Data Resampling for Federated Learning with Non-IID Labels

Zhenheng Tang[1] , **Zhikai Hu**[1] , **Shaohuai Shi**[2] , **Yiu-ming Cheung**[1] , **Yilun Jin**[2] , **Zhenghang Ren**[2] , **Xiaowen Chu**[1]

[1] Department of Computer Science, Hong Kong Baptist University

[2]Department of Computer Science and Engineering, The Hong Kong University of Science and Technology

{zhtang, cszkhu, ymc, chxw}@comp.hkbu.edu.hk
shaohuais@cse.ust.hk
{yilun.jin, zrenak}@connect.ust.hk

Abstract

Recently, federated learning has received increasing attention from academe and industry, since it makes training models with decentralized data possible. However, most existing federated learning approaches suffer from Non-Independent and Identically data distribution in clients. Observing that each client has an imbalanced label distribution in many federated learning scenarios, we examine the effects of combining imbalanced learning techniques with federated learning. Through comprehensive experiments, we obtain the following findings: (1) By data resampling, the label sampling probabilities are made more similar across clients, which leads to faster convergence; (2) Imbalanced data resampling results in final accuracy decreasing on local dataset. Based on these two key findings, we propose a simple but effective data resampling strategy named Imbalanced Weight Decay Sampling (IWDS) that dynamically regulates the sampling probability of labels, remarkably accelerating the training process. The effectiveness of IWDS has been verified on several modern federated learning algorithms such as FedAvg, FedProx, and FedNova.

1 Introduction

In recent years, with the help of federated learning, many cross-silo organizations and cross-device users can train the same deep learning model collaboratively without sharing their data. Thus, the data privacy can be protected to some extend. There has been a lot of applications benefiting from federated learning, like Google keyboard [Yang *et al.*, 2018], real-world image classification [Hsu *et al.*, 2020], and object detection [Luo *et al.*, 2019][He *et al.*, 2021].

Federated learning (FL) is a distributed learning paradigm that can make use of decentralized datasets to train a global deep learning model or many personalized models, protecting the privacy of clients. Formally, the objective function is as follows:

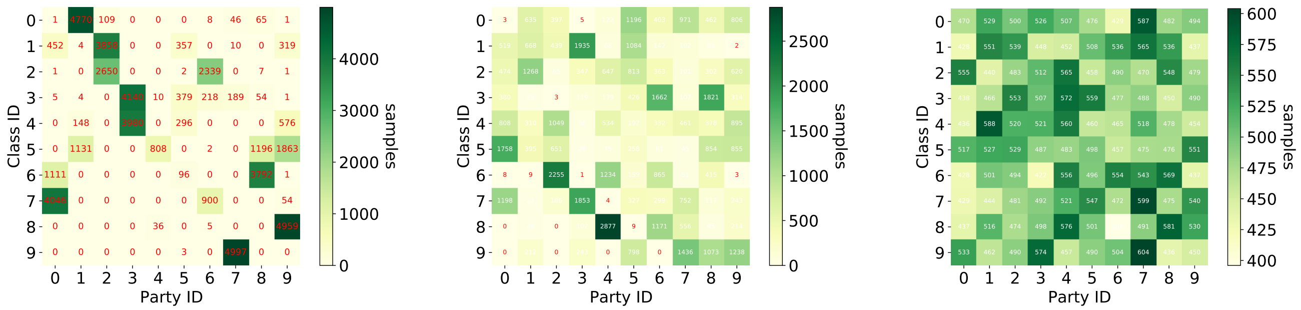
$$\min_{\boldsymbol{W}} F(\boldsymbol{W}) \overset{\text{def}}{=} \min_{\boldsymbol{W}} \sum_{k=1}^{K} \frac{N_k}{N} \cdot f_k(\boldsymbol{W}),$$
$$f_k(\boldsymbol{W}) = \frac{1}{N_k} \sum_{i=1}^{N_k} \ell(\boldsymbol{W}; \boldsymbol{X}_{(k,i)}, \boldsymbol{Y}_{(k,i)}), \tag{1}$$

in which, N_k is the size of dataset $\mathcal{D}_k$ on client k, N is the total size of all datasets, $\boldsymbol{X}_{(k,i)}$ is the i-th input data in $\mathcal{D}_k$, $\boldsymbol{Y}_{(k,i)}$ is the i-th label in $\mathcal{D}_k$, $\boldsymbol{W}$ is the model weights, $f_k(\boldsymbol{W})$ is the k-th client's local objective function that measures the local empirical risk on $\mathcal{D}_k$, and ℓ is the loss function of the global model.

Federated Average (FedAvg) [McMahan *et al.*, 2017] is a communication-efficient algorithm broadly used in many FL applications like computer vision and natural language processing [Hsu *et al.*, 2020][Yang *et al.*, 2018]. However, in FL, datasets of clients are collected from different environments like the smart city [Liu *et al.*, 2020], and cannot be shared among different clients and servers. This kind of dataset distribution is called Non-Independent and Identically Distributed (Non-I.I.D.). Under this data distribution, the performance of FedAvg plummets experimentally and theoretically [Hsu *et al.*, 2020][Li *et al.*, 2020b][Hsu *et al.*, 2019][Li *et al.*, 2020b].

