Enhanced Federated Learning with Adaptive Block-wise Regularization and Knowledge Distillation

Qingmin Zeng, Jianchun Liu*, Hongli Xu*, Zhiyuan Wang, Yang Xu, Yangming Zhao
School of Computer Science and Technology, University of Science and Technology of China
Suzhou Institute for Advanced Study, University of Science and Technology of China

2023 IEEE/ACM 31st International Symposium on Quality of Service (IWQoS) | 979-8-3503-9973-8/23/$31.00 ©2023 IEEE | DOI: 10.1109/IWQOS57198.2023.10188770

***Abstract*—Federated Learning (FL) has emerged as an efficient distributed model training framework that enables multiple clients cooperatively to train a global model without exposing their local data in edge computing (EC). However, FL usually faces statistical heterogeneity (*e.g.*, non-IID data) and system heterogeneity (*e.g.*, computing and communication capabilities), resulting in poor model training performance. To deal with the above two challenges, we propose an efficient FL framework, named *FedBR*, which integrates the idea of block-wise regularization and knowledge distillation (KD) into the pioneer FL algorithm *FedAvg*, for resource-constrained edge computing. Besides, we design a heuristic algorithm (GMBS) to determine the appropriate number of model blocks for clients according to their varied data distributions, computing, and communication capabilities. Extensive experimental results show that *FedBR* can reduce the time cost by 19.5% and the communication cost by 27% on average compared with the other three baselines when achieving the target testing accuracy under heterogeneous settings.**

***Index Terms*—Federated Learning, Edge Computing, Heterogeneity, Regularization, Knowledge Distillation.**

I. Introduction

As the computing power of edge devices (or clients) becomes more and more powerful, a distributed model training framework, federated learning (FL) [1], is applied to handle more complex tasks for edge computing. However, performing efficient federated learning still faces three main challenges in EC. (1) **Statistical Heterogeneity**. The local data of the clients are usually generated according to the preference and the location of clients [2]. Data samples from different clients are usually not independent and identically distributed (non-IID). This characteristic of non-IID data will seriously hurt the model training performance and reduce the convergence rate [3]. (2) **System Heterogeneity**. The clients participating in the model training have different computing and communication capabilities [4], [5], which are closely related to the time of model updating and transmission, respectively. In FL, the training time of each global round always depends on the slowest client, resulting in a longer completion time. (3) **Communication Limitation**. It takes a lot of bandwidth to deliver models between the server and clients during the training. Therefore, the limited communication bandwidth on the server is also a bottleneck in FL [6].

To mitigate the impact of statistical heterogeneity, regularization [7] has been applied to *FedAvg*, *e.g. FedProx* [8]. *FedMLB* [9] is a new regularization method combined with knowledge distillation (KD) technology [10], which blocks the model hierarchically based on *FedAvg* and constructs multiple auxiliary branches. It puts the output obtained by each auxiliary branch and the output obtained by the local model into the loss function as a regularization item through knowledge distillation to obtain more global information. Although this method can effectively alleviate the non-IID data issue, the introduction of auxiliary branches greatly increases the computing cost [11]. Besides, since all clients use the same number of auxiliary branches and receive the entire global model, *FedMLB* performs poorly under system heterogeneity and limited communication resource according to the results of experiments.

