ISSN 2723-6471

Enhancing Federated Learning Performance through Adaptive Client Optimization with Hyperparameter Tuning

Made Adi Paramartha Putra[1,*], Komang Ram Pramartha Utama[2], Nengah Widya Utami[3], I Gede Juliana Eka Putra[4]

[1,2,3,4]*Faculty of Information Technology and Design, Primakara University, Denpasar, Indonesia*

(Received: February 25, 2024; Revised: March 29, 2024; Accepted: April 29, 2024; Available online: May 31, 2024)

Abstract

The effectiveness of Industrial Internet of Things (IIoT) systems requires a robust fault detection mechanism, a task effectively accomplished by leveraging Artificial Intelligence (AI). However, the current centralized learning approach proves inadequate. In response to this limitation, Federated Learning (FL) enables decentralized training, ensuring the protection of individual data. The traditional FL settings are not sufficient to provide an effective learning process, which needs to be refined. This paper introduces an Adaptive Distributed Client Training (ADCT) mechanism designed to optimize performance for each FL participant, thereby establishing an efficient and resilient system. The proposed ADCT utilizes two parameters, namely the accuracy threshold and grid search step, to find the optimal hyperparameter for each client in a specific number of federation rounds. The evaluation results, conducted using the MNIST and FMNIST datasets in non-IID settings, indicate that the proposed ADCT enhances the F1-score by up to 37.13% compared to state-of-the-art methods.

Keywords: Adaptive Training, Federated Learning, Grid Search, Hyperparameter Optimization, Non-IID

1. Introduction

In recent decades, the progress of AI has brought about numerous advantages, enhancing the quality of services across various domains [1], [2]. AI expedites decision-making by inputting information into the machine in a format consistent with its training. Currently, the predominant approach involves centralized training, where all information resides on a powerful single computer, and training is conducted locally. While potent, this approach has significant drawbacks, including data leakage and susceptibility to a single point of failure, which can lead to a system malfunction.

To overcome these issues, FL has emerged as a decentralized training method, aiming to rectify the limitations of centralized learning [3]. FL enables distributed training on client devices, ensuring privacy and robust model creation without exposing client data. The effectiveness of FL methodologies heavily depends on the careful selection of suitable clients to enable swift model convergence [4]. However, FL must consider factors such as device diversity, security, and data distribution to maximize efficiency. Techniques like client selection and local training optimization prove valuable in enhancing FL performance.

Referring to one of the popular technologies in recent years, the metaverse, where users engage with diverse scenarios and environments [5], FL's capability to distribute learning across different devices without consolidating sensitive data becomes especially advantageous. The creation of personalized avatars, reflecting users' preferences and behaviors, demands advanced AI models capable of adjusting to individualized training requirements. FL's ability to customize training on the client side plays a crucial role in developing more precise and responsive avatars, ultimately enriching the overall user experience [6].

The local training optimization is a crucial aspect of improving the FL process. Configuring each local server properly leads to enhanced global performance. Various optimization methods have been explored, including Two-Stage Federated Optimization Algorithm (TSFOA), which dynamically assigns model weights based on data distribution

***Corresponding author: Made Adi Paramartha Putra (adi@primakara.ac.id)**
DOI: https://doi.org/10.47738/jads.v5i2.251

among all FL participants [7]. Additionally, a technique called AMBLE, which adaptively adjusts mini-batch size and local epochs, has been introduced to enhance FL performance, especially in non-Independent and Identically Distributed (non-IID) settings [8].

Recognizing the significance of local training optimization, it is essential to emphasize that incorrect configurations can result in longer training times and increased computing resource demands. To address and reduce processing time, our focus in this article is on considering only two parameters, outlining the primary contribution of our work as follows:

1) We suggest the implementation of ADCT as a means to customize training for individual clients, thereby elevating their performance. ADCT utilizes hyperparameter tuning optimization to enhance the local model performance and, consequently, the overall performance of the FL model.
2) An ablation study of the proposed ADCT is conducted to obtain the best configuration, resulting in a more efficient learning process.
3) We extensively assess the performance of FL using the MNIST and FMNIST dataset in a non-IID configuration to showcase the effectiveness of the proposed method in comparison to state-of-the-art FL methods.

