模式分析与机器智能
工业和信息化部重点实验室
MIIT Key Laboratory of
Pattern Analysis & Machine Intelligence

ParNeC 模式识别与神经计算研究组
PAttern Recognition and NEural Computing

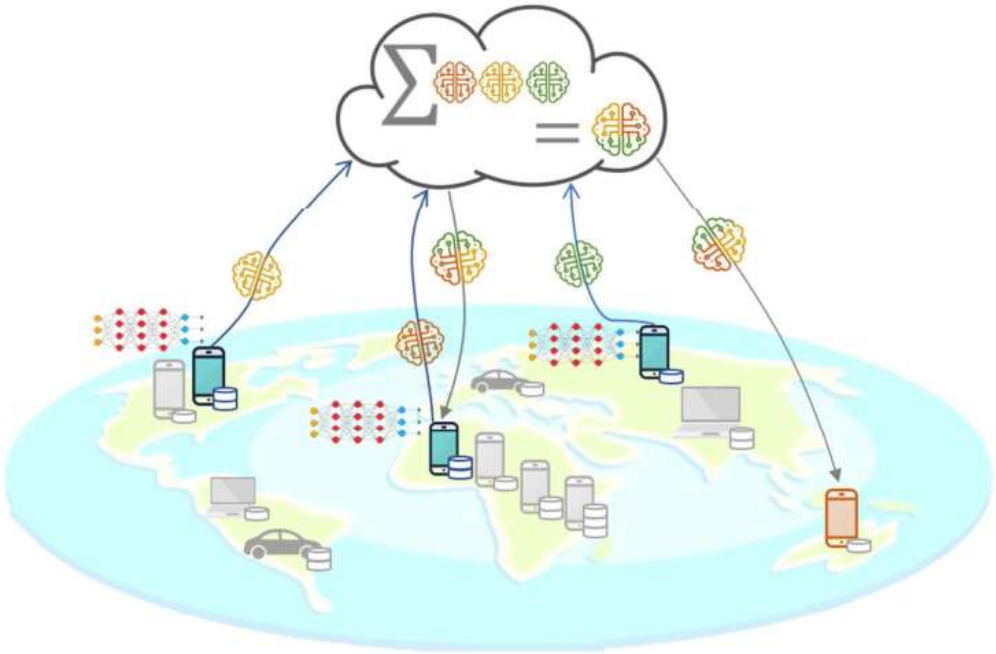
FedBR: Improving Federated Learning on Heterogeneous Data via Local Learning Bias Reduction

Yongxin Guo [1 2] Xiaoying Tang [1 2 3] Tao Lin [4 5]

ICML 2023

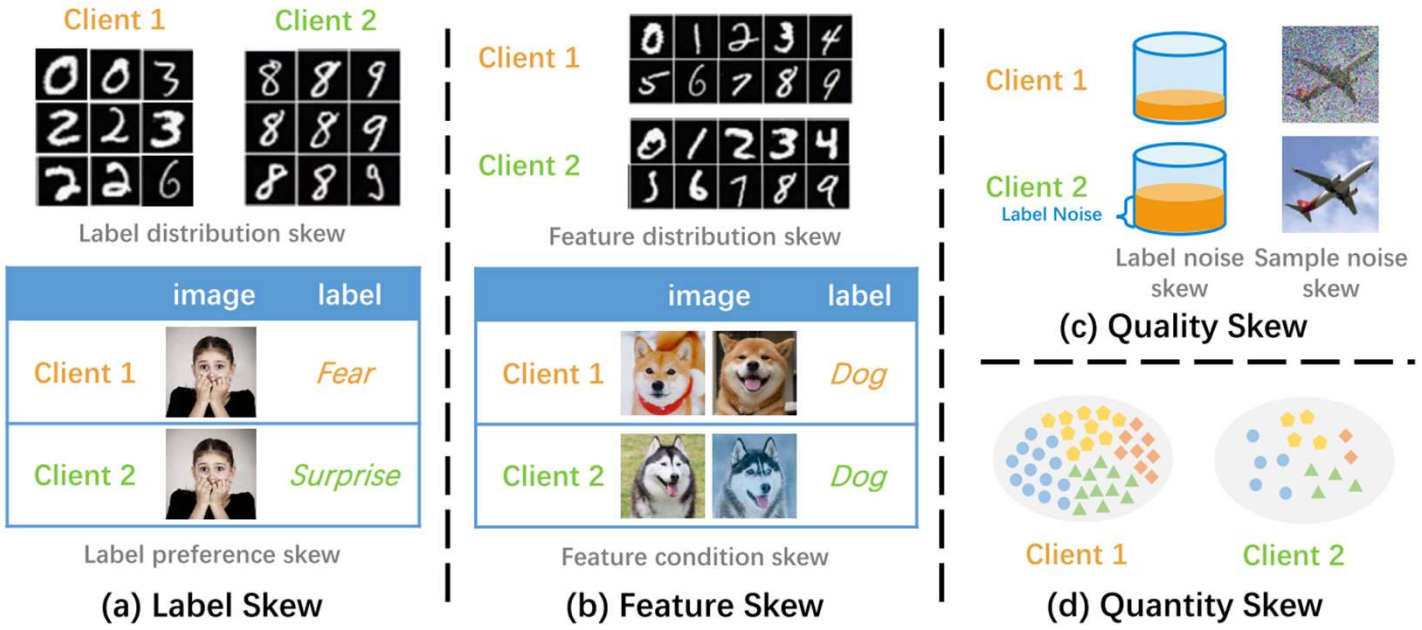
Federated Learning (FL)