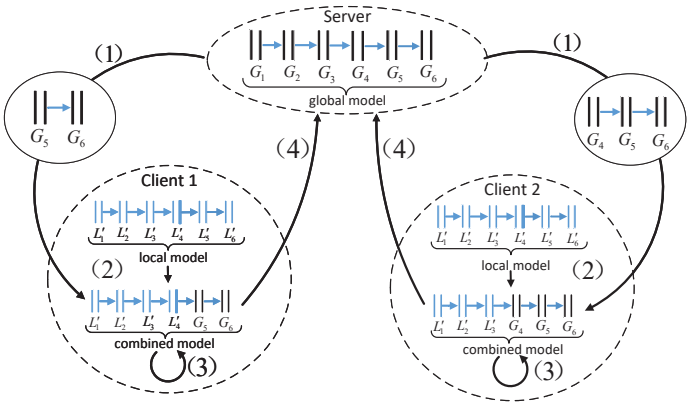
Motivated by *FedMLB*, we design an efficient federated learning framework *FedBR*, which integrates the idea of block-wise regularization and knowledge distillation into *FedAvg*, to solve the above three challenges simultaneously. Specifically, we divide the model into multiple blocks according to the layer order of DNN. To alleviate the impact of non-IID data, we design a new regularization technique, called block-wise regularization, to absorb the knowledge of the global model blocks via KD, where KD is adopted to reduce the discrepancy between the outputs of different paths. Besides, the server only distributes the last several consecutive blocks of the global model in *FedBR*, thus significantly reducing the communication cost compared with *FedAvg*. Although more number of global model blocks allow clients to learn more global information, it will lead to more computing and communication cost. Therefore, *how to dynamically determine the proper number of global model blocks sent by the server for each client* is a key challenge. Based on this, we summarize the main contributions of this paper as follows:

- We propose an efficient FL framework, called *FedBR*, which can reduce the communication cost and alleviate the heterogeneous challenges.
- We propose a heuristic algorithm (termed as GMBS) that adaptively determines the number of global model blocks for clients.
- We evaluate the performance of *FedBR* through extensive experiments. The experimental results show that *FedBR* can shorten the time cost by 19.5% and reduce the bandwidth consumption by 27% on average compared with the other three benchmark methods when achieving the same target testing accuracy under heterogeneous settings.

II. Proposed Framework and Problem Formulation

A. Overview of FedBR

In this section, we propose an efficient federated learning framework with block-wise regularization (*FedBR*) to reduce the communication cost and alleviate the heterogeneous challenges. Specifically, we divide the model into M blocks, where



(1) Model Broadcast (2) Model Combination
(3) Local Updating (4) Model Aggregation

Fig. 1: The workflow of *FedBR*. Multiple consecutive layers in the neural network constitute a model block (*e.g.*, the two layers form a model block in the figure). G_* and L'_* represent the global model blocks (black blocks) and the local model blocks (blue blocks) saved by clients at the previous global round, respectively, where $*$ denote the indexes of the model block.

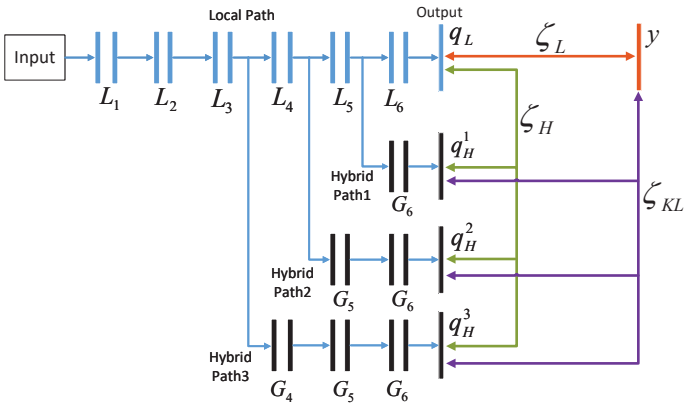


Fig. 2: The block-wise regularization in *FedBR*. Given global round t, the local network of client k consists of α_k^t hybrid paths and one local path. The blue arrow indicates the forward propagation path of the input and q denotes the output of each path. y and ζ represent the truth label of the input and loss term, respectively. Double-headed arrows indicate different loss terms in the loss function.

M is determined by the server according to the layer structure of the model. Concretely, the workflow of *FedBR* is divided into the following four steps as shown in Fig. 1:

(1) **Model Broadcast**: At the beginning of global round t, the server sends different numbers of consecutive global model blocks to the clients. We use x^t and α_k^t to denote the global model and the number of global model blocks which are sent to client k at global round t, respectively. To prevent client k from forgetting the information belonging to the first several blocks of the global model ($e.g., G_1, G_2, G_3$ in Fig. 1), the server will send an entire global model to all clients every $\tau + 1$ rounds, where $\tau \geq 1$.

