GPFL: Simultaneously Learning Global and Personalized Feature Information for Personalized Federated Learning

Jianqing Zhang[1], Yang Hua[2], Hao Wang[3], Tao Song[1]
Zhengui Xue[1], Ruhui Ma[1]*, Jian Cao[1], Haibing Guan[1]
[1]Shanghai Jiao Tong University [2]Queen's University Belfast [3]Louisiana State University
{tsingz, songt333, zhenguixue, ruhuima, cao-jian, hbguan}@sjtu.edu.cn
Y.Hua@qub.ac.uk, haowang@lsu.edu

A. Convergence Analysis

Here, we empirically show the convergence of GPFL. Recall that our learning objective is

$$\{W_1, \ldots, W_N\} = \arg\min \ \mathcal{G}(\mathcal{F}_1, \ldots, \mathcal{F}_N), \qquad (1)$$

where $\mathcal{G}(\mathcal{F}_1, \ldots, \mathcal{F}_N) = \sum_{i \in [N]} n_i \mathcal{F}_i$. In Figure 1, we visualize a more detailed loss curve of GPFL on Amazon Review dataset using the 3-layer MLP in the *feature shift* setting. As the FL process continues, the loss values of $\mathcal{G}$ before and after the local model training become the same, and the loss values do not change for more than 100 iterations, revealing the convergence of GPFL.

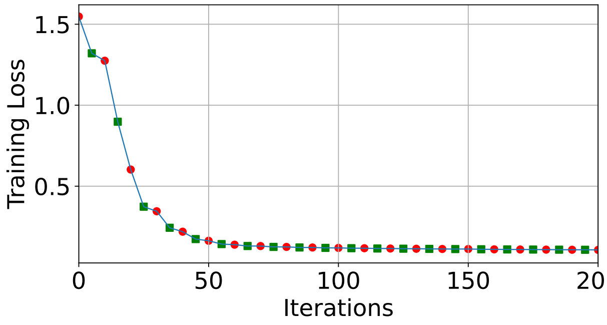


Figure 1. The training loss curve of GPFL on Amazon Review dataset using the 3-layer MLP in the *feature shift* setting. The red circles and green cubes represent the loss value of $\mathcal{G}$ before local learning and after local learning, respectively. Best viewed in color.

B. Results on FMNIST

The results in Table 1 show the superiority of GPFL on FMNIST when compared to other baselines.

Table 1. The test accuracy (%) on FMNIST in *label skew* settings.

Settings	**Pathological setting**	**Practical setting**
FedAvg	97.93±0.05	98.81±0.01
FedProx	98.01±0.09	98.82±0.01
Per-FedAvg	99.63±0.02	98.90±0.05
pFedMe	99.75±0.02	99.52±0.02
Ditto	99.81±0.00	99.64±0.00
FedPer	99.70±0.02	99.47±0.04
FedRep	99.77±0.03	99.48±0.02
FedRoD	99.90±0.00	99.66±0.00
FedPHP	99.73±0.00	99.58±0.00
FedProto	99.85±0.01	99.63±0.03
GPFL	**99.92±0.07**	**99.72±0.04**

C. Various Statistical Heterogeneity

Table 2. The test accuracy (%) on TINY regarding various statistical heterogeneity.

	$\beta = 0.01$	$\beta = 0.5$
FedAvg	15.70±0.46	21.14±0.47
FedProx	15.66±0.36	21.22±0.47
Per-FedAvg	39.39±0.30	16.36±0.13
pFedMe	41.45±0.14	17.48±0.61
Ditto	50.62±0.02	18.98±0.05
FedPer	51.83±0.22	17.31±0.19
FedRep	55.43±0.15	16.74±0.09
FedRoD	49.17±0.06	23.23±0.11
FedPHP	48.63±0.02	21.09±0.07
FedProto	50.49±0.39	16.57±0.09
GPFL	**56.62±0.18**	**26.67±0.34**

---

*Corresponding author.

Table 3. The test accuracy (%) of variants of CoV on Tiny-ImageNet in practical *label skew* setting ($\beta = 0.1$). "*w/o*" is short for "without", and "Affine" is short for "affine transformation in LN".

