

# Federated Learning for distribution skewed data using sample weights

Hung Nguyen<sup>\*</sup>, Peiyuan Wu<sup>†</sup> and J. Morris Chang<sup>‡</sup>

<sup>\*‡</sup> Electrical Engineering Dept, University of South Florida, USA

<sup>†</sup> Electrical Engineering Dept, National Taiwan University, Taiwan

Email: <sup>\*</sup>nsh@usf.edu

**Abstract**— One of the most challenging issues in federated learning is that the data is often not independent and identically distributed (non-IID). Clients are expected to contribute the same type of data and drawn from one global distribution. However, data are often collected in different ways from different resources. Thus, the data distributions among clients might be different from the underlying global distribution. This creates a weight divergence issue and reduces federated learning performance. This work focuses on improving federated learning performance for skewed data distribution across clients. The main idea is to adjust the client distribution closer to the global distribution using sample weights. Thus, the machine learning model converges faster with higher accuracy. We start from the fundamental concept of empirical risk minimization and theoretically derive a solution for adjusting the distribution skewness using sample weights. To determine sample weights, we implicitly exchange density information by leveraging a neural network-based density estimation model, MADE. The clients' data distribution can then be adjusted without exposing their raw data. Our experiment results on three real-world datasets show that the proposed method not only improves federated learning accuracy but also significantly reduces communication costs compared to the other experimental methods.

**Impact Statement**—Non-IID issue is a well-known problem in machine learning applications, especially for Federated Learning, as clients often collect data from different sources and in different conditions. The problem significantly reduces machine learning performance and increases communication costs. To alleviate the negative impact of non-IID data, several works have been proposed. However, they mostly require clients to share part of their private data or rely on the global information from the global model, which already suffered the non-IID. In this work, the proposed method adjusts the distribution via sample weight in the loss function during training. We only ask clients to share some extra models. Similar to a typical FL framework, clients are not required to expose their raw data. The results on three real-world datasets showed that the proposed method is much more efficient than other experimental methods. It increases the accuracy of ML model and significantly reduces communication costs e.i., up to eight times for real non-IID dataset FEMNIST.

**Index Terms**—feature skewness, communication cost reduction, privacy preservation, deep learning.

## I. INTRODUCTION

Since the demand for massive data in artificial intelligent machines, federated learning (FL) was first introduced in 2016, [1], a collaboratively decentralized learning framework, in contrast to centralized learning approaches (in which datasets are sent to an aggregator), FL encourages data holders to

contribute without the privacy concern of exposing their raw data. For example, a number of hospitals holding patient records would participate in a machine learning (ML) system to provide better disease predictions via a FL framework without the concern of privacy disclosure. An FL framework is then set up to train a global machine learning model with the participation of all hospitals (clients) and an aggregator coordinating the model transfer between clients and the aggregator. Instead of sharing raw data, each client trains their ML model for a small number of iterations and updates the model parameters to the aggregator. The aggregator then merges all clients' model parameters into a new global model and sends them back to the clients for further training iterations. Clients receive the global model and continue to train for the next iterations. The process is repeated until the global model is fully trained.

Because the data in such collaborative learning usually come from different sources, they might be drawn from local distributions that are different from the underlying global distribution. This can be considered a non-IID issue, and it might cause a performance reduction. The primary problem is the divergence of model weights, as found in [2] by Zhao et al. The authors showed that the model's weights tend to be more diverged for non-IID data compared to that for IID data. This causes a performance reduction, and it worsens as the data distribution becomes more skewed. For example, the accuracy dropped by about 10% for image dataset Cifar-10 [3], and speech recognition dataset KWS [4] with a non-IID setting. In FL, this issue arises due to differences between client individual distribution and the global distribution. To address this problem, we proposed an algorithm that utilizes sample weights to adjust individual client distributions closer to the global distribution during the training process. However, obtaining global information across clients is challenging in an FL setting because clients need to allow the exposure of their raw data. To overcome this challenge, the proposed method implicitly shares statistical information of client data without revealing the client's raw data. The method only requires clients to exchange additional model weights using a typical FL procedure. Once the adjustment weights are acquired, the machine learning model can be trained using a standard FL framework. The proposed method is demonstrated to improve FL accuracy and significantly reduce FL communication costs through experiments on three real-world datasets.