(2) **Model Combination**: If client k receives the last α_k^t global model blocks at global round t, it will combine the first $M - \alpha_k^t$ blocks of local model $x_k^t(te)$ with the last α_k^t blocks of the global model to form the combined model. We use $x_k^t(i)$ to represent the local model of client k after i local iterations, where i is equal to the total number of local iterations from the beginning of model training. If client k receives entire global model at global round t, it will directly take the received model as the combined model.

(3) **Local Updating**: At global round t, client k uses the combined model as the new local model $x_k^t(te)$. Then, client k performs e local iterations based on its own local dataset D_k using the block-wise regularization in Fig. 2 to obtain updated local model $x_k^{t+1}((t+1)e)$. When e local iterations are completed, client k returns the entire local model $x_k^{t+1}((t+1)e)$ to the server.

(4) **Model Aggregation**: After the server collects all local models from K clients, it will aggregate all models to derive an up-to-date global model x^{t+1} according to the following equation:

$$x^{t+1} = \frac{1}{K} \sum_{k=1}^{K} x_k^{t+1}((t+1)e). \tag{1}$$

Then, *FedBR* continues to perform the Model Broadcast at the next global round $t + 1$ until the global model reaches convergence or network resources are exhausted.

B. Problem Definition

Assume that there are N clients in the network and only K clients participating in the model training ($K \leq N$). Without loss of generality, computing and communication resources are taken into consideration in this work. The total budgets of computing and communication resources are denoted as B_c and B_b in the network, respectively. Besides, the computing cost of one batch and the consumption of transferring the entire model are denoted as c and b, respectively. Let c_k^t represent the computing cost of client k at each local iteration of global round t. It can be expressed as:

$$c_k^t = (1 + \frac{\alpha_k^t(\alpha_k^t + 1)}{2M}) \cdot \frac{n_k}{B} \cdot c, \tag{2}$$

where n_k is the number of local samples in client k, B is the size of batch, and the coefficient (*i.e.*, $1 + \alpha_k^t(\alpha_k^t + 1)/2M$) is related to the multiplexing of intermediate results between the local path and hybrid paths. We use b_k^t to denote the average communication cost of client k at global round t. The communication cost of client k can be expressed as $(1 + \alpha_k^t/M) \cdot b$. Combining the above two cases, the average communication cost of client k in each global ground can be expressed as:

$$b_k^t = (1 + \frac{1}{\tau} + (1 - \frac{1}{\tau})\frac{\alpha_k^t}{M}) \cdot b, \tag{3}$$

We aim to minimize the completion time while finding the proper α_k^t for each client k at round t in *FedBR*. Accordingly, we formulate the FLMBT problem as follows:

$$\min \sum\nolimits_{t=1}^{T} H_t \tag{4}$$

$$s.t. \begin{cases} f(x^T) \leq \mathcal{F}, & \\ \sum_{t=1}^{T} \sum_{k=1}^{K} e \cdot c_k^t \leq B_c, & \forall k, t \\ \sum_{t=1}^{T} \sum_{k=1}^{K} b_k^t \leq B_b, & \forall k, t \\ \alpha_k^t \in \{1, \cdots, M-1\}, & \forall k, t, \end{cases}$$

where H_t represents the completion time of global round t, *i.e.*, the time that K clients complete their local training after the last global round. The first inequality expresses the convergence requirement, where $\mathcal{F}$ is the convergence threshold of the loss value after T global rounds. The second set of inequalities and the third set of inequalities indicate that the computing and communication cost during T global rounds should not exceed their total budget. The fourth group of formulas denote that the number of blocks should be an integer and not exceed the number of total model blocks.