The rest of this research is structured as follows: In Section II, prior studies on FL optimization, specifically local training, are reviewed. Section III outlines the proposed system model in detail. The performance evaluation of the proposed model, along with its counterpart, is presented in Section IV. Finally, Section V concludes this work and provides some future directions.

2. Related Works

FL optimization can be divided into several aspect, such as client selection [9], local optimization, as well as the security enhancement. The local training process on each selected client is a crucial factor in ensuring the performance of FL. Considering participant device heterogeneity, computing resources, and dataset distribution vary widely, especially in non-IID scenarios. Furthermore, the phenomenon of client drift, where local model updates tend towards a local optimal solution, can negatively impact the performance of the global model.

Addressing client drift issues involves employing model aggregation and model training methods [10]. FedAvgM [11] introduces momentum on top of Stochastic Gradient Descent (SGD) for model aggregation, a concept similarly utilized in FedMIM [12]. FedMIM incorporates weighted global gradient estimation, contributing to global objective optimization across all FL participants. Another approach, FedNova, normalizes and scales local updates based on the number of local epochs before aggregation [13]. All four systems: FedAvgM, FedMIM, FedNova, and adaptive bias estimation [14] demonstrate faster convergence compared to the traditional FedAvg.

In terms of model training, the TSFOA method [7] involves adaptively assigning model weights by considering differences in data distribution among FL participants. Results with the non-IID MNIST dataset indicate that TSFOA outperforms FedAvg in convergence time. Adaptive optimization methods for server updates, introduced in AdaGrad [15], are complemented by local optimization methods in [16], demonstrating faster convergence with local adaptivity. Local hyperparameter optimization in FL includes dynamic learning rates to adapt to wireless computation environments, leading to higher accuracy with MNIST and CIFAR datasets. Techniques such as Adaptive FL Dropout (AFD) [17] and FedDUAP [18], involving dynamic updates and adaptive pruning, minimize communication costs and enhance efficiency. AMBLE [8] dynamically adjusts mini-batch size, local epochs, and learning rate, achieving faster convergence in both IID and non-IID settings.

Furthermore, an algorithm in [19] considers client heterogeneity factors (computing power, storage, and network) to enhance local convergence, defining criteria to stop local training once met. Results show the superiority of this optimization technique over the baseline FL. State-of-the-art studies emphasize the importance of hyperparameter settings for individual FL clients in enhancing system performance. However, excessive tuning may increase computing resource consumption, necessitating the establishment of appropriate criteria for hyperparameter configurations, considering overall computing costs.

 ISSN 2723-6471

While numerous researchers aim to enhance the performance of all FL participants, this pursuit can result in increased computing costs, given the need for multiple iterations of adaptive local training. Presently, the incorporation of local client optimization often leads to prolonged training times due to the iterative nature of local training. Therefore, it becomes imperative to decrease the total number of iterations conducted, striking a balance between minimizing costs and delivering an optimal model for the overall FL process.

3. Proposed System

In this section, the proposed ADCT process is detailed, presenting both the workflow diagram and pseudocode. Subsequently, we discuss the simulation details and evaluation metrics employed to compare the proposed work.

3.1. Adaptive Distributed Client Training

ADCT is employed for the dynamic optimization of the training process on individual FL clients. The parameters introduced by ADCT (γ, s) consist of γ for the accuracy threshold and s for the grid search step. Together, these parameters regulate the scope of the grid search process every federation round. The workflow of the proposed ADCT is detailed in figure 1.