Our contributions are as follows:

Code and Data are available at <https://github.com/nsh135/FedDiskPytorch>

- 1) Provide a theoretical base for skewed feature distribution data for federated learning by adjusting sample weights derived from the machine learning empirical risk.
- 2) Provide a practical solution to mitigate the problem of learning from non-IID data for the FL framework without sharing clients' draw data. The proposed method only requires clients to share additional model parameters, similar to a typical federated learning framework.
- 3) Several experiments were conducted on three datasets, including MNIST, non-IID benchmark dataset FEM-NIST and real-world dataset Chest-Xray. The results demonstrate that the proposed method outperforms other experimental methods in classification accuracy and dramatically reduces the communication cost.
- 4) As the proposed method needs to exchange additional information, we also provide a theoretical analysis to analyze the potential privacy leakage. We showed that the leakage information becomes insignificant when the number of clients increases.
- 5) To our best knowledge, the proposed method is the first method utilizing data distribution information and sample weights to tackle the FL Non-IID issue.

The rest of this paper is organized as follows. A brief review was conducted in Section II Related Work. Section III formulates our problem and our goal to achieve. Section IV introduces a neural network-based model that is leveraged in our work to carry density information. Our proposed solution is introduced in Section V. A comprehensive study on the modules in the proposed method is introduced in Section VI Ablation Study. We provide a privacy leakage analysis in Section VIII as the proposed method indirectly exchanges distribution information. Section VII shows our experimental results and illustrates the proposed method's performance. Section IX summarizes our study and discusses future work to improve the proposed method.

## II. RELATED WORKS

To learn a model utilizing data from multiple clients without directly accessing clients' data, authors in [1] first introduced Federated Averaging (FedAvg) and demonstrated its robustness in 2016. The main idea is that clients (data holders) are involved in a model training process by exchanging local models' parameters instead of exchanging raw data. Since then, FL has been seen in various applications in different fields [5], [6], [7], [8], [9], [10]. Federated Learning can be categorized into two primary scenarios: cross-silo and cross-device FL. In the former, there are fewer clients, each having substantial data (such as hospitals). In contrast, the latter scenario involves numerous clients with lightweight devices, and each client may have a smaller dataset (like mobile users). One of the main concerns in FL is that the data might come from different sources and have different distributions. Thus, FL performance is significantly reduced because this violates a fundamental machine learning assumption that data should be independent and identically distributed (IID). The FL over non-IID data has been shown in existing works [11], [12], [13], [14], [15], [16], [17], [18], [39] that its performance deteriorates dramatically.

In this work, we focus on tackling the non-IID data issue in a federated learning system in which the collected data feature distribution is skewed. Many different reasons might cause the skewness. For example, clients might perform different sampling methods, apply different normalization methods, or sample using different devices.

Over the past few years, there have been a number of approaches aiming at reducing non-IID data impacts. While many current works focus on skewed label distribution, there are only limited approaches considering skewed feature distribution data, which is very common in various fields, e.g., medical images collected from different X-ray machines. We classify the existing works into three categories as follows.

- **Sharing data:** This approach mainly focuses on adjusting model weights or calibrating model parameters using sharing data. However, they require a certain amount of raw data to be shared among users. For example, authors in [2], [19] proposed an alleviation by finetuning the model on globally shared data to adjust the distribution drift. Thus, this still poses a privacy concern. Zhu et al. in [20] used adversarial training and shared the generator for generating synthetic samples, which might contain global information. However, training generators also suffered from the non-IID itself.
- **Training stabilization:** This approach focuses on stabilizing the local training process by regulating the deviation between local parameters and global parameters. This could be implemented by normalizing layers, adding regularization terms in loss functions, or sharing specific layers. For example, Li et al. illustrated in their work (FedBN) [21] that Local Batch Normalization would help to reduce the problem of non-IID data. FedBN suggests clients not synchronize local batch normalization parameters with the global model. Sahu et al. introduce FedProx [22] to solve the weight-divergence issue by proposing a loss function that constrains the local models to stay close to the global model. FedNova [23] suggests to normalize local weights before synchronizing with the aggregator. FedMA [24], AFL [25], and PFNM [26] consider combinations of layer-wise parameters and provide an aggregation of such parameters to alleviate the non-IID issue. In FedRod [27], Chen and Chao deal with the non-IID issue by learning hyper-networks locally, which results in personalized classifiers for clients and clients' class distributions. Recently, Tan et al. [28] tackled the non-IID data issue by exchanging representation vectors of samples in a given class instead of the model's parameters, enabling clients to have personalized model architecture.
- **Weighted aggregation:** The approach mitigates heterogeneity by adjusting the model weight during aggregation. However, this works on the model level, so it is ineffective if the data is non-IID within a client. FedCL [29] and FedDNA [30] share statistical parameters of models (means and variances) and aim at finding averaging weights for each client's model to minimize models' weights divergence across clients. However, as

this approach only considers the aggregating weights for each model, the improvement is minor.