In fact, it is difficult to directly solve the FLMBT problem in Eq. (4). Since the decision variable of FLMBT is an integer, this is a typical integer programming problem. In general, finding the optimal solution for an integer programming problem is NP-hard [12]. Consequently, solving problem in Eq. (4) at each global round will be time-consuming. Meanwhile, the time-varying network conditions also aggravate the difficulty of the problem.

III. Algorithm Design

In this section, we propose an efficient algorithm to solve the FLMBT problem in Eq. (4). The key challenge to solve problem in Eq. (4) is to find the appropriate α_k^t for each client k at the beginning of any global round t while minimizing the completion time. The number of blocks α_k^t should depend on the data distribution as well as the communication and computing time of client k. The difference between the local model and the global model is proportional to the bias between the local data distribution and the overall data distribution [13]. We use d_k to denote the deviation between the data distribution of client k and the overall data distribution. Let $t_{k,b}$ and $t_{k,c}$ represent the communication time and the computing time of client k, respectively. To facilitate comparison, we employ normalization for the communication and computing time of client k. Apparently, α_k^t should be positively correlated with d_k and inversely proportional to $t'_{k,b}$ and $t'_{k,c}$ for client k. As a result, we design a feedback variable $r_{k,m}^t$:

$$r_{k,m}^t \triangleq \frac{d_k \cdot \Delta\zeta_L}{e^{t'_{k,c}+t'_{k,b}}}, \tag{5}$$

where the exponential function is applied to enhance the effect of time cost on the feedback variable, ζ_L is the local loss function. To determine the proper α_k^t for client k, we design a greedy-based model blocks selection (GMBS) algorithm. The server holds one vector p_k for each client k to remember the feedback information of the previous global round, where $p_{k,m}$ corresponds to the feedback information of sending m global model blocks to client k. We need to remember previous feedback information to decrease the interference of error feedback information caused by a certain round. Therefore, $p_{k,m}$ is updated by the following equation:

$$p_{k,m} = \lambda r_{k,m}^t + (1-\lambda)p_{k,m}, \tag{6}$$

where λ ($0 \leq \lambda \leq 1$) is a hyperparameter that reflects the weight of real-time feedback information. Since the number of global model blocks is randomly selected at the beginning of training and the states of clients are variable, we cannot always adopt m corresponding to the highest value of feedback information as α_k for the decision of client k. Therefore, we add a penalty item for feedback information $p_{k,m}$. Finally, we use the following decision variable $V_{k,m}$ to select the number of global model blocks:

$$V_{k,m} = p_{k,m} + \frac{\sqrt{\ln(t+1)}}{n_{k,m}+1}, \tag{7}$$

where $n_{k,m}$ represents the frequency that the server selects m global model blocks to send to client k. We choose m corresponding to the largest decision variable as the number of global model blocks received by client k at round t ($i.e., m = \arg\max_m V_{k,m}$).

IV. Performance Evaluation

A. Experimental Setup

Datasets and Model: We use two real-world classical datasets, *i.e.*, CIFAR10 and CIFAR100 [14] to evaluate the performance of *FedBR* and the baselines. Besides, we train a ResNet18 [15] model, which is often adopted for image recognition tasks.