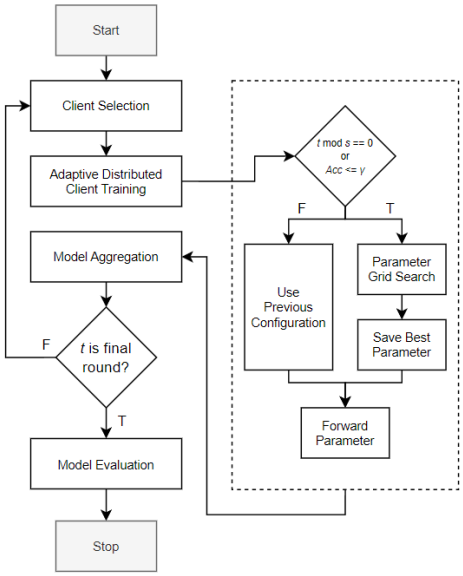

Figure 1. Workflow of the proposed ADCT inserted in the traditional FL system

Initially, similar to traditional FL, client selection is performed to choose an appropriate participant for the federation round. The grid search is then conducted based on two parameters in ADCT, γ and s, as detailed previously. If the conditions are met, the grid search is executed; otherwise, the previous hyperparameter configuration is retained. To better understand the concept of the proposed ADCT, the pseudocode is detailed in figure 2.

 ISSN 2723-6471

Algorithm 1 Adaptive Distributed Client Training

Initialize FL parameter and global model ω_l.
Initialize list of grid search parameters *grid_list*.

FL Server executes:
for each communication round t **do**
  Select client for training.
  for each client in parallel **do**
    $ADCT(\omega_g, t, \overline{Acc})$
  end for
  $\omega_k \leftarrow$ collect parameter from all clients.
  $\overline{Acc} \leftarrow$ calculate the average accuracy of all clients.
  $\omega_{g_{t+1}} \leftarrow$ aggregate new global parameter.
  $Acc, F_1 \leftarrow$ evaluate the updated global parameter.
end for

Function $ADCT(\omega_c, t, \overline{Acc})$
$\omega_l \leftarrow$ Collect ω_g from FL server.
if $Acc \leq \overline{Acc} \,||\, t \; mod \; s \; == \; 0$ **then**
  for *epoch*, *lr* in *grid_list* **do**
    $Acc \leftarrow$ evaluate the updated local model.
    if $Acc \geq \overline{Acc}$ **then**
      break
    end if
  end for
end if
$\omega_l \leftarrow$ best performing local model.
Store updated local parameter ω_l, t to FL server.

Figure 2. The overall algorithm of the proposed ADCT.

Moreover, it is worth noting that the grid search method stands out as a widely embraced technique for hyperparameter optimization in AI models. Frequently employed, this method seeks the most effective combination of hyperparameters, aiming to achieve optimal performance for a given system. In this work, we consider two hyperparameters as the main goals, namely the learning rate η and local epochs e_l.

In total, six different configurations of hyperparameters can be adopted by each FL participant to deliver better learning performance. The comprehensive details of the hyperparameter configuration are provided in table 1.

Table 1. Local hyperparameter configuration

Parameter	Value
Learning rate	0.00001, 0.0001, 0.001
Epochs	1, 2
Batch Size	32
Early Stopping	Accuracy (*patience =1)*
Data Augmentation	None

3.2. Simulation Details

The Flower framework [20] is employed to establish the FL environment. Flower allows for the creation of customizable and adaptable FL systems tailored to specific use cases and optimization requirements. Developed using the Python programming language, Flower can be tailored to align with the architecture proposed in this research. Regarding available modules and aggregation strategies, Flower predominantly offers FedAvg as its principal aggregation technique.

In this work, we explore two types of datasets with different task complexities. First, we utilized MNIST, which represents handwritten digits with 10 classes. The next dataset is FMNIST, representing fashion image classification, also with 10 classes. Both datasets used in this work are popular for the FL system evaluation process.

To mimic real-world application use cases, the datasets are divided into non-IID settings. Each client is configured to have only one class of data, resulting in a prolonged learning process. The illustration of the data allocation for the MNIST and FMNIST dataset is shown in figure 3 and figure 4, respectively.