### III. PROBLEM STATEMENT

In this section, we introduce and formulate the scenario of FL with skewed feature distribution across clients. Our scenario is a learning collaboration between  $K$  clients to build a global classification model that maximizes the global accuracy given arbitrary data. Each client holds a number of individual records that they are not willing to share with others due to privacy concerns. This study focuses on preventing the performance of the global model from deteriorating because of the distribution skewness issue [31] across clients.

Our goal is to adjust the clients' distributions to be closer to the global distribution via sample weights. We denote the data and associated labels held by client  $k \in \{1, \dots, K\}$  as  $\{(\mathbf{x}_k^j, y_k^j)\}_{j=1}^{N_k}$  where  $\mathbf{x}_k^j \in \mathbb{R}^d$  and  $y_k^j \in \mathbb{N}$ . Let the data distribution of the  $k^{th}$  client be  $q_k(\mathbf{x})$  and the ground truth global distribution is  $p(\mathbf{x})$ . Our problem becomes finding an adjusting weight function for each client  $k$ ,  $\alpha(\mathbf{x}_k)$  such that

$$\alpha_k(\mathbf{x})q_k(\mathbf{x}) = p(\mathbf{x}) \quad (1)$$

### IV. PRELIMINARY: MASKED AUTOENCODER FOR DISTRIBUTION ESTIMATION (MADE)

The proposed method asks the clients to share additional model weights that carry their local datasets' distribution information instead of sharing the raw data. We utilize a neural network-based density estimation, namely, Masked Autoencoder for Distribution Estimation (MADE) [32]. This section briefly introduces MADE.

MADE is designed to estimate the probability distribution of input components (e.g., pixels in an image). MADE assumes input components are dependent instead of independent, which is relevant in many applications. For example, MADE can decompose the distribution of an instance  $\mathbf{x}$  consisting  $n$  components  $x_1, x_2, x_3, \dots, x_n$  as follows:

$$p(\mathbf{x}) = p(x_1|x_2, x_3, \dots, x_n) \cdot p(x_2|x_3, \dots, x_n) \dots p(x_{n-1}|x_n) \cdot p(x_n). \quad (2)$$

In our study, the instances are images and each pixel can be considered as a component. Thus,  $n$  is the size of a flatten image vector.

For MADE implementation, a shallow neural network is utilized. Its input and output size are equal (similar to an Autoencoder), for example a size of  $n$  for the above example. The main idea is to mimic Equation 2 by masking neuron connections across layers to control the seen and unseen connections to model output. Specifically, MADE poses constraints on the model that each output component in a certain layer only connects to its dependent input components in the previous layer. Masks are created based on such principle, and applied to the weights of the model.

Specifically, MADE assigns each unit in a hidden layer an integer  $m$  between 1 and  $D - 1$ , where  $D$  is the number of dimensions. Denote  $m(k)$  as the maximum number of units in

the previous layer to which the  $k^{th}$  hidden unit can connect, the weight mask  $M$  is then formulated as follows:  $M_{k,d} = 1_{m(k) \geq d} = \begin{cases} 1 & \text{if } m(k) \geq d \\ 0 & \text{otherwise,} \end{cases}$  for  $d \in \{1, \dots, D\}$  and  $k \in \{1, \dots, K\}$  with  $K$  being the number of hidden layer units.

### V. FEDERATED LEARNING FOR DISTRIBUTION SKEWED DATA USING SAMPLE WEIGHTS

In this section, a solution is proposed to alleviate the negative impact of distribution skewness across clients for federated learning by adjusting client data distribution during the training process. The proposed method aims to find weights for training samples in order to adjust client data distributions. The remainder of this section introduces how we design sample weights. We also show that the goal in Equation 1 can be derived from the machine learning optimization problem as described in this section.

By applying the sample weights for the local training on each client, the proposed method reduces the distribution skewness of each client's data and prevents clients' raw data from being exposed. Some statistical information between clients and the aggregator must be exchanged to find sample weights. However, instead of exchanging the raw information, which might hurt clients' privacy, the proposed method only exchanges model parameters, similar to a typical FL framework.

Our framework is illustrated in Figure 1. Where  $f(\cdot)$  denotes local inference models and  $\mathbf{w}_k$  is the model's parameter of the  $k^{th}$  client. The proposed method, namely FedDisk, requires a 2-phase process. First, clients jointly learn a global density estimation model and their local density models utilizing MADE models. These models are then used to derive sample weights for the local training process. Second, the machine learning tasks can be learned by the conventional FL procedure, with the data skewness issue mitigated by the sample weights from the first phase.