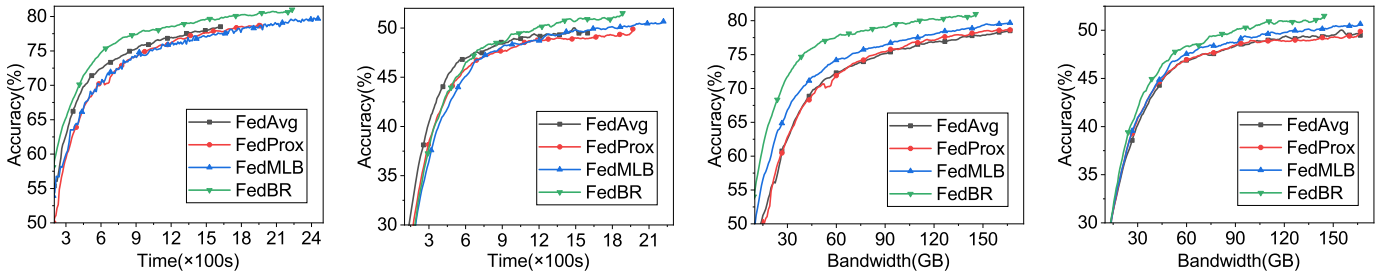
Data Division: We use ϵ to represent the non-IID level, which ranges from $\{0, 1, \cdots, 9\}$. When $\epsilon = 0$, it represents the data distribution is IID. For CIFAR10, when $\epsilon \in \{1, \cdots, 9\}$, it denotes that the $\epsilon \times 10\%$ local data of the client belongs to the same class and the rest of the local data is evenly divided into the remaining nine classes. For CIFAR100, when $\epsilon \in \{1, \cdots, 9\}$, it denotes that the local dataset of each client lacks $\epsilon \times 10$ kinds of images and the local data is evenly distributed among the remaining $100 - \epsilon \times 10$ classes.

Baselines and Metrics: We compare the performance of *FedBR* together with the three baselines, *i.e.*, *FedAvg* [1], *FedProx* [8], and *FedMLB* [9]. In this paper, we use the following three metrics to evaluate the performance of our proposed framework: (1) Testing accuracy. We will compute the testing accuracy of the global model on the testing set. (2) Time cost. We will record the completion time when the model training achieves the given target accuracy. (3) Communication cost. We will record the total bandwidth consumption of all clients for uploading and downloading models.

B. Experimental Results

Resource Consumption: We test the resource consumption of *FedBR* and the baselines to achieve a target testing accuracy, as shown in Fig. 4. For time cost, *FedBR* consumes the least time among all methods to train the model either on CIFAR10 or CIFAR100 according to the left plot of Fig. 4. For example, when CIFAR100 is trained on ResNet18 under $\epsilon = 7$, *FedBR* takes 954 seconds, which is 14.6% less than *FedAvg* (1,104 seconds), 20% less than *FedProx* (1,189 seconds), and 24% less than *FedMLB* (1,245 seconds). For communication cost, as shown in the right plot of Fig. 4, *FedBR* consumes the least communication bandwidth to train the model either on CIFAR10 or CIFAR100 when the global model achieves a target accuracy. For instance, when CIFAR100 is trained on ResNet18 under $\epsilon = 7$, we obtain that the communication bandwidth consumed by *FedBR* is 72.8GB, which saves 31%, 27.5%, and 22.3% of bandwidth when *FedAvg* (105.4GB), *FedProx* (100.4GB), and *FedMLB* (93.7GB) achieve the same testing accuracy, respectively.

Impact of Resource Constraints: We conduct experiments to test the performance of *FedBR* and three baselines with the



(a) Testing accuracy with completion time constraint on CIFAR10 (b) Testing accuracy with completion time constraint on CIFAR100 (c) Testing accuracy with bandwidth constraint on CIFAR10 (d) Testing accuracy with bandwidth constraint on CIFAR100

Fig. 3: Testing accuracy with completion time and bandwidth constraints under the non-IID level $\epsilon = 7$

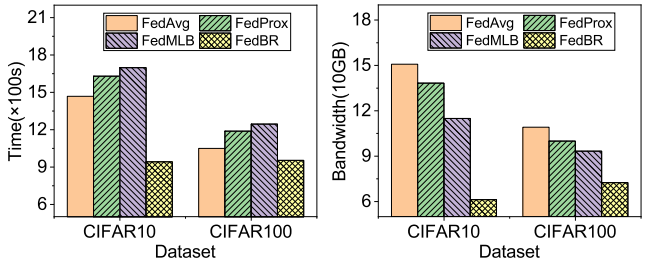


Fig. 4: Time and bandwidth consumption after the model achieves the target accuracy on CIFAR10 (0.8) and CIFAR100 (0.5).