 ISSN 2723-6471

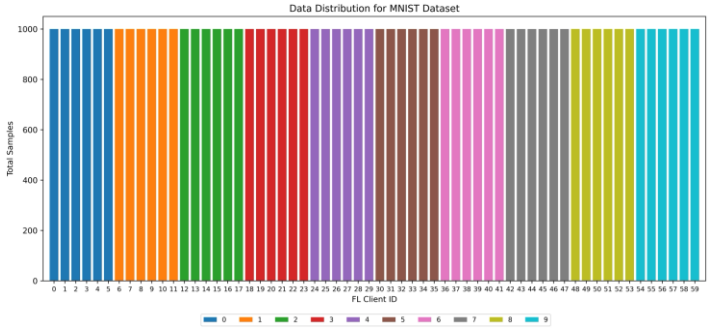

Figure 3. Non-IID settings for multiple clients based on the MNIST dataset

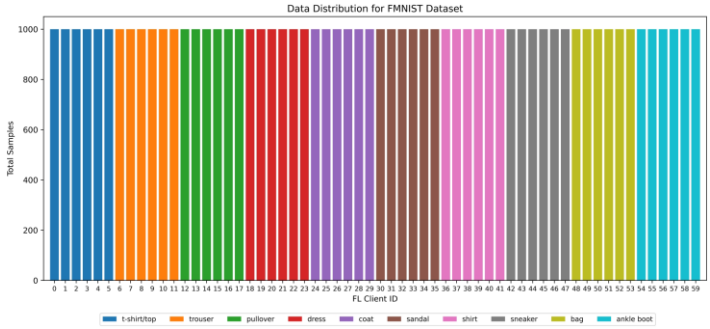

Figure 4. Non-IID settings for multiple clients based on the FMNIST dataset

4. Result and Discussion

In this section, the evaluation of the proposed ADCT is conducted. The performance is compared with state-of-the-art methods such as FedAvg [3], FedMedian [21], FedACS [9], FedYogi [22], and FedOpt [22]. Firstly, the final model accuracy at the end of the federation process is investigated. Subsequently, the total rounds required to achieve the target accuracy are also examined.

4.1. Model Accuracy

For the initial evaluation conducted using the MNIST dataset via an ablation study, the accuracy threshold and grid search step were varied. The results are shown in figure 5. Over a total of 50 communication rounds and nine variations, the highest performance was achieved by the ADCT configuration with an accuracy threshold of 50 and a grid search step of 5. Subsequently, similar performance was obtained using ADCT (25,15) and ADCT (50,15).

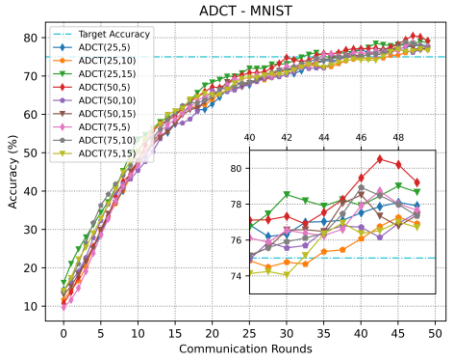

Figure 5. Ablation study of the proposed ADCT using MNIST dataset

For the subsequent dataset used in the evaluation, we employed FMNIST. This dataset is more complex compared to the MNIST dataset. Similarly, the evaluation was conducted using nine variations of ADCT, with variations in the accuracy threshold and grid search steps. The results depicted in Figure 6 indicate that ADCT with an accuracy

ISSN 2723-6471

threshold of 75 and a grid search step of 15 provides the best performance in terms of accuracy. This is followed by ADCT (75,5) and ADCT (50,10) in the second and third place, respectively.

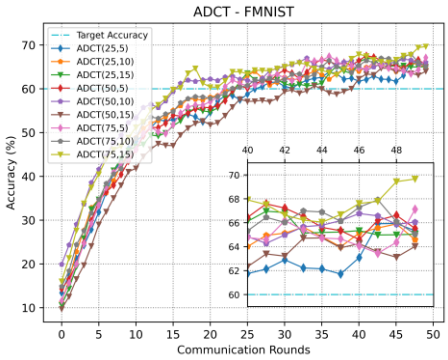

Figure 6. Ablation study of the proposed ADCT using FMNIST dataset.

To better comprehend the performance of all variations of ADCT evaluated in this study, table 2 presents the accuracy of each setting in two different datasets. Similarly, the performance depicted in the line graphs (figure 5 and figure 6) is aligned. The results show that the accuracy for MNIST and FMNIST is 78.75% and 69.35%, respectively.