	GPFL	FC Layers		LN		Activation Function		
	/	2 FC	3 FC	*w/o* LN	*w/o* Affine	ELU	Tanh	Sigmoid
4-layer CNN	43.37	44.16	**44.32**	43.64	34.99	43.95	43.97	43.79
ResNet-18	43.70	43.82	**43.97**	43.04	43.22	42.92	43.85	43.36

To further evaluate our GPFL under various statistical heterogeneity, we conduct additional two experiments by varying the degree of heterogeneity with different β in the practical *label skew* settings on TINY. The smaller the β is, the more heterogeneous the setting is. The range of $\beta \in \{0.01, 0.1, 0.5\}$ is widely used for CV tasks in FL [2, 1]. We have shown the results for $\beta = 0.1$ in the main body, and only show the results for $\beta \in \{0.01, 0.5\}$ here.

According to Table 2, the superiority of pFL methods is more evident in more heterogeneous settings, where less global information exists among clients. However, even in the extremely heterogeneous setting with $\beta = 0.01$, clients can still benefit from global information to facilitate their local model training. For example, using global parameter information, Ditto still achieves higher accuracy than pFedMe. In the scenario with a relatively large β and moderate heterogeneity, most pFL methods perform worse than the traditional FL methods, as they excessively focus on the client side. FedRoD performs well with a large β in the *label skew* settings because of its BSM loss. GPFL still performs the best since we train the GCE in an end-to-end manner, so GPFL can adapt to different environments.

D. Time Cost

Table 4. The time cost on Tiny-ImageNet using ResNet-18 in the practical *label skew* setting ($\beta = 0.1$).

	Total time	**Iterations**	**Avg. time**
FedAvg	365 min	230	1.59 min
FedProx	325 min	163	1.99 min
Per-FedAvg	121 min	34	3.56 min
pFedMe	1157 min	113	10.24 min
Ditto	318 min	57	5.58 min
FedPer	83 min	43	1.92 min
FedRep	471 min	115	4.09 min
FedRoD	87 min	50	1.74 min
FedPHP	264 min	65	4.06 min
FedProto	138 min	25	5.52 min
GPFL	171 min	75	2.28 min

We report the total time and the iteration amount required for convergence in Table 4. Some pFL methods cost relatively more total time than traditional FL methods, such as pFedMe and FedRep. As for the average time per iteration (avg. time for short), all the pFL methods cost more time as they introduce more operations for personalized tasks during local training. GPFL takes relatively little time due to the small computational overhead of embedding training.

E. CoV Design

By default, CoV consists of an FC layer, a ReLU activation, and a layer-normalization layer (denoted by LN). Here, we investigate the effect of different structures of CoV on GPFL by changing the number of FC layers, the activation function, and the LN. The accuracy results with an underline are higher than the one of GPFL, as shown in Table 3.

We found that the default structure of CoV in GPFL is not the best. Therefore, we can further improve GPFL by designing other structures for CoV. With more FC layers added into CoV, the accuracy on both the 4-layer CNN and ResNet-18 increases. However, more FC layers introduce more computational costs for the client. There is an accuracy-computation trade-off. We can add the FC layer appropriately on clients with sufficient computing resources. As for LN, removing it improves the performance of GPFL on the 4-layer CNN but not on ResNet-18. This difference also exists when we remove the affine transformation in LN. As for the activation function, using Tanh[1] instead of ReLU[2] improves the accuracy of GPFL on both the 4-layer CNN and ResNet-18, whereas ELU[3] and Sigmoid[4] do not.