FL can improve model's performance while protecting users' privacy.



Statistical Heterogeneity

Statistical heterogeneity refers to the case where the data distribution across clients in federated learning is inconsistent and does not obey the same sampling, i.e., Non-IID.



Related work

Local training

FedAvg
(McMahan et al.,2016)
classic FL algorithm requires many communication rounds to train an effective global model.

FedProx
(Li et al.,2020)
adjusts the local training procedure to pull back local models from global model.

SCAFFOLD
(Karimireddy et al.,2020)
uses control variates (variance reduction) to correct for the client-drift in its local updates.

MOON
(Li et al., 2021)
has proposed to employ contrastive loss to reduce the distance between global and local features.

Related work

Data augmentation

FedMix
(Yoon et al.,2021b)

creates the privacy-protected augmentation data by averaging local batches and then applying Mixup (Zhang et al., 2018) (linear interpolation between actual data instances) in local iterations.

VHL
(Tang et al., 2022)

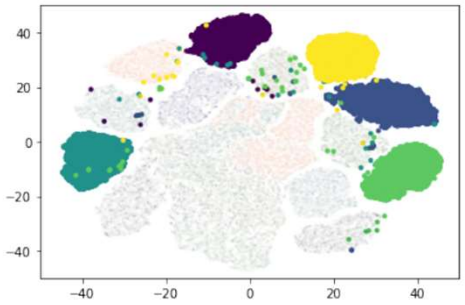
relies on the created virtual data with labels and forces the local features to be close to the features of same-class virtual data.

Bias caused by local updates

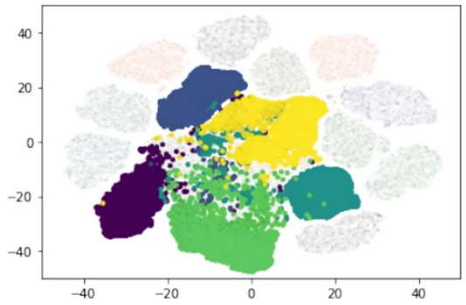
For FedAvg, the local models after local epochs could be biased, in detail,

- **Biased local feature**: For local feature extractor $F_i(\cdot)$, and centralized trained global feature extractor $F_g(\cdot)$, we have: 1) Given the data input X, $F_i(X)$ could deviate largely from $F_g(X)$. 2) Given the input from different data distributions X_1 and X_2, $F_i(X_1)$ could be very similar or almost identical to $F_i(X_2)$.

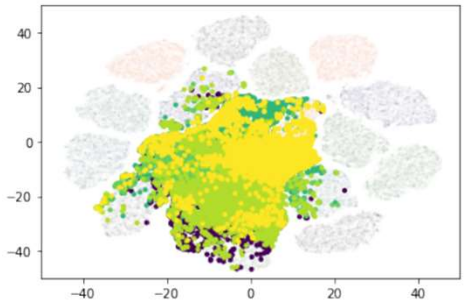
- **Biased local classifier**: After a sufficient number of iterations, local models classify all samples into only the classes that appeared in the local datasets.