#### A. Sample Weights Design

As we do not have sufficient information about the true distribution, we consider the combination of all clients' dataset distribution as our true distribution. Thus, we consider the probability density function (pdf) of the true distribution as

$$p(\mathbf{x}) = \sum_{k=1}^K c_k q_k(\mathbf{x}), \quad (3)$$

where  $q_k(\mathbf{x})$  denotes the pdf of the  $k^{th}$  client's data.  $c_k$  depicts the client's weight, as determined by the ratio of the client's sample count  $N_k$  to the total number of samples  $N$ :

$$c_k = \frac{N_k}{N}. \quad (4)$$

To jointly learn a global model, the system finds the expectation of the loss function  $l(g(\mathbf{x}), y)$  with sample  $\mathbf{x}$  that drawn from the true distribution. The expected loss is formulated by the associated risk [33] as follows:

$$\mathbb{E}[l(g(\mathbf{x}), y)] = \iint l(g(\mathbf{x}), y) p(y|\mathbf{x}) p(\mathbf{x}) d\mathbf{x} dy, \quad (5)$$

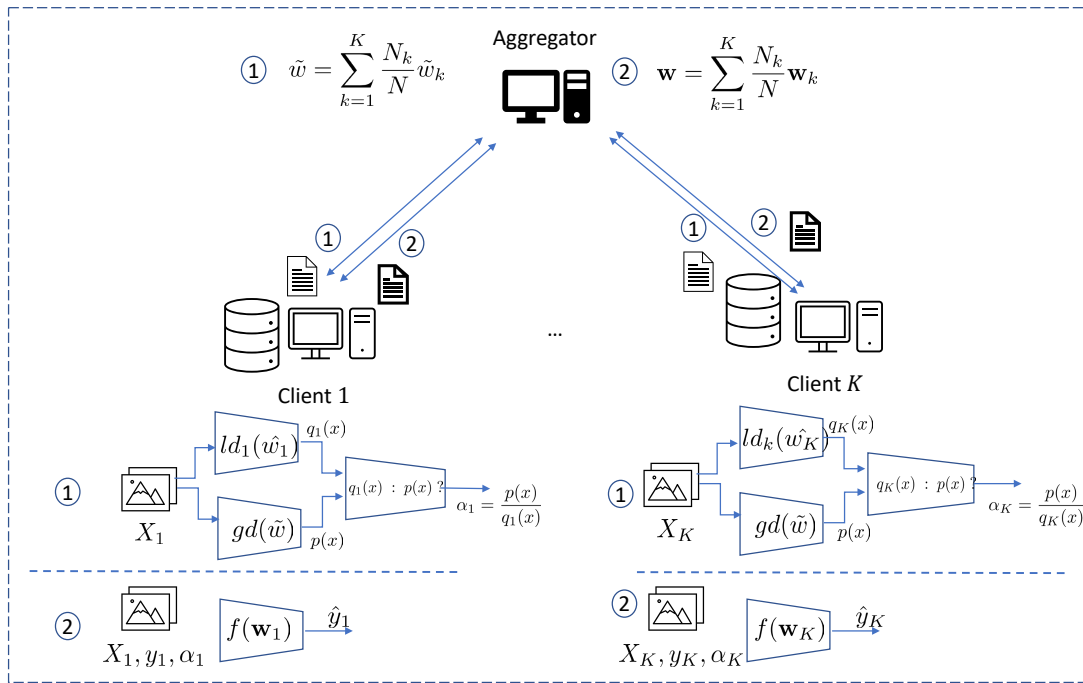


Figure 1: FedDisk Framework: The proposed framework has two phases. First, local and global probability density functions ( $p(x), q(x)$ ) are estimated via MADE models leveraging FL procedures. Then, the sample weights  $\alpha$  are computed by approximating density ratio via class probability estimation. Second, the machine learning tasks (e.g., classification) can be performed similar to a typical FL method (i.e., FedAvg) with the sample weights acquired from phase 1.

where  $p(\mathbf{x}, y)$  is the joint pdf of a sample  $\mathbf{x}$  and its associated label  $y$ , and  $p(y|\mathbf{x})$  is the conditional probability of a label  $y$  given a sample  $\mathbf{x}$ . We also assume that for any client  $k \in \{1, \dots, K\}$  with local data distribution  $q_k(\mathbf{x})$ , the conditional probability of a label  $y$  given a sample  $\mathbf{x}$  is equivalent to that of the true distribution, namely

$$q_k(y|\mathbf{x}) = p(y|\mathbf{x}). \quad (6)$$

From Equation 3, 5, 6, and by multiplying with factor  $\frac{q_k(\mathbf{x})}{q_k(\mathbf{x})} = 1$ , the expected loss in Equation 5 can be expanded as follows:

$$\mathbb{E}[l(g(\mathbf{x}), y)] = \iint l(g(\mathbf{x}), y) p(y|\mathbf{x}) p(\mathbf{x}) d\mathbf{x} dy, \quad (7)$$

$$= \iint l(g(\mathbf{x}), y) q_k(y|\mathbf{x}) \frac{q_k(\mathbf{x})}{q_k(\mathbf{x})} p(\mathbf{x}) d\mathbf{x} dy \quad (8)$$

$$= \iint l(g(\mathbf{x}), y) q_k(\mathbf{x}, y) \frac{p(\mathbf{x})}{q_k(\mathbf{x})} d\mathbf{x} dy. \quad (9)$$