It is not uncommon that different clients have different amounts of data samples for some labels. This label distribution skew is a type of Non-I.I.D. that has received the most attention in recent years [Hsu *et al.*, 2019][Wang *et al.*, 2020b], and is the focus of this paper. In [Hsu *et al.*, 2019], Hsu et al. proposed a partition method, Label-based Dirichlet Partition (LDA), using Dirichlet Sampling with a hyper-parameter α_d to simulate Label-based Non-I.I.D. data distribution. This is a common practice in FL research to obtain synthetic federated datasets [Wang *et al.*, 2020b][Reddi *et al.*, 2021]. The visualization of the Non-I.I.D. datasets sampled from CIFAR-10 with different α_d is shown in Figure 1.

As shown in Figure 1(a), the amounts of data samples of different labels in each client are different. For example, client 0 has 452 samples of class 1, but 4046 samples of class 7. Under this observation, we find that the learning process of one client (without communication) can be seen as the *imbalanced learning* [Cao *et al.*, 2019]. Those classes with less amount of samples are called **tail class**, and classes with larger amount



(a) CIFAR-10 with $\alpha_d = 0.1$ (b) CIFAR-10 with $\alpha_d = 0.5$ (c) CIFAR-10 with $\alpha_d = 100.0$

Figure 1: Visualization of CIFAR-10 on 10 clients with LDA. The α_d is the parameter controlling Non-I.I.D. degree in LDA partition.

of samples are called **dominant class**. This kind of dataset is also called **imbalanced dataset** [Cao *et al.*, 2019].

Intuitively, we hope to assign tail classes higher sampling weight to assist in model training, which is a common practice, called **Data Resampling** [Buda *et al.*, 2018] in imbalanced learning. However, there is a key difference between imbalanced learning and FL. In FL, although each client has a **local imbalanced dataset**, the aggregated dataset of all clients actually may be a **global balanced dataset**, like the Figure 1 shows. And FL clients can learn the knowledge of tail classes through communication with other clients, which is inapplicable in imbalanced learning.

To understand the effect of directly using the data resampling in FL, we conduct some preliminary experiments and have the following two interesting findings:

1. *By data resampling, the label sampling probabilities are made more similar across clients, which leads to faster convergence.* Other researchers [Hsu *et al.*, 2019][Li *et al.*, 2020b] have experimentally verified that similar label distribution could improve FL training. We find out that the clients could apply data resampling to get a more similar label sampling probability distribution, which can also accelerate training convergence.
2. *Imbalanced data resampling results in final accuracy decreasing on local dataset.* Data resampling could make the models over-fit on the tail classes or under-fit on the dominant classes [Cao *et al.*, 2019]. Thus, the potential reason of phenomenon may be that the imbalanced data resampling harms "Special Knowledge" (local dataset) learning of individual clients in FL, resulting in low final accuracy.

Based on these observations and preliminary experiment results, we propose a simple but effective data resampling strategy for FL, **Imbalanced Weight Decay Sampling (IWDS)**. In this method, the sampling weights of all data samples decay along the training time. Therefore, we can let clients have more similar label sampling probability at the early stage to converge faster, and original label sampling probability at the later stage to better learn their special knowledge.

There have been many novel algorithms proposed to address the Non-I.I.D. problem [Li *et al.*, 2020a][Wang *et al.*, 2020b]. These algorithms aim to modify the optimization scheme, loss function, or regularization methods. Our proposed method is orthogonal to these existing approaches, and can be directly applied to not only FedAvg [McMahan *et al.*, 2017], but also other recently developed FL algorithms like FedProx [Li *et al.*, 2020a] and FedNova [Wang *et al.*, 2020b].

We summarize our contributions as below:

- We conduct experiments to verify the effect of the naive data resampling technique in FL, and conclude two key findings of data resampling in FL.
- We propose a simple but effective data resampling strategy IWDS for FL usage. It can be seamlessly combined with many current FL algorithms without any extra expense or data communication.
- We conduct comprehensive experiments to verify the improvement on the convergence and performance of FL training with IWDS. And we also examine the effect of combining it with other FL algorithms.

2 Related work

2.1 Federated Learning

FedAvg [McMahan *et al.*, 2017] makes clients do some local updates and then communicate. The models of chosen clients will be averaged on the server side. This simple average operation takes into some convergence problems especially under Non-I.I.D datasets [Li *et al.*, 2020b]. FedProx [Li *et al.*, 2020a] can be seen as a kind of client model regularization, which tries to make client models closer to the global model during every update iteration.

FedNova [Wang *et al.*, 2020b] is proposed to normalize the accumulated gradients of clients and dynamically scale the effect of the aggregated updates. This kind of algorithm focuses on designing new distributed optimization schemes. The data resampling technique can be directly integrated into these algorithms.