resource constraints. For the completion time constraint, Fig. 3(a) and Fig. 3(b) show that the testing accuracy will increase with the increasing completion time budget on both CRFAR10 and CIFAR100. Note that *FedBR* can achieve the best training performance in four methods with the same completion time budget. For example, the testing accuracy of *FedBR* is about 77.4% on CIFAR10 when given the completion time is 900s, while that of *FedAvg*, *FedProx* and *FedMLB* is about 75.3%, 74.1% and 74.0%, respectively. For bandwidth constraint, more communication bandwidth budget will significantly improve the performance of all methods according to Fig. 3(c) and Fig. 3(d). However, *FedBR* obtains better performance than *FedAvg*, *FedProx*, and *FedMLB* while consuming the same network bandwidth. For instance, given the total network bandwidth budget of 50GB, the testing accuracy of *FedBR* is about 76.8% on CIFAR10, while that of *FedAvg*, *FedProx*, and *FedMLB* is about 70.8%, 70.3%, and 72.3%, respectively.

V. CONCLUSIONS

In this work, we have proposed the *FedBR* framework to deal with statistical heterogeneity and system heterogeneity for resource-constraint edge computing. We have designed a heuristic algorithm (GMBS) to adaptively determines the number of global model blocks for each client. We have built a simulation environment and evaluated the performance of *FedBR*. The results demonstrate the effectiveness of *FedBR* in improving the model accuracy and reducing resource consumption.

ACKNOWLEDGEMENT

This article is supported in part by the National Key Research and Development Program of China (Grant No. 2021YFB3301500); in part by the National Science Foundation of China (NSFC) under Grants 62102391, 62132019 and 61936015; in part by the Jiangsu Province Science Foundation for Youths (Grant No. BK20210122); in part by Xiaomi Young Scholar.

REFERENCES

[1] 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.

[2] B. Varghese, N. Wang, S. Barbhuiya, P. Kilpatrick, and D. S. Nikolopoulos, "Challenges and opportunities in edge computing," in *2016 IEEE International Conference on Smart Cloud (SmartCloud)*. IEEE, 2016, pp. 20–26.

[3] 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.

[4] B. Luo, W. Xiao, S. Wang, J. Huang, and L. Tassiulas, "Tackling system and statistical heterogeneity for federated learning with adaptive client sampling," in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*. IEEE, 2022, pp. 1739–1748.

[5] Z. Wang, H. Xu, J. Liu, Y. Xu, H. Huang, and Y. Zhao, "Accelerating federated learning with cluster construction and hierarchical aggregation," *IEEE Transactions on Mobile Computing*, 2022.

[6] D. Verma, S. Julier, and G. Cirincione, "Federated ai for building ai solutions across multiple agencies," *arXiv preprint arXiv:1809.10036*, 2018.

[7] F. Girosi, M. Jones, and T. Poggio, "Regularization theory and neural networks architectures," *Neural computation*, vol. 7, no. 2, pp. 219–269, 1995.

[8] 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.

[9] J. Kim, G. Kim, and B. Han, "Multi-level branched regularization for federated learning," in *International Conference on Machine Learning*. PMLR, 2022, pp. 11 058–11 073.

[10] G. Hinton, O. Vinyals, J. Dean *et al.*, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, vol. 2, no. 7, 2015.

[11] A. Mora, I. Tenison, P. Bellavista, and I. Rish, "Knowledge distillation for federated learning: a practical guide," *arXiv preprint arXiv:2211.04742*, 2022.

[12] J. Liu, H. Xu, G. Zhao, C. Qian, X. Fan, and L. Huang, "Incremental server deployment for scalable nfv-enabled networks," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 2020, pp. 2361–2370.

[13] D. A. E. Acar, Y. Zhao, R. Zhu, R. Matas, M. Mattina, P. Whatmough, and V. Saligrama, "Debiasing model updates for improving personalized federated training," in *International Conference on Machine Learning*. PMLR, 2021, pp. 21–31.

[14] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.