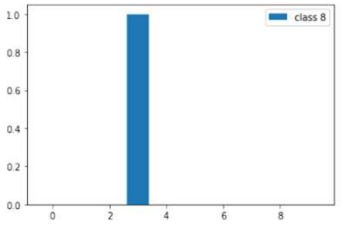
(a) Global feature of X_1, $F_g(X_1)$



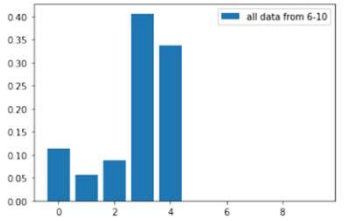
(b) Local feature of X_1, $F_1(X_1)$



(c) Local feature of X_2, $F_1(X_2)$



(a) Sample from class 8



(b) All samples in X_2

Motivation

Defects in previous works for local learning bias

- **FedProx** defines local drifts as the differences in model weights, while **SCAFFOLD** considers gradient differences as client drifts. These methods, though have been effective on traditional optimization tasks, may only have marginal improvements on deep models.
- **MOON** minimizes the distance between global and local features, but its performance is limited because they use only the projection layer as part of the feature extractor, and the contrastive loss diminished without our designed max step.
- **VHL** defines local learning bias as the shift in features between samples of the same classes; however, this approach requires prior knowledge of local label information and results in a much larger virtual dataset, especially when increasing the number of classes.

Overview of the FedBR

first projecting features onto spaces that can distinguish global and local feature best:

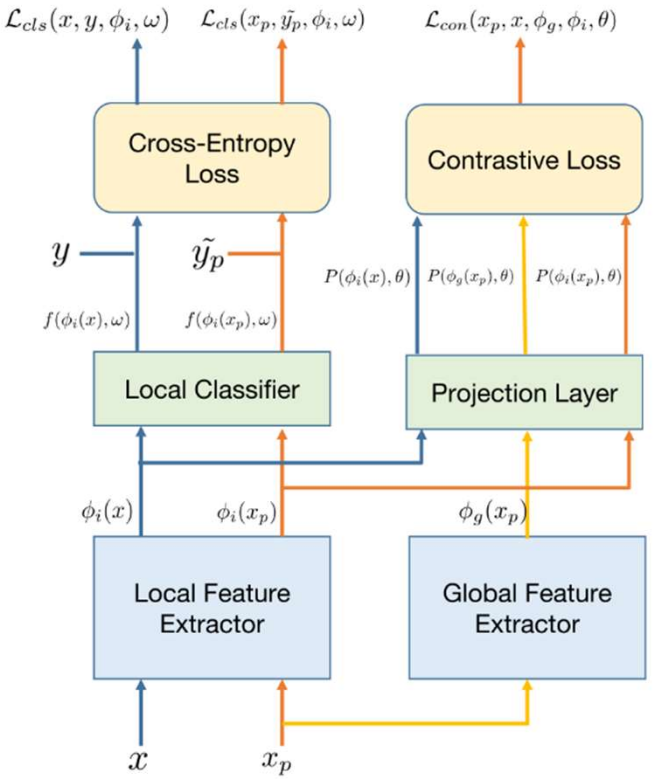
$$\textbf{Max Step: } \max_{\boldsymbol{\theta}} \mathcal{L}_{adv}(D_p, D_i)$$

$$:= \mathbb{E}_{\mathbf{x}_p \sim D_p, \mathbf{x} \sim D_i} \left[\mathcal{L}_{con}(\mathbf{x}_p, \mathbf{x}, \boldsymbol{\phi}_g, \boldsymbol{\phi}_i, \boldsymbol{\theta}) \right] . \quad (2)$$

then 1) minimizing the distance between the global and local features of pseudo-data and maximizing distance between local features of pseudo-data and local data; 2) minimize classification loss of both local data and pseudo-data:

$$\textbf{Min Step: } \min_{\boldsymbol{\phi}_i, \boldsymbol{\omega}} \mathcal{L}_{gen}(D_p, D_i)$$