There are some research works [Li *et al.*, 2020a][Duan *et al.*, 2019] proposed to do a more intelligent client sampling rather than random sampling in FedAvg. These methods try to alleviate the client drift by choosing clients in a better way. Because each client has a special dataset and choosing clients implies choosing datasets, this kind of algorithms can be regarded as a "global data sampling" technique. Usually, these algorithms need to communicate extra information like local data distribution, or require more server operations like computation of local model similarities. Comparing with these methods, our data resampling technique is a kind of "local data sampling" method without any extra communication or computation. And our method can be naturally combined with those "global data sampling" techniques.

2.2 Class-imbalanced learning

Many real-world datasets have long-tailed label distribution [Liu *et al.*, 2019][Van Horn *et al.*, 2018]. Directly training on these datasets normally has a poor performance on those classes with less amount of data samples. Over-sampling the minority classes [Buda *et al.*, 2018][Byrd and Lipton, 2019] and under-sampling the frequent classes [Buda *et al.*, 2018] are two kinds of broadly used sampling strategies. Over-sampling could make the models over-fitting on the tail classes, and under-sampling cannot work well when data imbalance is extreme [Cao *et al.*, 2019].

In [Cao *et al.*, 2019], the authors proposed to utilize a two-stage data resampling strategy named deferred resampling. They train the model normally at the first stage and then use data resampling to train the model at the second stage. This method and our IWDS are both dynamic data resampling strategy, i.e., assigning different sampling weights at different training stages. A benchmark work of imbalanced learning [Kang *et al.*, 2020] also summarizes dynamic data resampling strategy as the progressively-balanced sampling.

Interestingly, the changing of sampling weights of our IWDS along the training epoch is in the exactly reverse direction of the deferred resampling [Cao *et al.*, 2019]. This is because we make a totally different assumption from [Cao *et al.*, 2019]: we assume that the resampling makes clients have more similar label sampling probabilities, thus accelerating the convergence in FL. However, deferred resampling is based on the assumption that training without resampling could make models have a good initial representation in imbalanced learning [Cao *et al.*, 2019].

3 FL with Data Resampling

In this section, we firstly introduce how to directly use a classical Data Resampling method in FedAvg. Then, we conduct some experiments and analyze the results. Based on these results and analyses, we propose the Imbalanced Weight Decay Sampling (IWDS) in FL.

3.1 Directly using Data Resampling in FedAvg

The effective number of samples [Cui *et al.*, 2019] is proposed to measure the benefit to learning of the volume of samples. And authors propose a simple formula to estimate the effective number of samples as $(1-\beta^n)/(1-\beta)$, in which n is the number of samples and $\beta \in [0,1)$ is a hyperparameter. Then, based on the effective number of samples, they propose to do data resampling by the inverse effective number of samples and attain a great improvement.

Here, we explore using data resampling with the inverse effective number of samples in FL. Assuming there are K clients with datasets $\mathcal{D}_0, \mathcal{D}_1, \cdots, \mathcal{D}_K$ respectively, and the amount of samples of label $c \in \{1, 2, \cdots, C\}$ in $\mathcal{D}_k$ is $N_{k,c}$, where C is the total number of classes. For the i-th data sample in dataset $\mathcal{D}_k$, we set the sampling weight of it based on the amount of its label $\boldsymbol{Y}_{(k,i)}$ in $\mathcal{D}_k$:

$$w_{k,i} = \frac{1-\beta}{1-\beta^{N_{k,\boldsymbol{Y}_{(k,i)}}}}, \tag{2}$$

where the $N_{k,\boldsymbol{Y}_{(k,i)}}$ is the amount of samples of label $\boldsymbol{Y}_{(k,i)}$ in dataset $\mathcal{D}_k$. [1]

Then the probability of that client k samples the i-th data sample is

$$p(k,i) = \frac{w_{k,i}}{\sum_i^{N_k} w_{k,i}}. \tag{3}$$

Note that no clients need to send any data statistic of them to the server or other clients, strictly protecting data privacy, and they only need to calculate the sampling probability at the beginning of the training, which requires very little computation.

With the data resampling, the probability of that client k samples one data that has the label c is $q(k,c) = N_{k,c}p(k,i), (\boldsymbol{Y}_{(k,i)} = c)$. We name it as the **label sampling probability**. Therefore, the ratio of sampling probability between two different classes is

$$\frac{q(k,c_1)}{q(k,c_2)} = \frac{N_{k,c_1}p(k,i_1)}{N_{k,c_2}p(k,i_2)} = \frac{N_{k,c_1}(1-\beta^{N_{k,c_2}})}{N_{k,c_2}(1-\beta^{N_{k,c_1}})}. \tag{4}$$