---

[1] `https://pytorch.org/docs/stable/generated/torch.nn.Tanh.html`

[2] `https://pytorch.org/docs/stable/generated/torch.nn.ReLU.html`

[3] `https://pytorch.org/docs/stable/generated/torch.nn.ELU.html`

[4] `https://pytorch.org/docs/stable/generated/torch.nn.Sigmoid.html`

F. Experimental Details

F.1. Additional Implementation Details

We conduct experiments using six public datasets: Fashion-MNIST (FMNIST)[5], Cifar100[6], Tiny-ImageNet[7] (100K images with 200 categories), AG News[8] (a news classification dataset with four categories, more than 30K samples per category), Amazon Review[9] (a sentimental analysis dataset including four domains: Books, DVDs, Electronics and Kitchen Appliances), and *Human Activity Recognition* (HAR) dataset[10], where the sensor signal (accelerometer and gyroscope) data is collected from 30 users who perform six activities (WALKING, WALKING_UPSTAIRS, WALKING_DOWNSTAIRS, SITTING, STANDING, LAYING) wearing a smartphone (Samsung Galaxy S II) on the waist. Specifically, this dataset contains 9 dimensions of the inputs: (body_acc_, body_gyro_ and total_acc_) on the x, y, and z axes. On HAR, following previous work [3], we consider each dimension as a channel and use a HAR-CNN[11] to perform a 1D convolution on them and flat the output of each channel to a unified DNN layer after convolution and pooling. We run all experiments on a machine with two Intel Xeon Gold 6140 CPUs (36 cores), 128G memory, eight NVIDIA 2080 Ti GPUs, and CentOS 7.8.

F.2. Hyperparameters

For hyperparameter tuning, we use grid search to find optimal hyperparameters, including λ and μ. Specifically, grid search is performed in the following search space:

- λ: 0, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10
- μ: 0, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10

In this paper, we set $\lambda = 10^{-2}, \mu = 10^{-1}$ for the 4-layer CNN and 3-layer MLP, set $\lambda = 10^{-4}, \mu = 0$ for the ResNet-18 and the fastText, and set $\lambda = 10^{-2}, \mu = 1$ for HAR-CNN.

F.3. Number of FC Layers in the Personalized Head

Although we split the given backbones into a feature extractor and a personalized head following FedRep, we do not modify the structure of the backbones, such as adding or deleting layers. Here, we study the effect of the number of FC layers in the personalized head on the 4-layer CNN and the 3-layer MLP, as shown in Table 5. Note that once we add an FC layer in the backbone to the personalized head, this FC layer is removed from the feature extractor so that the combination of the feature extractor and the personalized head is still the backbone.

Table 5. The test accuracy (%) on the 4-layer CNN and the 3-layer MLP in the practical *label skew* setting ($\beta = 0.1$). "nFC" is short for "number of FC layers in the personalized head". Recall that K is the dimension of the feature vector.

Backbone	4-layer CNN				3-layer MLP		
Dataset	FMNIST		Cifar100		Amazon Review		
nFC	1	2	1	2	1	2	3
K	512	1024	512	1600	100	500	1000
Acc.	99.72	98.89	61.86	58.67	89.32	88.84	88.26

According to Table 5, the test accuracy decreases as the nFC increases. More nFC means fewer layers in the feature extractor, which reduces the feature extraction ability of the feature extractor, thus reducing the performance of GPFL. Besides, a larger K also requires more memory to store trainable category embeddings. Therefore, we set nFC to 1 unless explicitly mentioned.

G. Societal Impacts

As mentioned in the main body, pFL has three features: privacy-preserving, collaborative learning, and personalization. We devise GPFL based on the pFL scheme, so GPFL also has these three features. Moreover, the excellent performance in terms of scalability, fairness, stability, and privacy demonstrates the ability of GPFL to meet the needs of massive clients in complex real-world environments.

H. Data Distribution Visualization

We illustrate the data distributions (including training and test data) in the experiments here.

References

[1] Qinbin Li, Bingsheng He, and Dawn Song. Model-Contrastive Federated Learning. In *CVPR*, 2021. 2

[2] Tao Lin, Lingjing Kong, Sebastian U Stich, and Martin Jaggi. Ensemble Distillation for Robust Model Fusion in Federated Learning. In *NeurIPS*, 2020. 2

[3] Ming Zeng, Le T Nguyen, Bo Yu, Ole J Mengshoel, Jiang Zhu, Pang Wu, and Joy Zhang. Convolutional neural networks for human activity recognition using mobile sensors. In *6th international conference on mobile computing, applications and services*, pages 197–205. IEEE, 2014. 3