Table 2. Accuracy comparison among various accuracy threshold and grid search steps in the proposed ADCT

Variations (xxx)	Accuracy (%)	
	MNIST	FMNIST
Baseline	62.04	57.11
ADCT (25, 5)	77.75	63.23
ADCT (25, 10)	76.62	63.71
ADCT (25, 15)	78.40	65.02
ADCT (50, 5)	78.75	64.97
ADCT (50, 10)	77.60	66.28
ADCT (50, 15)	77.77	64.57
ADCT (75, 5)	77.66	68.46
ADCT (75, 10)	77.00	64.91
ADCT (75, 15)	76.44	69.35

4.2. Round to Achieve Target Accuracy

To validate the previously mentioned evaluation metrics, the calculation of the rounds required to achieve the target accuracy is also conducted during the evaluation. In this study, a total of 50 rounds are considered for both the MNIST and FMNIST datasets. Achieving the target accuracy with a lower number of rounds is crucial to enhance the efficiency of the FL system. Table 3 displays the total number of rounds required by each variation of the proposed ADCT to achieve the target accuracy.

Table 3. Round to achieve specific target accuracy comparison among various accuracy threshold and grid search steps in the proposed ADCT

Variations (xxx)	Accuracy (%)	
	MNIST	FMNIST
Baseline	n/a	23
ADCT (25, 5)	39	27
ADCT (25, 10)	41	23

ADCT (25, 15)	33	25
ADCT (50, 5)	31	24
ADCT (50, 10)	40	17
ADCT (50, 15)	37	32
ADCT (75, 5)	36	23
ADCT (75, 10)	39	24
ADCT (75, 15)	40	16

For MNIST, the target is set to 75%, while for FMNIST, it is configured to 60% due to the larger complexity of the dataset. Moreover, the results show that the findings from the previous section align with the total rounds needed to achieve the target accuracy. This is evidenced by the optimal setting for ADCT, which only takes 31 rounds to achieve the target accuracy in the MNIST dataset with an accuracy threshold of 50 and a grid search step of 5. Similarly, the most efficient configuration for the FMNIST dataset is using ADCT (75,15), which only requires 16 federation rounds.

4.3. Comparative Analysis

To demonstrate the improvement of the proposed ADCT compared to its counterparts, the F1-score of various FL techniques with similar learning configurations was assessed. In total, six different state-of-the-art methods, such as FedAvg, FedACS, FedMedian, FedYogi, and FedOpt, are considered for the comparative analysis along with the proposed ADCT. The performance of target accuracy under two datasets depicted in table 4.

Table 4. Round to achieve specific target accuracy comparison among various accuracy threshold and grid search steps in the proposed ADCT

ADCT Final Configuration	
MNIST	**FMNIST**
ADCT (50, 5)	ADCT (75,15)

It is worth noting that the configuration for MNIST and FMNIST differs in the proposed ADCT. This variation is attributed to the differences in dataset difficulty and complexity. Based on the ablation study detailed earlier, the optimal configuration of ADCT used in the comparative analysis is outlined in Table 4. For the MNIST dataset, ADCT (50, 5) is utilized, while for FMNIST, the configuration of ADCT (75,15) is applied. The detailed comparative performance evaluation using MNIST and FMNIST datasets in terms of F1-score is presented in table 5.

Table 5. Comparative performance evaluation of the proposed ADCT and the state-of-the-art FL techniques, assessed using three different datasets under non-IID settings, in terms of F1-score

Techniques	**F1-Score (%)**	
	MNIST	**FMNIST**
FedAvg [3]	64.15	53.54
FedACS [9]	66.87	36.18
FedMedian [21]	35.74	21.33
FedYogi [22]	67.23	55.71
FedOpt [22]	68.23	54.98
Proposed ADCT	70.62	58.46

As detailed earlier, the performance evaluation is conducted using non-IID settings with the same distribution for each FL technique. For the MNIST dataset, the proposed ADCT achieved the highest performance with an F1-score of 70.62%. This value is 2.39% to 34.88% better compared to state-of-the-art techniques. Similarly, for the FMNIST dataset, the best performance was attained by the proposed ADCT, yielding a total F1-score of 58.46% at the end of the learning process. In summary, the proposed ADCT exhibited a performance improvement ranging from 3.48% to 37.13% for the FMNIST dataset when compared to its counterparts.