$$:= \mathbb{E}_{(\mathbf{x},\mathbf{y}) \sim D_i} \left[\mathcal{L}_{cls}(\mathbf{x}, \mathbf{y}, \boldsymbol{\phi}_i, \boldsymbol{\omega}) \right] + \lambda \mathbb{E}_{\mathbf{x}_p \sim D_p} \left[\mathcal{L}_{cls}(\mathbf{x}_p, \tilde{\mathbf{y}}_p, \boldsymbol{\phi}_i, \boldsymbol{\omega}) \right] + \mu \mathbb{E}_{\mathbf{x}_p \sim D_p, \mathbf{x} \sim D_i} \left[\mathcal{L}_{con}(\mathbf{x}_p, \mathbf{x}, \boldsymbol{\phi}_g, \boldsymbol{\phi}_i, \boldsymbol{\theta}) \right] . \quad (3)$$



Overview of the FedBR

Algorithm 1 Algorithm Framework of FedBR

Require: Local datasets $D_1, \ldots, D_N$, pseudo dataset D_p where $|D_p| = B$, and B is the batch size, number of local iterations K, number of communication rounds T, number of clients chosen in each round M, weights used in designed loss λ, μ, local learning rate η.

Ensure: Trained model $\boldsymbol{\omega}_T, \boldsymbol{\theta}_T, \boldsymbol{\phi}_T$.

Initialize $\boldsymbol{\omega}_0, \boldsymbol{\theta}_0, \boldsymbol{\phi}_0$.
for $t = 0, \ldots, T-1$ **do**
  Send $\boldsymbol{\omega}_t, \boldsymbol{\theta}_t, \boldsymbol{\phi}_t, D_p$ (optional) to all clients.
  for chosen client $i = 1, \ldots, M$ **do**
    $\boldsymbol{\omega}_i^0 = \boldsymbol{\omega}_t, \boldsymbol{\theta}_i^0 = \boldsymbol{\theta}_t, \boldsymbol{\phi}_i^0 = \boldsymbol{\phi}_t, \boldsymbol{\phi}_g = \boldsymbol{\phi}_t$
    for $k = 1, \ldots, K$ **do**
      # Max Step
      $\boldsymbol{\theta}_i^k = \boldsymbol{\theta}_i^{k-1} + \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}_{adv}$.
      # Min Step
      $\boldsymbol{\omega}_i^k = \boldsymbol{\omega}_i^{k-1} - \eta \nabla_{\boldsymbol{\omega}} \mathcal{L}_k$.
      $\boldsymbol{\phi}_i^k = \boldsymbol{\phi}_i^{k-1} - \eta \nabla_{\boldsymbol{\phi}} \mathcal{L}_{gen}$.
    Send $\boldsymbol{\omega}_i^K, \boldsymbol{\theta}_i^K, \boldsymbol{\phi}_i^K$ to server.
  $\boldsymbol{\omega}_{t+1} = \frac{1}{M} \sum_{i=1}^{M} \boldsymbol{\omega}_i^K$.
  $\boldsymbol{\theta}_{t+1} = \frac{1}{M} \sum_{i=1}^{M} \boldsymbol{\theta}_i^K$.
  $\boldsymbol{\phi}_{t+1} = \frac{1}{M} \sum_{i=1}^{M} \boldsymbol{\phi}_i^K$.

Construction of the pseudo-data

Random Sample Mean (RSM)

One RSM sample of the pseudo-data is estimated through a weighted combination of a random subset of local samples, and the pseudo-label is set to $\widetilde{y_p} = \frac{1}{C} \cdot 1$

Mixture of local samples and the sample mean of a proxy dataset (Mixture)

This strategy relies on applying the procedure of RSM to irrelevant and globally shared proxy data. To guard the distribution distance between the pseudo-data and local data, one sample of the pseudo-data at each client is constructed by

$$\tilde{\mathbf{x}}_p = \frac{1}{K+1}(\mathbf{x}_p + \sum_{k=1}^{K} \mathbf{x}_k), \tilde{\mathbf{y}}_p = \frac{1}{K+1}(\frac{1}{C} \cdot \mathbf{1} + \sum_{k=1}^{K} \mathbf{y}_k), \quad (4)$$