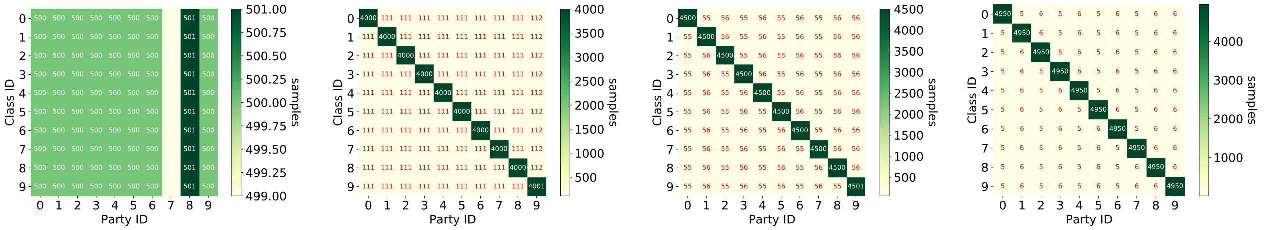
Assuming we have a tail class with 5 samples, and a dominant class with 4950 samples. With uniform sampling, the ratio of sampling probability between the tail class and the dominant class is $\frac{5}{4950}$. When fixing $\beta = 0.9999$ [Cao *et al.*, 2019], the ratio changes into $\frac{5\times(1-0.9999^{4950})}{4950\times(1-0.9999^{5})} \simeq 0.7889$. Therefore, the sampling probability between these two labels become similar, making the label sampling probability between any two clients become similar.

Non-I.I.D. long-tail distribution

Besides the LDA partition, We utilize another kind of partition method to simulate the Non-I.I.D. datasets, each of which has a long tail distribution. Assuming we have $K(K=C)$ clients, and a global dataset $\mathcal{D}$ of C same-amount classes, we define a parameter α_l to control the Non-I.I.D. degree. For each class c, we partition all samples of it into K parts, one of which has $\alpha_l N_{*,c}$ samples that will be given to client $k(k=c)$, and each of other $K-1$ parts has $\frac{1-\alpha_l}{K-1}N_{*,y}$ samples that will be given to other clients $k(k \neq c)$. So each client has one dominant class and $C-1$ tail classes. Figure 2 shows the visualization of CIFAR-10 dataset with LLT partition.

We call this kind of Non-I.I.D. partition method as local long-tail (LLT) partition, which is also utilized in [Wang *et al.*, 2020a]. This kind of data distribution may also appear in many real-world applications, e.g. smartphone users like to take many photos of some styles that they like, and few photos of styles they dislike. Note that the LLT partition may not well simulate the real-world FL data distribution as LDA partition do. Nevertheless, because it looks like a classical imbalanced learning scenario, we start from this scenario to verify the effect of data resampling in FL, then we extend to broader scenarios.

[1] Note that in one dataset $\mathcal{D}_k$, for those samples with the same label c, they have the same sampling weights, i.e. $w_{k,i} = w_{k,j}$ and $N_{k,\boldsymbol{Y}_{(k,i)}} = N_{k,\boldsymbol{Y}_{(k,j)}} = N_{k,c}$ when $\boldsymbol{Y}_{(k,i)} = \boldsymbol{Y}_{(k,j)}$.



(a) CIFAR-10 with $\alpha_l = 0.1$ (b) CIFAR-10 with $\alpha_l = 0.8$ (c) CIFAR-10 with $\alpha_l = 0.9$ (d) CIFAR-10 with $\alpha_l = 0.99$

Figure 2: Visualization of CIFAR-10 dataset on 10 clients with LLT partition. When $\alpha_l = 0.1$ we have a label-balanced data distribution on every client. With the growth of α_l, the amount of the dominant class in each client increase.

Preliminary results

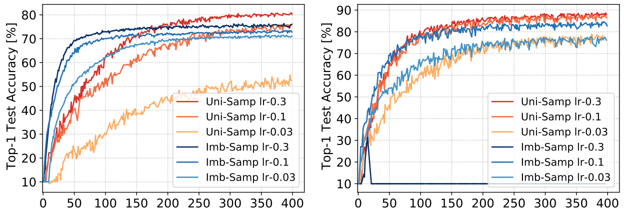
We conduct some preliminary experiments to examine the effect of directly using Data Resampling in FedAvg. We use FedAvg algorithm to train VGG-9 on CIFAR-10 datasets with LLT and LDA partition. The number of total clients is 10, and 5 clients will be randomly chosen in each communication round. We use SGD optimization without momentum.

The experiment results are shown in Figure 3. It is obvious that the convergence is greatly accelerated when trained on LLT partitioned CIFAR 10, and slightly accelerated when trained on LDA partitioned CIFAR-10. But the ultimate performance of using Imb-Samp drops several percents.

So we obtain the two key findings as in Section 1 for this experiment results:

1. *By data resampling, the label sampling probability are made more similar across clients, which leads to faster convergence.*
2. *Imbalanced data resampling results in final accuracy decreasing on local dataset.*

These two findings can be more convincingly supported by the experiment results in Section 4.



(a) LLT partition with $\alpha_l = 0.9$ (b) LDA partition with $\alpha_d = 0.5$

Figure 3: FedAvg on CIFAR-10 dataset with 10 clients. Uni-Samp (Uniform Sampling) means each data sample has the same sampling weight, and the Imb-Samp (Imbalance Sampling) means each data sample use the inverse of effective number as the sampling weight.