The objective of the global model thus amounts to minimize the empirical risk over all  $K$  clients' datasets:

$$\underset{g}{\text{minimize}} \frac{1}{N} \sum_{k=1}^K \sum_{j=1}^{N_k} \alpha_k^j l(g(\mathbf{x}_k^j), y_k^j), \quad (10)$$

where  $\mathbf{x}_k^j, y_k^j$  are the  $j^{\text{th}}$  sample and its label.  $N$  is the total number of samples.  $\alpha_k^j$  denotes  $\alpha_k(\mathbf{x}^j)$ , which represents client  $k$  sample weight function with respect to  $\mathbf{x}$ , computed as

$$\alpha_k^j = \frac{p(\mathbf{x})}{q_k(\mathbf{x})} = \frac{\sum_{i=1}^K q_i(\mathbf{x})}{q_k(\mathbf{x})}. \quad (11)$$

Our problem becomes minimizing the summation of the loss functions (Equation 10) over all clients. For each client, the loss function is minimized over local samples with the corresponding  $j^{\text{th}}$  sample weight of the client  $k^{\text{th}}$ ,  $\alpha_k^j$ . The sample weights could be estimated by the density ratio between the true distribution (global distribution) and the client distributions (local distributions). For each client, the local distribution can be estimated using its local data. However, the challenge is to achieve the true distribution without having access to other clients' data. To solve this, we leverage a neural network-based density estimation model to learn the global density function via a typical federated learning procedure. Thus, clients can implicitly exchange some statistical information, while still preserving the privacy in client data.

### B. Probability Density Approximation

To estimate global density and preserve client privacy at the same time, we propose to leverage a neural network-based density estimation so that we can exchange local density information (via models' weights) with the aggregator without sharing the raw data. In this work, we leverage a well-known method, namely, Masked Autoencoder for Distribution Estimation (MADE, [32]), which is briefly reviewed in Section IV. Elaborately, each client aims to estimate its local probability density  $q_k(\mathbf{x})$  using its own dataset, and all  $K$  clients jointly estimate the global probability density  $p(\mathbf{x}) = q_1(\mathbf{x}) + \dots + q_K(\mathbf{x})$ . Learned MADE models are used to approximate local probability density functions, and the global MADE model approximates the global probability density. The learning process is described as follows.

The  $k^{th}$  client learns a local density estimation model  $ld_k(\hat{w}_k)$  (where  $ld(\cdot)$  approximates density estimation function with parameter  $\hat{w}$ ) using its local data. It then jointly learns a global density estimation model  $gd(\tilde{w})$  (where  $gd(\cdot)$  represents the global density function with the parameter  $\tilde{w}$ ) using the procedure similarly to FedAvg [1]. Specifically, for the local model density estimation models, each client train a MADE model on its local data until the loss function can not be improved. For the global density estimation model, each client trains its data locally for a certain number of iterations, and then model parameters are sent to an aggregator for the aggregation. Since clients might own different number of samples, a weight of  $N_k/N$  (where  $N_k$  and  $N$  are the number of samples of the  $k^{th}$  client and the the total number of samples over all clients) is used for adjusting client parameter significance, similar to FedAvg. After aggregating all clients' model parameters, the aggregator shares global model parameters to all clients. The steps are repeated until the validation loss starts increasing. The global MADE model aggregation from  $K$  clients at iteration  $t$  can be described as follows:

$$\tilde{w}^t = \sum_{k=1}^K \frac{N_k}{N} \tilde{w}_k^t \quad (12)$$

### C. Sample Weight Approximation

After the local and global density approximations by MADE models are fully learned (Section V-B), we can estimate sample weights in Equation 11. Since MADE models output vectors of conditional probabilities for each element in the  $d$ -dimensional input  $\mathbf{x}$ , an intuitive way to compute  $p(\mathbf{x})$  is to multiply all the conditional probabilities. However, as  $p(\mathbf{x})$  vanishes when any of the conditional probabilities vanishes, we instead keep the output as a vector of conditional probabilities (same size as input) and approximate the density ratio in Equation 11 using a class probability estimation method inspired by [34]. The method aims at training a binary classifier to output a probability that represents the ratio between  $p(\mathbf{x})$  and  $q(\mathbf{x})$ . The solution detail is described in the rest of this subsection.