Reducing bias components

Component 1: Reducing Bias in Local Classifiers

implicitly mimic the global data distribution by using the pseudo-data constructed in to regularize the outputs and thus debias the classifier

$$\lambda \mathbb{E}_{\mathbf{x}_p \sim D_i} \left[\mathcal{L}_{cls}(\mathbf{x}_p, \tilde{\mathbf{y}}_p, \boldsymbol{\phi}_i, \boldsymbol{\omega}) \right] .$$

Component 2: Reducing Bias in Local Features

1. construct a projection layer as the critical step to distinguish features extracted by the global and local feature extractor: can be achieved by maximizing the distance between global and local features of pseudo-data and simultaneously minimizing the distance between local features of pseudo-data and local data.
2. minimize the local feature biases under the trained projection space, to enforce the learned local features of pseudo-data to be closer to the global features of pseudo-data but far away from the local features of real local data.

$$f_1 = \exp\left(\frac{\text{sim}\left(P_{\boldsymbol{\theta}}(\boldsymbol{\phi}_i(\mathbf{x}_p)), P_{\boldsymbol{\theta}}(\boldsymbol{\phi}_g(\mathbf{x}_p))\right)}{\tau_1} \right) , \quad (5)$$

$$f_2 = \exp\left(\frac{\text{sim}\left(P_{\boldsymbol{\theta}}(\boldsymbol{\phi}_i(\mathbf{x}_p)), P_{\boldsymbol{\theta}}(\boldsymbol{\phi}_i(\mathbf{x}))\right)}{\tau_2} \right) , \quad (6)$$

$$\mathcal{L}_{con}(\mathbf{x}_p, \mathbf{x}, \boldsymbol{\phi}_g, \boldsymbol{\phi}_i, \boldsymbol{\theta}) = -\log\left(\frac{f_1}{f_1 + f_2} \right) , \quad (7)$$

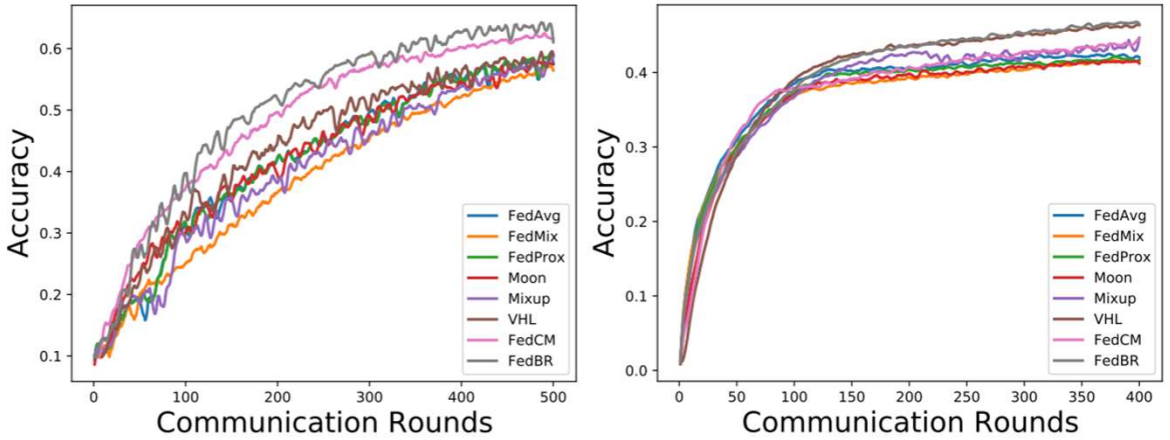
Experiment

The superior performance of FedBR over existing FL and DG algorithms

Table 1: **Performance of algorithms.** We split RotatedMNIST, CIFAR10, and CIFAR100 to 10 clients with $\alpha = 0.1$, and ran 1000 communication rounds on RotatedMNIST and CIFAR10 for each algorithm, 800 communication rounds CIFAR100. We report the mean of maximum (over rounds) 5 test accuracies and the number of communication rounds to reach the threshold accuracy.