5. Conclusion

The paper introduces an optimization approach to achieve adaptive and high-performance model training through distributed client training. This method takes into account hyperparameters such as learning rate and local iteration during the grid search process. The effectiveness of this proposed mechanism is assessed using the MNIST and FMNIST datasets in non-IID settings. The outcomes distinctly indicate that ADCT surpasses state-of-the-art methods in terms of accuracy and F1-score, demonstrating performance enhancements of up to 34.88% and 37.13% for the MNIST and FMNIST datasets, respectively. While the proposed ADCT introduced in this work demonstrates superiority, there is still room for further improvement in the FL system. For instance, potential enhancements include designing a novel aggregation scenario, considering device resources for client selection, and mitigating single points of failure by adopting decentralized FL.

6. Declarations

6.1. Author Contributions

Conceptualization: M.A.P.P., K.R.P.U., N.W.U., and I.G.J.E.P.; Methodology: M.A.P.P.; Software: M.A.P.P.; Validation: M.A.P.P., K.R.P.U., N.W.U., and I.G.J.E.P.; Formal Analysis: M.A.P.P., K.R.P.U., N.W.U., and I.G.J.E.P.; Investigation: M.A.P.P.; Resources: K.R.P.U.; Data Curation: N.W.U.; Writing Original Draft Preparation: M.A.P.P. and I.G.J.E.P.; Writing Review and Editing: M.A.P.P. and I.G.J.E.P.; Visualization: I.G.J.E.P.; All authors have read and agreed to the published version of the manuscript.

6.2. Data Availability Statement

The data presented in this study are available on request from the corresponding author.

6.3. Funding

The authors received no financial support for the research, authorship, and/or publication of this article.

6.4. Institutional Review Board Statement

Not applicable.

6.5. Informed Consent Statement

Not applicable.

6.6. Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] M. A. P. Putra, A. P. Hermawan, D.-S. Kim, and J.-M. Lee, "Data prediction-based energy- efficient architecture for industrial iot," *IEEE Sensors Journal*, vol. 23, no. 14, pp. 15856–15866, 2023. DOI: 10.1109 / JSEN.2023.3280485.

[2] S. M. Rachmawati, M. A. P. Putra, J. M. Lee, and D. S. Kim, "Digital twin-enabled 3d printer fault detection for smart additive manufacturing," *Engineering Applications of Artificial Intelligence*, vol. 124, no. 1, pp. 106 430, 2023, ISSN: 0952-1976. DOI: https://doi.org/10.1016/j.engappai.2023.106430.

[3] H. B. McMahan, E. Moore, D. Ramage, and B. A. y Arcas, "Federated learning of deep networks using model averaging," *CoRR*, vol. abs/1602.05629, no. 1, pp. 1 2016. arXiv: 1602.05629.

[4] M. A. P. Putra, G. A. Sampedro, D.-S. Kim, and J.-M. Lee, "Ecsm: An ensembled client selection mechanism for efficient federated learning," *in 2023 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT)*, vol. 1, no. 1, pp. 13–17, 2023. DOI: 10 . 1109 /IAICT59002.2023.10205864.

[5] A. Zainudin, R. N. Alief, M. A. P. Putra, R. Akter, D.-S. Kim, and J.-M. Lee, "Blockchain-Assisted Privacy-Preserving Intrusion Detec tion for Secured Metaverse TT -," *Korean, in 한국통신학회 학술대회논문집, 한국통신학회*, vol. 1, no.

1, pp. 322–323, 2023.

[6] E. Tuli, A. Zainudin, M. J. A. Shanto, J. M. Lee, and D.-S. Kim, "Mediapipe-based real-time interactive avatar generation for metaverse," in 한국통신학회 학술대회논문집, 한국통신학회, vol. 1, no. 1, pp. 1, Jun. 2023.