After each client receives the final global MADE model and trains its own local MADE, it starts to evaluate sample weights for its local data. The training data of the  $k^{th}$  client,  $X_k$ , is then fed into both the global MADE (the global MADE is downloaded to clients so that this step can be done locally) and the local MADE to estimate  $p(\mathbf{x})$  and  $q_k(\mathbf{x})$ , respectively. Denote  $\mathbf{u}$  as the output vector of density estimation models, and  $l$  be the pseudo label indicating whether it is sampled from the global destination ( $l = 1$ ) or the local distribution ( $l = 0$ ). Each client then trains a binary classifier to differentiate whether the output  $\mathbf{u}$  comes from  $p(\mathbf{x})$  or  $q_k(\mathbf{x})$ . Outputs of the two MADE models (the sample size of each output is  $N_k$ ) are concatenated to a new vector dataset including samples  $\{(\mathbf{u}_k^i, l_k^i)\}_{i=1}^{2N_k}$ , and is used to train the binary classifier. The conditional probabilities of the binary classification model  $h(\mathbf{u}, w_h)$  (where  $\mathbf{u}$  is the input variable,  $w_h$  is the model parameter) can be approximated as following:

$$\mathcal{P}(\mathbf{u}|l=0) \propto q_k(\mathbf{x}), \quad \mathcal{P}(\mathbf{u}|l=1) \propto p(\mathbf{x}). \quad (13)$$

From Bayes' rule, we have

$$\begin{aligned} \frac{p(\mathbf{x})}{q(\mathbf{x})} &= \frac{\mathcal{P}(\mathbf{u}|l=1)}{\mathcal{P}(\mathbf{u}|l=0)} = \left( \frac{\mathcal{P}(l=1|\mathbf{u})\mathcal{P}(\mathbf{u})}{\mathcal{P}(l=1)} \right) \left( \frac{\mathcal{P}(l=0)}{\mathcal{P}(l=0|\mathbf{u})\mathcal{P}(\mathbf{u})} \right) \\ &= \frac{\mathcal{P}(l=1|\mathbf{u})\mathcal{P}(l=0)}{\mathcal{P}(l=0|\mathbf{u})\mathcal{P}(l=1)}. \end{aligned} \quad (14)$$

We approximate the marginal probability ratio between two distributions ( $\mathcal{P}(l=0)$  and  $\mathcal{P}(l=1)$ ) by the number of samples from the two distributions  $N_k$  over the concatenated dataset size ( $2N_k$ ). Thus, We have  $\frac{\mathcal{P}(l=0)}{\mathcal{P}(l=1)} = \frac{N_k}{2N_k} \frac{2N_k}{N_k} = 1$ .

The density ratio then can be estimated as follows:

$$\frac{p(\mathbf{x})}{q(\mathbf{x})} = \frac{\mathcal{P}(l=1|\mathbf{u})}{\mathcal{P}(l=0|\mathbf{u})} = \frac{\mathcal{P}(l=1|\mathbf{u})}{1 - \mathcal{P}(l=1|\mathbf{u})}, \quad (16)$$

where  $\mathcal{P}(l=1|\mathbf{u})$  is the classifier's probability-liked output indicating how likely an input vector  $\mathbf{u}$  comes from the global probability  $p(\mathbf{x})$ .

To summarize, the  $j^{th}$  training sample of client  $k$ ,  $\mathbf{x}_k^j$ , is fed into the client's local MADE model to achieve its corresponding density estimation  $\mathbf{u}_k^j$ .  $\mathbf{u}_k^j$  is then fed into the binary classification function  $h(\mathbf{u})$  to achieve the class probability  $\mathcal{P}(l=1|\mathbf{u}_k^j)$ . This is used to estimate the sample weight  $\alpha_k^j$  (Equation 11) based on Equation 16. In words, the binary classification model  $h(\mathbf{u}, \bar{w})$  is expected to return higher weights for samples that are likely belonging to the true distribution and vice versa.

### D. Learning With Skewed Distribution Data Across clients

After acquiring sample weights, each client starts to train the model on the local dataset and corresponding sample weights for a machine learning task (e.g., classification) as a typical FL framework. In this work, we follow the procedure introduced by FedAvg to learn the global model. The aggregator aggregates clients' local models as follows:

$$\mathbf{w}^t = \sum_{k=1}^K \frac{N_k}{N} \mathbf{w}_k^t \quad (17)$$

where  $\mathbf{w}^t$  and  $\mathbf{w}_k^t$  are the global and local model parameter of  $k^{th}$  client at the  $t^{th}$  iteration.