Algorithm	RotatedMNIST (CNN)		CIFAR10 (VGG11)		CIFAR100 (CCT)	
	Acc (%)	Rounds for 80%	Acc (%)	Rounds for 55%	Acc (%)	Rounds for 43%
Local	14.67	-	10.00	-	1.31	-
FedAvg	82.47	828 (1.0X)	58.99	736 (1.0X)	44.00	550 (1.0X)
FedProx	82.32	824 (1.0X)	59.14	738 (1.0X)	43.09	756 (0.7X)
Moon	82.68	864 (0.9X)	58.23	820 (0.9X)	42.87	766 (0.7X)
DANN	84.83	743 (1.1X)	58.29	782 (0.9X)	41.83	-
GroupDRO	80.23	910 (0.9X)	56.57	835 (0.9X)	44.34	444 (1.2X)
FedBR (Ours)	**86.58**	**628 (1.3X)**	64.65	496 (1.5X)	45.14	352 (1.5X)
FedAvg + Mixup	82.56	840 (1.0X)	58.57	826 (0.9X)	46.37	358 (1.6X)
FedMix	81.33	902 (0.9X)	57.37	872 (0.8X)	42.69	-
FedBR + Mixup (Ours)	83.42	736 (1.1X)	**65.32**	**392 (1.9X)**	**47.75**	294 (1.9X)

The superior performance of FedBR over existing FL and DG algorithms



(a) CIFAR10 Convergence (b) CIFAR100 Convergence

Figure 5: **Convergence curve of algorithms on different datasets.** We split RotatedMNIST, CIFAR10, and CIFAR100 datasets to 10 clients, and report the mean accuracy on all local test datasets for each communications rounds. More Details refer to Figure 9 of Appendix C.

Experiment

Performance on other aspects

Table 5: **Performance of algorithms with 100 clients.** We split CIFAR10 dataset into 100 clients with $\alpha = 0.1$. We run 1000 communication rounds for each algorithm on the VGG11 model and report the mean of the maximal 5 accuracies (over rounds) during training on test datasets.

Methods	FedAvg	FedDecorr	FedMix	FedProx	Mixup	VHL	FedBR
Acc	38.20	35.53	34.71	37.90	36.63	40.93	41.59

Table 6: **Performance of local model on balanced global test datasets.** We split CIFAR10 to 10 clients with $\alpha = 0.1$, and report the test accuracies achieved by the local models/aggregated models at the end of each communication round. For FedBR, pseudo-data only transfer once (32 pseudo-data).

Algorithm	FedAvg	FedDecorr	VHL	FedBR
Local Model Performance	21.01	21.18	32.81	21.83
Aggregated Model Performance	46.37	47.10	46.80	47.67

Table 7: **Parameter transmitted and mean simulation time in each round.** We split CIFAR10 and CIFAR100 to 10 clients with $\alpha = 0.1$. For FedBR, pseudo-data only transfer once (32 pseudo-data). The simulation time only includes the computation time per step, and do not includes the communication time. CIFAR100 experiments use Mixup as backbone.

CIFAR10 (VGG11)	FedAvg	Moon	VHL	FedCM	FedBR
Parameters (Millions)	9.2	9.7	9.2	18.4	9.7
Mean simulation time (s)	0.29	0.69	0.43	0.36	0.60
CIFAR100 (CCT)	FedAvg	Moon	VHL	FedCM	FedBR
Parameters (Millions)	22.4	22.6	22.4	44.8	22.6
Mean simulation time (s)	0.67	1.97	1.44	0.85	1.19

Experiment

Comparison with VHL

Table 2: **Comparison with VHL.** We split CIFAR10 and CIFAR100 to 10 clients with $\alpha = 0.1$, and report the mean of maximum (over rounds) 5 test accuracies and the number of communication rounds to reach the threshold accuracy. We set different numbers of virtual data to check the performance of VHL, and pseudo-data only transfer once in FedBR (32 pseudo-data). For CIFAR100, we choose Mixup as the backbone.

Algorithm	CIFAR10 (VGG11)		CIFAR100 (CCT)	
	Acc (%)	Rounds for 60%	Acc (%)	Rounds for 46%
VHL (2000 virtual data)	61.23	886 (1.0X)	46.80	630 (1.0X)
VHL (20000 virtual data)	59.65	998 (0.9X)	46.51	714 (0.9X)
FedBR (32 pseudo-data)	**64.61**	**530 (1.8X)**	**47.67**	**554 (1.1X)**

1. FedBR always outperforms VHL.
2. FedBR overcomes several shortcomings of VHL, e.g. the need for labeled virtual data and the large size of the virtual dataset.