---

[5] https://pytorch.org/vision/stable/datasets.html#fmnist

[6] https://pytorch.org/vision/stable/datasets.html#cifar

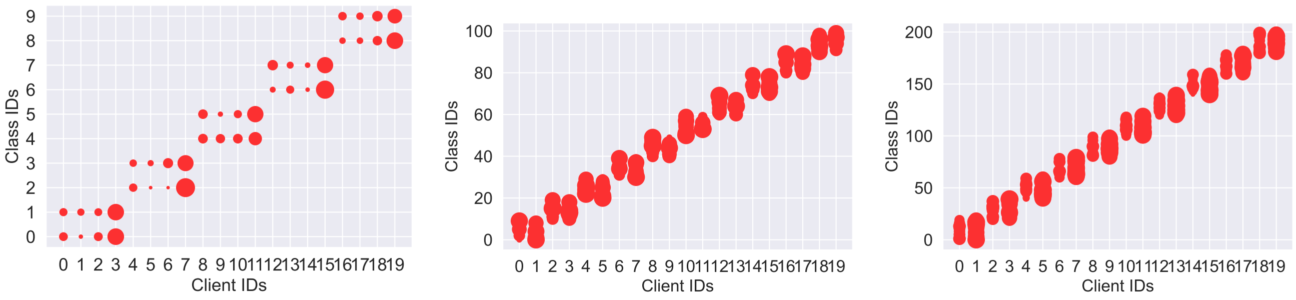
[7] http://cs231n.stanford.edu/tiny-imagenet-200.zip

[8] https://pytorch.org/text/stable/datasets.html#ag-news

[9] https://github.com/FengHZ/KD3A#install-datasets

[10] https://archive.ics.uci.edu/ml/datasets/human+activity+recognition+using+smartphones

[11] https://github.com/jindongwang/Deep-learning-activity-recognition/blob/master/pytorch/network.py

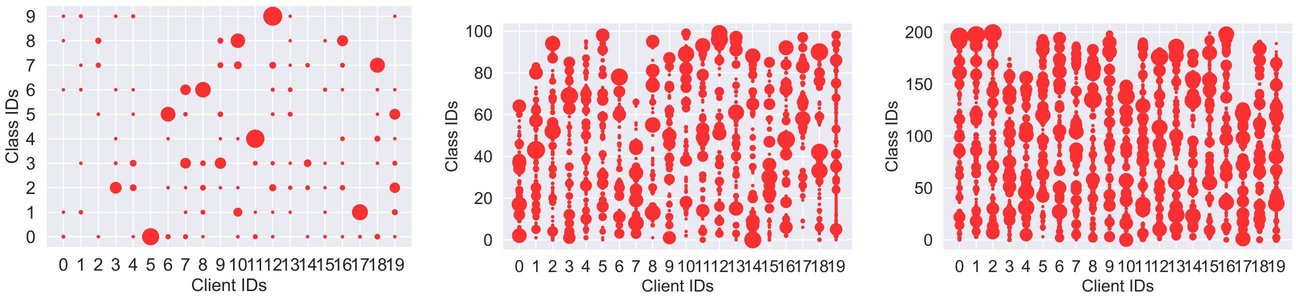


(a) FMNIST

(b) Cifar100

(c) Tiny-ImageNet

Figure 2. The data distribution of each client on FMNIST, Cifar100, and Tiny-ImageNet, respectively, in the pathological *label skew* settings. The size of a circle represents the number of samples.