### E. Algorithm

Our main strategy can be described in Algorithm 1. We only concentrated on describing the first phase of FedDisk as the second is the same as a typical FL framework. First, each client trains its own local MADE model on the local data to obtain local distribution information (lines 1 – 3). All clients then jointly train global MADE utilizing a typical FL framework (lines 4 – 14). After achieving these two models, data are sampled from the two output models to acquire data samples containing local and global information (lines 15 – 17). These samples are concatenated with the pseudo label of 0 for samples that come from local distribution and 1 for ones from the global distribution (line 18). They are then used for training an adversarial binary classifier (line 19). The purpose is to differentiate the two datasets sampled from the two distributions. The samples that are similar to the global samples will return a higher probability of belonging to class 1 (come from global distribution) and vice versa. Thus, the classifier probability-like output that represents class 1 is then used to be the weights for the sample (line 20).

## VI. ABLATION STUDY: FEDDISKAB

In this section, we examine the idea of using sample weight for non-IID data and the FedDisk sample weight effectiveness by looking at the case when the weights are derived directly from the raw data. Specifically, instead of learning sample weights from the local and global MADE model output, the weights are learned directly

---

**Algorithm 1** Phase 1: Sample Weight Computing.

---

**Input:** Client  $k^{th}$ : Dataset  $\{X_k, y_k\}$ .

**Parameter:**  $K$  : Number of client.

$N$  : Number of total samples.

$N_k$ : Number of sample of client  $k^{th}$ .  $\sum_{k=1}^K N_k = N$

$local\_iter$ : Number of iterations for training local model

$global\_iter$  : Number of iterations for training global model.

$ld(\tilde{w})$  : Local MADE model containing **local distribution** information

$gd(\tilde{w})$  : Global MADE model containing **global distribution** information

$h(\tilde{w}_k)$ : Shallow Binary Classifier to differentiate output from

$p_x$  and  $q_x$   $\alpha_k$ : sample weights for client  $k$  data

**Output:**  $\alpha_k$ : Sample weights for  $X_k$

{Training local MADE model}

1: **for**  $k \leftarrow 1$  to  $K$  **do**

• Client  $k$ :

2: Fully train  $ld_k(\tilde{w}_k)$  on local data  $\{X_k, y_k\}$ .

3: **end for**

{Training global MADE model}

4: **for**  $i \leftarrow 1$  to  $global\_iter$  **do**

5: **for**  $k \leftarrow 1$  to  $K$  **do**

• Client  $k$ :

6: Update  $\tilde{w}^{i-1}$  from the Aggregator

7: **for**  $j \leftarrow 1$  to  $local\_iter$  **do**

8: Train  $gd(\tilde{w})$  on  $\{X_k, y_k\}$ .

9: **end for**

10: Sending  $gd(\tilde{w})$  to Aggregator

11: **end for**

• Aggregator:

12: Aggregate  $\tilde{w}^i = \sum_{k=1}^K \frac{N_k}{N} \tilde{w}_k^i$

13: Broadcasting  $\tilde{w}^i$  to clients

14: **end for**

{Training shallow binary classifier}

15: **for**  $k \leftarrow 1$  to  $K$  **do**

Sample data from local distribution:

16:  $X_k^{local} \leftarrow ld(X_k | \tilde{w}_k)$ ,  $y_k^{local} \leftarrow [0...0]$

Sample data from global distribution:

17:  $X_k^{glob} \leftarrow gd(X_k | \tilde{w}_k)$ ,  $y_k^{global} \leftarrow [1...1]$

18:  $X'_k \leftarrow concat(X_k^{local}, X_k^{global})$ ,

$y'_k \leftarrow concat(y_k^{local}, y_k^{global})$

• Client  $k$ :

19: Fully train  $h(\tilde{w}_k)$  on local data  $\{X'_k, y'_k\}$ .

{Estimate sample weight}

20:  $\alpha_k \leftarrow h(X'_k | \tilde{w})[1]$

21: **end for**

---

from the raw local and global data. To obtain the global data, we combine all client's data and randomly sample the same number of the client dataset size to avoid data imbalance. This setting variant of FedDisk (namely FedDiskAb) is an ideal case for a sample weight-based approach as it assumes to have access to the raw data. To obtain sample weights, FedDiskAb only needs to train the binary

classifier on the combination of local data and global data, aiming to discriminate the two datasets. The classifier's output is used to derive the sample weight, similar to FedDisk.