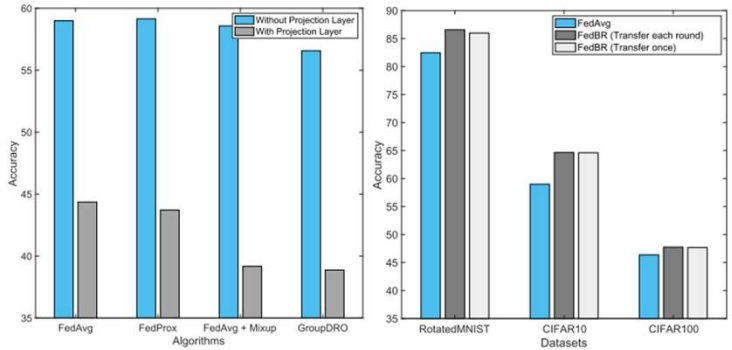
Ablation Study

Table 4: **Ablation studies of FedBR** on the effects of two components. We show the performance of two components and remove the max step (Line 8 in Algorithm 1) of component 2. We split RotatedMNIST, CIFAR10, and CIFAR100 to 10 clients with $\alpha = 0.1$. We run 1000 communication rounds on RotatedMNIST and CIFAR10 for each algorithm and 800 communication rounds on CIFAR100. We report the mean of maximum (over rounds) 5 test accuracies and the number of communication rounds to reach the target accuracy.

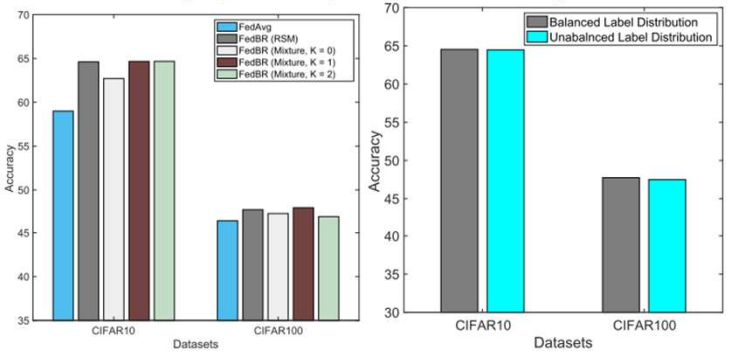
Algorithm	RotatedMNIST (CNN)		CIFAR10 (VGG11)		CIFAR100 (CCT)	
	Acc (%)	Rounds for 80%	Acc (%)	Rounds for 55%	Acc (%)	Rounds for 43%
FedAvg	82.47	828 (1.0X)	58.99	736 (1.0X)	46.37	358 (1.0X)
Component 1	84.40	770 (1.1X)	64.32	**442 (1.7X)**	47.22	330 (1.1X)
+ min step	80.81	922 (0.9X)	62.98	562 (1.3X)	46.54	358 (1.0X)
Component 2	86.25	648 (1.3X)	63.44	483 (1.5X)	**47.78**	308 (1.2X)
+ w/o max step	81.24	926 (0.9X)	58.84	584 (1.3X)	43.50	512 (0.7X)
FedBR	**86.58**	**628 (1.3X)**	**64.65**	496 (1.5X)	**47.75**	**294 (1.2X)**

Experiment

Ablation Study



(a) w/ and w/o projection layer. (b) comm. of pseudo-data.



(c) choices of pseudo-data. (d) label distribution of RSM.

Figure 6: **Ablation studies of FedBR**, regarding the impact of the projection layer, the communication strategy of pseudo-data, and the choices of pseudo-data. In Figure 6(a), we show the performance of algorithms with/without the additional projection layer on the CIFAR10 dataset with the VGG11 model. In Figure 6(b), we show the performance of FedBR on RotatedMNIST, CIFAR10, and CIFAR100 datasets when only transferring pseudo-data once (at the beginning of training) or generating new pseudo-data each round. In Figure 6(c), we show the performance of FedBR using different types of pseudo-data. In Figure 6(d), we show the performance of FedBR when constructing RSM using data with balanced and unbalanced label distribution. Pseudo-data *transfer once* at the beginning of the training in Figure 6(c), and Figure 6(d).

Thanks