3.2 Imbalanced Weight Decay Sampling

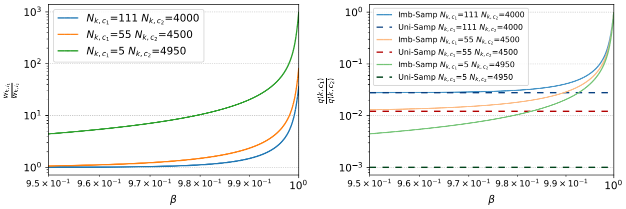
Based on the two findings, we propose to decay the weights in the imbalance resampling with the training time, such that the convergence can be accelerated at early stage and the model can learn more information of its own special knowledge well in the late training stage.

In the formula 2, we note that we can dynamically adjust the value of β to adjust the ratio of sampling probability between tail class and dominant class. The Figure 4(a) shows the relationship between β and the ratio of the sampling probability of sample i_1 of tail class c_1 and sample i_2 of dominant class c_2, which is calculated as

$$\frac{p(k, i_1)}{p(k, i_2)} = \frac{w_{k,i_1}}{w_{k,i_2}} = \frac{1 - \beta^{N_{k,c_2}}}{1 - \beta^{N_{k,c_1}}}. \tag{5}$$

From Figure 4(a), we can see that the higher β could make samples of tail class have higher sampling probability than dominant class.

The Figure 4(b) shows the ratio of the sampling probability $\frac{q(k,c_1)}{q(k,c_2)}$ of tail class and dominant class. When using uniform sampling, it is calculated as $\frac{N_{k,c_1}}{N_{k,c_2}}$. When using imbalance resampling, it is calculated as $\frac{N_{k,c_1}(1-\beta^{N_{k,c_2}})}{N_{k,c_2}(1-\beta^{N_{k,c_1}})}$. When β converges to 1, the $\frac{q(k,c_1)}{q(k,c_2)}$ converges to 1, meaning that c_1 and c_2 have a similar sampling probability. When β becomes smaller, $\frac{q(k,c_1)}{q(k,c_2)}$ becomes closer to it with uniform sampling.



(a) The ratio of the sampling weights of two samples of tail class and dominant class (b) The ratio of the sampling probability $\frac{q(k,c_1)}{q(k,c_2)}$ of tail class and dominant class

Figure 4: The relationship between the value of β and how much it can re-balance two classes.

So, in order to make clients have similar label sampling probability by re-balancing, we should firstly use high β at early training stages. And then, to diminish the effect of imbalance resampling, we gradually decay it to a value that makes $\frac{q(k,c_1)}{q(k,c_2)}$ close to uniform sampling case. Therefore, we change the formula 2 into the following form:

$$w_{k,i,t} = (1 - \beta_t)/(1 - \beta_t^{N_{k,\mathbf{Y}(k,i)}}),$$

in which t is the t-th communication round. The β_t is updated during each communication round as:

$$\beta_t = \beta_m + (\beta_0 - \beta_m) * \rho^t, \tag{6}$$

in which ρ is the decay rate. This equation makes the β decay from β_0 to β_m with the exponential rate ρ.

One advantage of dynamically decaying the β instead of just removing the data resampling is that, for some extreme imbalance cases like $N_{k,c_1} = 5, N_{k,c_2} = 4950$, we may still need to do re-balance them at some degree in the late training period because this huge imbalance may make clients difficult to learn. As shown in Figure 4(b), for the same value of β, these extreme imbalance classes ($N_{k,c_1} = 5, N_{k,c_2} = 4950$) could still be re-balanced more than those slight imbalance classes ($N_{k,c_1} = 111, N_{k,c_2} = 4000$).

4 Experiment results

4.1 Experiment settings

In this section, we verify the effect of IWDS by comparing it with Uni-Samp and Imb-Samp. And we adjust the β_m to observe its effect. Besides the FedAvg algorithm, we also conduct experiments with FedProx and FedNova to test the effect of combing IWDS with different FL algorithms. The experiment settings are as follows.

Datasets and models. We evaluate our methods on CIFAR-10 with VGG-9, and Fashion-MNIST with a simple CNN used in [McMahan *et al.*, 2017].

Datasets partition. We conduct experiments of FL with 10 clients, and 5 clients will be chosen in every communication round. For both two datasets, 4 different ways of data partition are tested: LDA partition with $\alpha_d = 0.5$ and $\alpha_d = 0.1$, LLT partition with $\alpha_l = 0.9$ and $\alpha_l = 0.99$. The visualization of data distribution is shown in Figure 1 and 2.

Hyper-parameters. For all experiments, the batch size is fixed as 32. We test the learning rate in $[0.01, 0.03, 0.1, 0.3, 0.5]$ and show the best result for all algorithms. For FedAvg and FedProx, the momentum is set as 0, same with the original paper [McMahan *et al.*, 2017][Li *et al.*, 2020a]. And for FedNova, we set the momentum as 0.9, to see whether we could combine the momentum acceleration in FedNova with our method. For the Imb-Samp, the β is fixed as 0.9999. For IWDS, The β_0 is set as 0.9999, which is the same as [Cui *et al.*, 2019]. We evaluate different values of β_m in $[0.95, 0.98, 0.99]$, and the sampling weight decay rate ρ is set as 0.992. And we also do learning rate decay with 0.992 each communication round, same with [McMahan *et al.*, 2017].