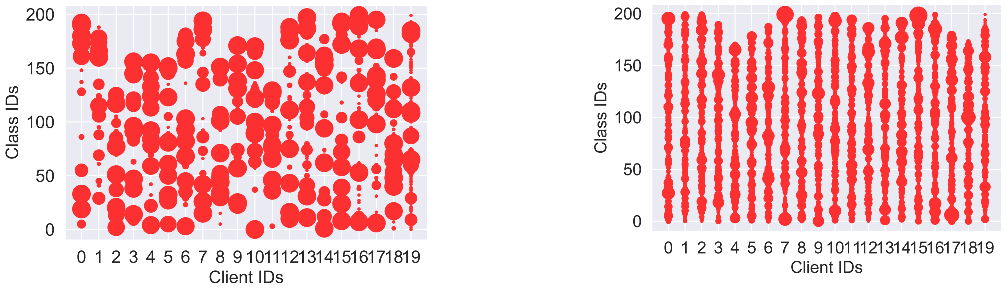


(a) FMNIST

(b) Cifar100

(c) Tiny-ImageNet

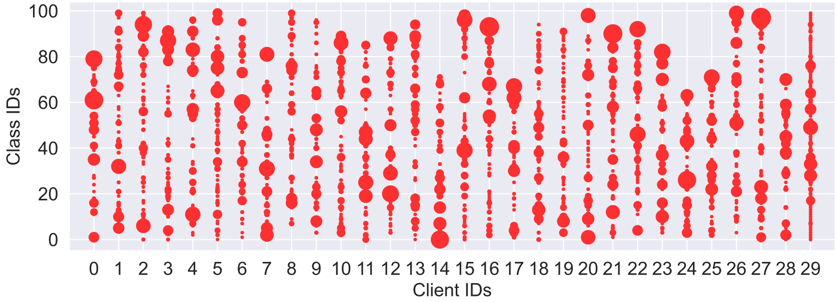
Figure 3. The data distribution of each client on FMNIST, Cifar100, and Tiny-ImageNet, respectively, in practical *label skew* settings ($\beta = 0.1$). The size of a circle represents the number of samples.



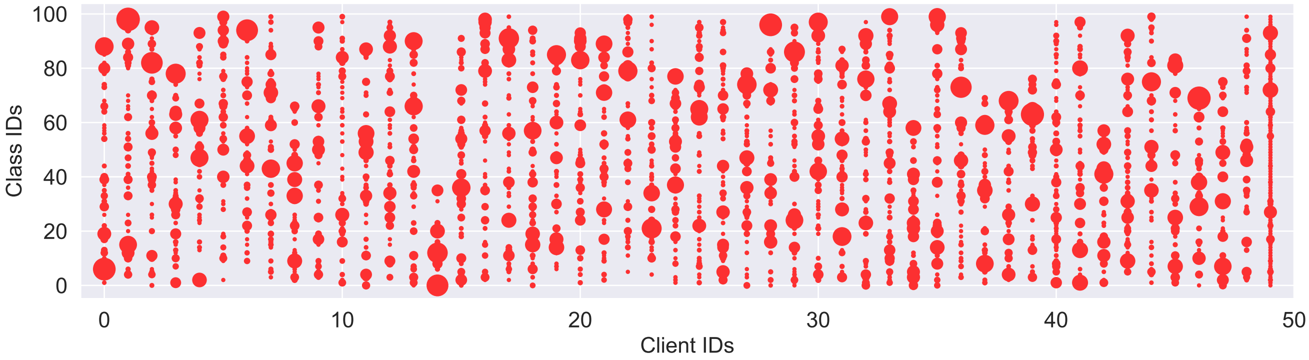
(a) $\beta = 0.01$

(b) $\beta = 0.5$

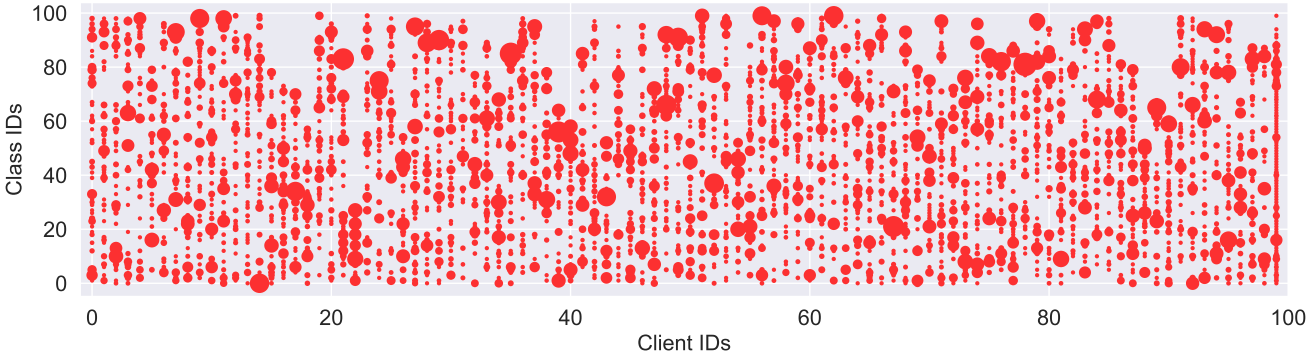
Figure 4. The data distribution on all clients on Tiny-ImageNet in two additional practical *label skew* settings. The size of a circle represents the number of samples. The degree of heterogeneity decreases as β in $Dir(\beta)$ increases.