[7] J. Zhang, Z. Ning, and F. Xue, "A two-stage federated optimization algorithm for privacy computing in internet of things," Future Generation Computer Systems, vol. 145, no. 1, pp. 354–366, 2023, ISSN: 0167-739X. DOI: https : / /doi.org/10.1016/j.future .2023.03.042.

[8] J. Park, D. Yoon, S. Yeo, and S. Oh, "Amble: Adjusting mini-batch and local epoch for federated learning with heterogeneous devices," *Journal of Parallel and Distributed Comput ing*, vol. 170, no. 1, pp. 13–23, 2022, ISSN: 0743-7315. DOI: https : / / doi . org / 10 .1016/j.jpdc.2022.07.009.

[9] M. A. P. Putra, A. R. Putri, A. Zainudin, D.-S. Kim, and J.-M. Lee, "Acs: Accuracy-based client selection mechanism for federated industrial iot," *Internet of Things*, vol. 21, no. 1, p. 100 657, 2023, ISSN: 2542-6605. DOI: https : / / doi . org / 10 . 1016 /j . iot . 2022 . 100657

[10] Y. Shi, Y. Zhang, Y. Xiao, and L. Niu, "Optimization strategies for client drift in federated learning: A review," Procedia Computer Science, vol. 214, no. 1, pp. 1168–1173, 2022, 9th International Conference on Information Technology and Quantitative Management, ISSN: 1877-0509. DOI: https://doi.org/10. 1016/j.procs.2022.11.292.

[11] 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*, vol. 1, no. 1, pp. 1 2019.

[12] Y. Liu, Y. Sun, Z. Ding, L. Shen, B. Liu, and D. Tao, "Enhance local consistency in federated learning: A multi-step inertial momentum approach," *arXiv preprint arXiv:2302.05726*, vol. 1, no. 1, pp. 1, 2023.

[13] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization," in Proceedings of the 34th International Conference on Neural Information Processing Systems, ser. NIPS'20, Vancouver, BC, Canada: Curran Associates Inc., 2020, ISBN:9781713829546.

[14] F. Varno, M. Saghayi, L. Rafiee Sevyeri, S. Gupta, S. Matwin, and M. Havaei, "Adabest: Minimizing client drift in federated learning via adaptive bias estimation," in *Computer Vision – ECCV 2022, Cham: Springer Nature Switzerland,* vol. 1, no. 1, 2022, pp. 710–726.

[15] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. 61, pp. 2121–2159, 2011.

[16] J. Wang, Z. Xu, Z. Garrett, Z. Charles, L. Liu, and G. Joshi, "Local adaptivity in federated learning: Convergence and consistency," *arXiv preprint arXiv:2106.02305*, vol. 1, no. 1, pp. 1, 2021.

[17] N. Bouacida, J. Hou, H. Zang, and X. Liu, "Adaptive federated dropout: Improving communication efficiency and generalization for federated learning," in *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications Workshops (INFO-COM WKSHPS)*, vol. 1, no. 1, pp. 1‑6, 2021.

[18] H. Zhang, J. Liu, J. Jia, Y. Zhou, H. Dai, and D. Dou, "Fedduap: Federated learning with dynamic update and adaptive pruning using shared data on the server," *arXiv preprint arXiv:2204.11536*, vol. 1, no. 1, pp. 1, 2022.

[19] X. Yu, L. Li, X. He, S. Chen, and L. Jiang, "Federated Learning Optimization Algorithm for Automatic Weight Optimal," Computational Intelligence and Neuroscience, vol. 2022, M. F. Ijaz, Ed., p. 8 342 638, 2022, ISSN: 1687-5265.

[20] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, T. Parcollet, and N. D. Lane, "Flower: A friendly federated learning research framework," *arXiv preprint arXiv:2007.14390*, vol. 1, no. 1, pp. 1, 2020.

[21] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *International Conference on Machine Learning, PMLR,* vol. 1, no. 1, pp. 5650–5659, 2018.

[22] S. Reddi, Z. Charles, M. Zaheer, et al., "Adaptive federated optimization," *arXiv preprint arXiv:2003.00295*, vol. 1, no. 1, pp. 1, 2020.