Our code framework is based on FedML [He *et al.*, 2020], which is a widely used FL library.

4.2 Experiment results

All of our experiment results are shown in Figure 5, 6, 7, and 8. And we summarize the top-1 validation accuracy of different experiments in Table 1.

Improvement on FedAvg. We examine the effect of applying IWDS on FedAvg, trainnig with CIFAR-10 and Fashion-MNIST. The experiment results are shown in Figure 5 and 6. For LLT partition, we can see our IWDS remarkably improves the convergence speed and ultimate performance. However, when using LDA partition, the speedup is only marginal, and the ultimate performance cannot get much improvement. a **Improvement on FedProx.** We examine the effect of combining FedProx with our data resampling method on CIFAR-10. The experiment results are shown in Figure 7. When we use LLT partition, The speedup of convergence is remarkable.

Table 1: Comparison of top-1 validation accuracy.

FL Alg.	Method	CIFAR-10 LLT		CIFAR-10 LDA	
		$\alpha_l = 0.9$	$\alpha_l = 0.99$	$\alpha_d = 0.1$	$\alpha_d = 0.5$
FedAvg	Uni-Samp	80.81	35.9	**84.71**	**88.49**
	Imb-Samp	76.31	43.4	80.31	84.44
	IWDS $\beta_m = 0.95$	**80.26**	47.19	83.66	87.2
	IWDS $\beta_m = 0.98$	80.06	47.16	83.33	87.08
	IWDS $\beta_m = 0.99$	79.95	**47.26**	82.75	86.34
FedProx	Uni-Samp	76.49	26.28	**81.26**	81.55
	Imb-Samp	72.23	41.45	75.95	**86.17**
	IWDS $\beta_m = 0.95$	75.79	46.10	80.56	85.14
	IWDS $\beta_m = 0.98$	**79.08**	**47.39**	80.37	85.15
	IWDS $\beta_m = 0.99$	76.85	46.56	80.09	85.09
FedNova	Uni-Samp	10.02	10.02	89.54	**87.07**
	Imb-Samp	79.10	49.91	87.37	84.69
	IWDS $\beta_m = 0.95$	81.90	**57.29**	**89.55**	85.61
	IWDS $\beta_m = 0.98$	82.02	54.89	89.03	85.95
	IWDS $\beta_m = 0.99$	**82.23**	54.83	88.92	85.78

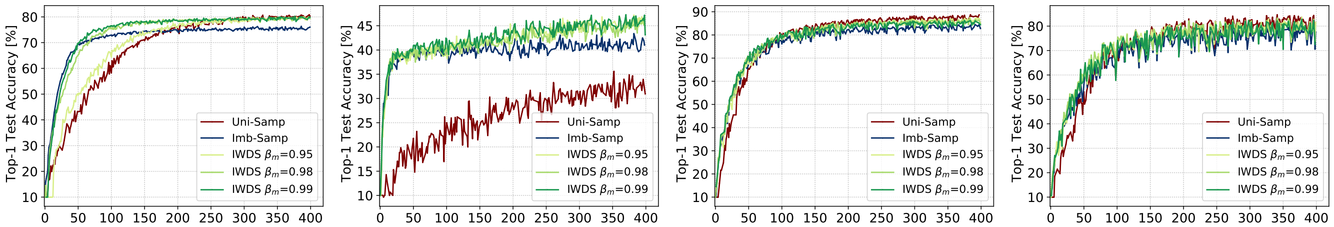
FL Alg.	Method	Fashion-MNIST LLT		Fashion-MNIST LDA	
		$\alpha_l = 0.9$	$\alpha_l = 0.99$	$\alpha_d = 0.1$	$\alpha_d = 0.5$
FedAvg	Uni-Samp	58.81	37.42	86.29	**91.29**
	Imb-Samp	89.25	84.31	87.82	91.14
	IWDS $\beta_m = 0.95$	88.86	83.65	87.43	91.22
	IWDS $\beta_m = 0.98$	89.23	84.29	87.63	91.21
	IWDS $\beta_m = 0.99$	**89.28**	**84.42**	**88.04**	91.18

Improvement on FedNova. We examine the effect of combining FedProx with IWDS on CIFAR-10. Comparing Figure 8 with 5 and 7, FedNova with Uni-Samp brings into convergence acceleration. However, FedNova with Uni-Samp cannot converge when using LLT partition. But the Imb-Samp and our method IWDS can help Fednova converge well. Comparing Figure 8(a-b) with Figure 5(a-b) and Figure 7(a-b), we can see that, with IWDS, FedNova with Momentum SGD attains an obvious speedup and more improvement of ultimate performance than FedProx and FedAvg.

Effect of β_m. Note that the Imb-Samp can be seen as a special case of IWDS when $\beta_m = 0.9999$. Combining them together, we can see that the value of β_m will influence the convergence a lot, like Figure 5(a-b), 7(a) and 8(a-b). But we can still find some interesting things from these results. For CIFAR-10 dataset with LLT partition with $\alpha_l = 0.9$, IWDS with $\beta_m = 0.99$ achieves the best trade-off of ultimate performance and convergence speed when using FedAvg, IWDS with $\beta_m = 0.98$ achieves the best trade-off when using FedProx, and IWDS with $\beta_m = 0.99$ achieves the best trade-off when using FedNova.