(a) 30 clients

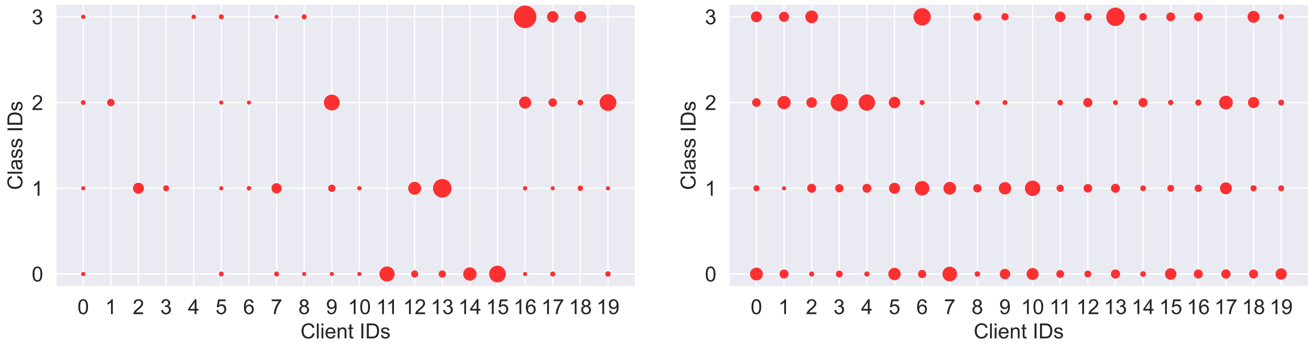


(b) 50 clients



(c) 100 clients

Figure 5. The data distribution of each client on Cifar100 in the practical *label skew* setting ($\beta = 0.1$) with 30, 50, and 100 clients, respectively. The size of a circle represents the number of samples.



(a) $\beta = 0.1$

(b) $\beta = 1$

Figure 6. The data distribution of each client on AG News in the practical *label skew* settings. The size of a circle represents the number of samples.

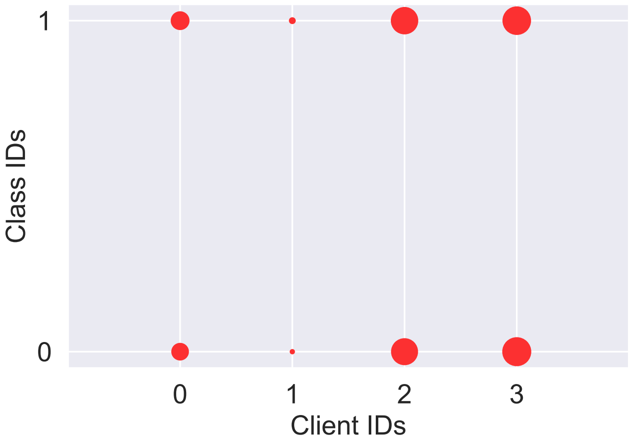


Figure 7. The data distribution of each client on Amazon Review in *feature shift* settings. The size of a circle represents the number of samples.

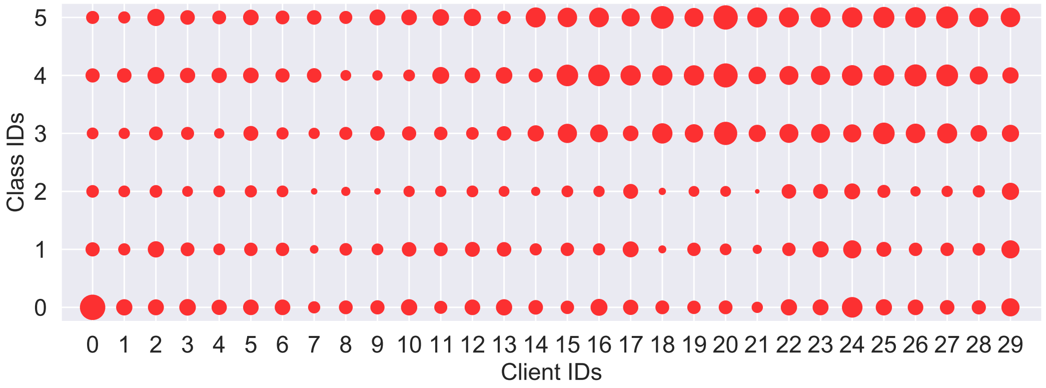


Figure 8. The data distribution of each client on HAR in *real-world* settings. The size of a circle represents the number of samples.