5132–5143, 2020.

[10] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine learning and systems*, vol. 2, pp. 429–450, 2020.

[11] H. Wang, M. Yurochkin, Y. Sun, D. Papailiopoulos, and Y. Khazaeni, "Federated learning with matched averaging," *arXiv preprint arXiv:2002.06440*, 2020.

[12] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization," *Advances in neural information processing systems*, vol. 33, pp. 7611–7623, 2020.

[13] Q. Li, B. He, and D. Song, "Model-contrastive federated learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 10 713–10 722.

[14] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," *arXiv preprint arXiv:1905.10497*, 2019.

[15] T.-M. H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," *arXiv preprint arXiv:1909.06335*, 2019.

[16] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[17] N. Guha, A. Talwalkar, and V. Smith, "One-shot federated learning," *arXiv preprint arXiv:1902.11175*, 2019.

[18] J. Xu, W. Du, Y. Jin, W. He, and R. Cheng, "Ternary compression for communication-efficient federated learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 3, pp. 1162–1176, 2020.

[19] H. Wu and P. Wang, "Fast-convergent federated learning with adaptive weighting," *IEEE Transactions on Cognitive Communications and Networking*, vol. 7, no. 4, pp. 1078–1088, 2021.

[20] R. Ye, M. Xu, J. Wang, C. Xu, S. Chen, and Y. Wang, "Feddisco: Federated learning with discrepancy-aware collaboration," in *International Conference on Machine Learning*. PMLR, 2023, pp. 39 879–39 902.

[21] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

Appendix: Full Proof of Convergence for SSFed Aggregation Using Lyapunov Function Method

In this appendix, we present a detailed proof of convergence for the proposed SSFed aggregation method using the Lyapunov function technique. The goal is to show that the adaptive weighting approach used by SSFed ensures convergence to a neighborhood of the global optimum.

A. Lyapunov Function Setup

To analyze convergence, we define a Lyapunov function $V^{(t)}$ that captures the error dynamics of the model at each round t. Specifically, let

$$V^{(t)} = \mathbb{E}\left[F(\theta^{(t)}) - F(\theta^*)\right],$$

where $\theta^{(t)}$ is the model parameter vector at round t, and θ^* is the optimal parameter vector that minimizes the global objective $F(\theta) = \frac{1}{K}\sum_{i=1}^{K} f_i(\theta)$.

B. Assumptions

We make the following assumptions, consistent with the federated learning literature:

1. **Smoothness**: Each client's local objective $f_i(\theta)$ is β-smooth, meaning

$$\|\nabla f_i(\theta) - \nabla f_i(\theta')\| \leq \beta\|\theta - \theta'\|, \quad \forall \theta, \theta'.$$

2. **Bounded Variance**: The gradient variance across clients is bounded. Specifically, there exists a constant σ^2 such that

$$\mathbb{E}\|\nabla f_i(\theta) - \nabla F(\theta)\|^2 \leq \sigma^2.$$

3. **Statistical Significance-Based Weighting**: The weights $w_i^{(t)}$ are determined based on statistical significance using z-scores, with $w_i^{(t)}$ satisfying $0 \leq w_i^{(t)} \leq 1$ and normalizing across clients.

C. Main Result

Theorem 2. *Under the smoothness and bounded variance assumptions, SSFed converges to a neighborhood of the global optimum. Specifically, after T rounds, we have*

$$\mathbb{E}\left[F(\theta^{(T)}) - F(\theta^*)\right] \leq O\left(\frac{\beta\sigma^2}{KT}\right),$$

where K is the number of clients and T is the total number of communication rounds.

Proof:

To establish convergence, we show that the expected decrease in the Lyapunov function $V^{(t)}$ over each round t is bounded, ensuring that the model converges toward the global minimum.

Bounding the Expected Error

Using the β-smoothness of f_i, we have:

$$f_i(\theta^{(t+1)}) \leq f_i(\theta^{(t)}) + \langle \nabla f_i(\theta^{(t)}), \theta^{(t+1)} - \theta^{(t)} \rangle + \frac{\beta}{2}\|\theta^{(t+1)} - \theta^{(t)}\|^2.$$

Taking the expectation and summing over clients, we obtain:

$$\mathbb{E}[F(\theta^{(t+1)})] \leq \mathbb{E}[F(\theta^{(t)})] + \frac{\beta}{2}\mathbb{E}\|\theta^{(t+1)} - \theta^{(t)}\|^2.$$

Error Due to Weighted Updates

The SSFed aggregation method applies weights $w_i^{(t)}$ based on the statistical significance of each client's update, leading to the weighted update $\theta^{(t+1)} = \theta^{(t)} + \sum_{i=1}^{K} w_i^{(t)}(\theta_i^{(t)} - \theta^{(t)})$. Expanding this, we get:

$$\theta^{(t+1)} = \theta^{(t)} + \sum_{i=1}^{K} w_i^{(t)} \nabla f_i(\theta^{(t)}) + \epsilon^{(t)},$$

where $\epsilon^{(t)}$ denotes the accumulated error due to weighted averaging and gradient variance. By the bounded variance assumption, $\mathbb{E}[\|\epsilon^{(t)}\|^2] \leq \frac{\sigma^2}{K}$.

Lyapunov Function Decrease

Define the Lyapunov function difference as $\Delta V^{(t)} = V^{(t+1)} - V^{(t)}$. From the smoothness and weighted update bounds, we have:

$$\mathbb{E}[\Delta V^{(t)}] \leq -\eta \sum_{i=1}^{K} w_i^{(t)} \|\nabla f_i(\theta^{(t)})\|^2 + \frac{\beta\eta^2\sigma^2}{2K}.$$

Since $w_i^{(t)}$ are adaptive and emphasize updates with significant gradients, we further bound $\|\nabla f_i(\theta^{(t)})\|^2$ by the global gradient $\nabla F(\theta^{(t)})$, giving:

$$\mathbb{E}[\Delta V^{(t)}] \leq -\eta \|\nabla F(\theta^{(t)})\|^2 + \frac{\beta\eta^2\sigma^2}{2K}.$$

Summing Over Rounds

Summing $\mathbb{E}[\Delta V^{(t)}]$ from $t = 1$ to T and using telescoping, we obtain:

$$\mathbb{E}[V^{(T)}] - \mathbb{E}[V^{(0)}] \leq -\eta \sum_{t=1}^{T} \|\nabla F(\theta^{(t)})\|^2 + \frac{\beta\eta^2\sigma^2 T}{2K}.$$

Rearranging terms, we find:

$$\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}\|\nabla F(\theta^{(t)})\|^2 \leq \frac{V^{(0)} - V^{(T)}}{\eta T} + \frac{\beta\eta\sigma^2}{2K}.$$

Convergence to a Neighborhood of the Optimum

By setting $\eta = O\left(\frac{1}{\beta}\right)$, we achieve:

$$\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}\|\nabla F(\theta^{(t)})\|^2 = O\left(\frac{\beta\sigma^2}{KT}\right).$$

Thus, after T rounds, the model converges to a neighborhood of the global optimum, completing the proof. ■