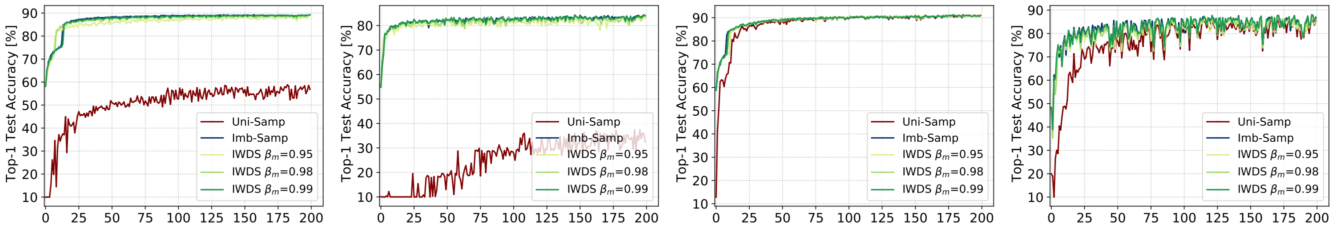
Comparison of LLT and LDA. From Figure 5, 6, 7, 8 and Table 1, we can see the huge improvement of our method on LLT data partition. However, on LDA partition, our method cannot benefit the ultimate performance. We make a concise analysis here.

Let's recall the characteristics of these two different distributions from Figure 1 and 2. In the LLT data partition, each client has at least one sample of each class. However, in the LDA data partition, many clients have no samples of some classes, especially when $\alpha_d = 0.1$. Therefore, it is difficult for the "local data sampling" methods to make clients have a more similar label sampling probability.



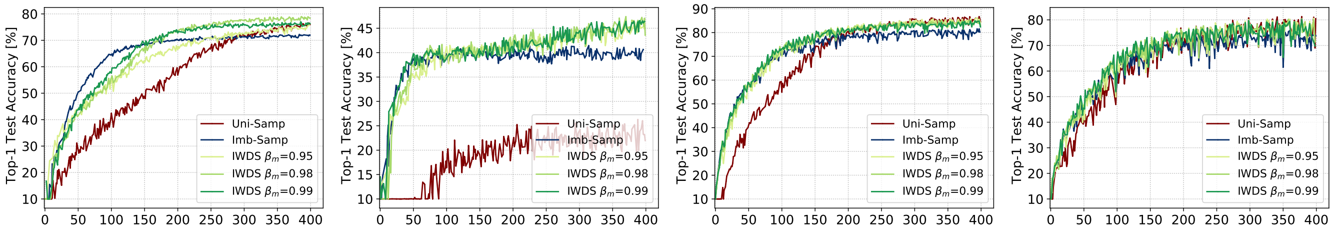
(a) LLT partition with $\alpha_l = 0.9$ (b) LLT partition with $\alpha_l = 0.99$ (c) LDA partition with $\alpha_d = 0.5$ (d) LDA partition $\alpha_d = 0.1$

Figure 5: Test accuracy of FedAvg, using VGG-9 on CIFAR-10.



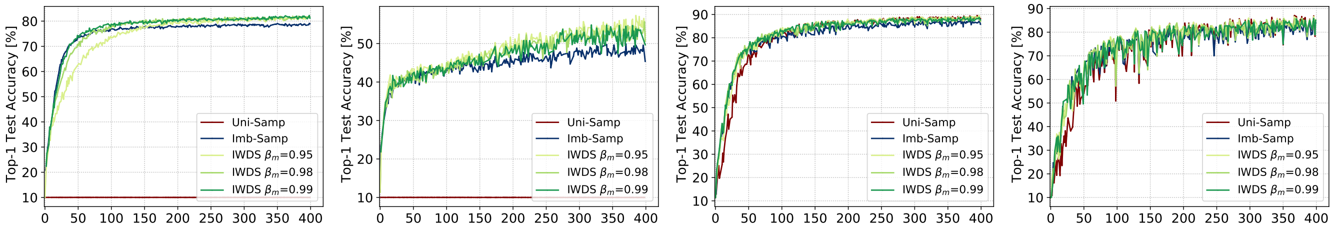
(a) LLT partition with $\alpha_l = 0.9$ (b) LLT partition with $\alpha_l = 0.99$ (c) LDA partition with $\alpha_d = 0.5$ (d) LDA partition $\alpha_d = 0.1$

Figure 6: Test accuracy of FedAvg, using a simple CNN on Fashion-MNIST.



(a) LLT partition with $\alpha_l = 0.9$ (b) LLT partition with $\alpha_l = 0.99$ (c) LDA partition with $\alpha_d = 0.5$ (d) LDA partition with $\alpha_d = 0.1$

Figure 7: Test accuracy of FedProx, using VGG-9 on CIFAR-10.



(a) LLT partition with $\alpha_l = 0.9$ (b) LLT partition with $\alpha_l = 0.99$ (c) LDA partition with $\alpha_d = 0.5$ (d) LDA partition with $\alpha_d = 0.1$

Figure 8: Test accuracy of FedNova with momentum acceleration, using VGG-9 on CIFAR-10.

5 Conclusion

In this paper, we conduct experiments to explore the effect of the naive data resampling technique in FL. By analyzing the preliminary experiment results, we have two key findings of data resampling in FL.

Based on these key findings, we proposed a simple but effective data resampling strategy IWDS for FL usage. It can be seamlessly and easily combined with many current FL algorithms without any extra expense or data communication. We conduct comprehensive experiments to verify the great improvement on the convergence and performance of FL training with IWDS. And we also examine that it can be seamlessly combined with other FL algorithms like FedProx and FdeNova, and get good results.

Except for our novel effective data resampling strategy, we would like to highlight the effect of data resampling of FL and our key findings here again. These two interesting phenomena offer a distinctive example for researchers to explore the mechanism in deep learning. And the reasons why more similar label sampling probability by data resampling could accelerate training convergence may inspire researchers to develop more effective methods.

Acknowledgments

This research was inspired by the Federated Learning course (COMP6211G) in HKUST.

References

[Buda *et al.*, 2018] Mateusz Buda, Atsuto Maki, and Maciej A Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106:249–259, 2018.

[Byrd and Lipton, 2019] Jonathon Byrd and Zachary Lipton. What is the effect of importance weighting in deep learning? In *International Conference on Machine Learning*, pages 872–881. PMLR, 2019.

[Cao *et al.*, 2019] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Aréchiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 1565–1576, 2019.

[Cui *et al.*, 2019] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9260–9269, 2019.

[Duan *et al.*, 2019] Moming Duan, Duo Liu, Xianzhang Chen, Yujuan Tan, Jinting Ren, Lei Qiao, and Liang Liang. Astraea: Self-balancing federated learning for improving classification accuracy of mobile deep learning applications. In *2019 IEEE 37th International Conference on Computer Design (ICCD)*, pages 246–254, 2019.

[He *et al.*, 2020] Chaoyang He, Songze Li, Jinhyun So, Mi Zhang, Hongyi Wang, Xiaoyang Wang, Praneeth Vepakomma, Abhishek Singh, Hang Qiu, Li Shen, Peilin Zhao, Yan Kang, Yang Liu, Ramesh Raskar, Qiang Yang, Murali Annavaram, and Salman Avestimehr. Fedml: A research library and benchmark for federated machine learning. *arXiv preprint arXiv:2007.13518*, 2020.

[He *et al.*, 2021] Chaoyang He, Alay Dilipbhai Shah, Zhenheng Tang, Di Fan, Adarshan Naiynar Sivashunmugam, Keerti Bhogaraju, Mita Shimpi, Li Shen, Xiaowen Chu, Mahdi Soltanolkotabi, and Salman Avestimehr. Fedcv: A federated learning framework for diverse computer vision tasks, 2021. https://fedml.ai/files/FedCV.pdf.

[Hsu *et al.*, 2019] T. Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *ArXiv*, abs/1909.06335, 2019.

[Hsu *et al.*, 2020] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Federated visual classification with real-world data distribution. In *ECCV*, pages 76–92, 2020.

[Kang *et al.*, 2020] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. In *International Conference on Learning Representations*, 2020.

[Li *et al.*, 2020a] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In *Proceedings of Machine Learning and Systems*, volume 2, pages 429–450, 2020.

[Li *et al.*, 2020b] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. In *International Conference on Learning Representations*, 2020.

[Liu *et al.*, 2019] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2537–2546, 2019.

[Liu *et al.*, 2020] Yang Liu, Anbu Huang, Yun Luo, He Huang, Youzhi Liu, Yuanyuan Chen, Lican Feng, Tianjian Chen, Han Yu, and Qiang Yang. Fedvision: An online visual object detection platform powered by federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13172–13179, 2020.

[Luo *et al.*, 2019] Jiahuan Luo, Xueyang Wu, Yun Luo, Anbu Huang, Yunfeng Huang, Yang Liu, and Qiang Yang. Real-world image datasets for federated learning. *arXiv preprint arXiv:1910.11089*, 2019.

[McMahan *et al.*, 2017] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282, 2017.

[Reddi *et al.*, 2021] Sashank J. Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and Hugh Brendan McMahan. Adaptive federated optimization. In *International Conference on Learning Representations*, 2021.

[Van Horn *et al.*, 2018] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8769–8778, 2018.

[Wang *et al.*, 2020a] Hao Wang, Zakhary Kaplan, Di Niu, and Baochun Li. Optimizing federated learning on non-iid data with reinforcement learning. In *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, pages 1698–1707, 2020.

[Wang *et al.*, 2020b] Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H. Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. In *Advances in Neural Information Processing Systems*, volume 33, pages 7611–7623, 2020.

[Yang *et al.*, 2018] Timothy Yang, Galen Andrew, Hubert Eichner, Haicheng Sun, Wei Li, Nicholas Kong, Daniel Ramage, and Françoise Beaufays. Applied federated learning: Improving google keyboard query suggestions, 2018.