Several experiments have been conducted for FedDiskAb and other methods in Section VII. The outcome demonstrates that both FedDiskAb and FedDisk surpass the performance of all alternative methods. This confirms the effectiveness of the sample weight-based strategy, whether acquired through learning the distribution from MADE models or directly from the data, in enhancing federated learning when faced with non-IID challenges. Furthermore, the performance of FedDisk closely aligns with that of FedDiskAb. This shows the fact that the local and global MADE models contribute significantly to the framework's ability to capture essential distribution information, much akin to the process of direct learning from the raw data. The details of the experimental results will be shown in Chapter VII.

## VII. EXPERIMENTS

In this section, we conduct several experiments to evaluate the proposed method on non-IID FL scenarios with three real image datasets (MNIST, Chest-Xray and FEMNIST). Our FL system goal is to learn a global classifier leveraging data from all clients. The classification accuracy is used as a metric to evaluate the performance of the proposed method. The communication cost is evaluated by counting the number of iterations needed for clients to exchange model parameters with the aggregator. We compare our method with other state-of-the-art methods, e.i., FedAvg, FedProx, FedBN, FedROD and FedPCL.

### A. Datasets & non-IID setting.

In this Section, we describe how datasets are used in our experiments. We categorize our dataset into two groups, simulated non-IID dataset (MNIST) and real non-IID datasets (Femnist & Chest-Xray). The first one contains images that have already been combined together so that our partitioning process is considered for sampling from the same contribution. Thus, we added different levels of noise to each client to simulate the feature skewness as inspired by settings in [17], and [28]. The second group's data are collected from different sources so that they are considered to be non-IID by nature. All the data are normalized and clipped to the range of [0,1] before training. Each client's data is split to 85% and 15% for training and testing sets, respectively. The detail of the datasets is described as follows.

1) *Simulated non-IID: MNIST dataset:* MNIST dataset [35] contains 60,000 (1x28x28) gray scale images of 10 digits (0-9). The number of unique output labels is 10 representing 10 digits. To mimic feature skewness, we split data equally into 100 partitions and add different level of noise to each client's data as inspired by the skewness simulation in [17]. The noise is drawn from Gaussian distribution with a mean of 0 and different values of standard deviations. More specifically, the  $k^{th}$  client ( $k \in [0, 99]$ ) is added noise with the variance of  $k * x / 100$  where  $x$  is the added noise variance.

2) *Real non-IID: Femnist dataset:* FEMNIST dataset is downloaded from <https://leaf.cmu.edu/>, which is considered a benchmark dataset for real non-IID data. It contains handwritten images of 62 digits and characters (corresponding to 62 unique labels) from different writers and strokes. In this study, we randomly select 100 different writers (each of them owns more than 300 images to avoid overfitting) and assign their data to 100 clients. The average sample size of clients is 387.47, and the standard deviation is 83.04. All images are resized to a (32x32) grayscale and normalized to the range of [0,1] before inputting to models.

3) *Real non-IID: Chest Xray dataset:* The Chest-Xray dataset, which contains pneumonia and normal chest Chest-Xray images, are collected from different sources (i.e., COVID-19 [36], Shenzhen Hospital [37], and University of California San Diego (UCSD) [38]) with different image sizes, colors and potentially taken from different medical devices. Thus, we consider this dataset non-IID by nature.

After partitioning the data into 100 clients, the mean and standard deviation of the client sample size are 325.50 and 63.74, respectively. All images are converted to grayscale and resized to (32x32). There are two unique output labels (binary classification) to predict chest X-ray images are normal or abnormal.

4) *Data examples.*: Figure 2 provides several sample images from the three datasets. The MNIST dataset images have various degrees of noise. Besides, the FEMNIST dataset includes images with different writing styles from various sources. The Chest-Xray dataset comprises images with varying resolutions and light conditions. This makes them non-IID across clients.

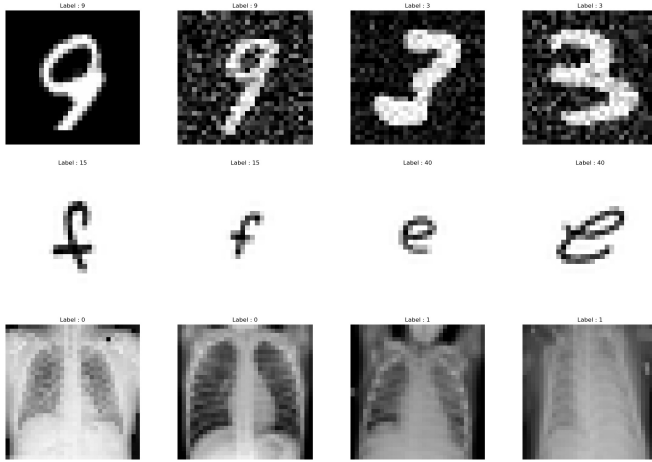


Figure 2: Example images from MNIST, FEMNIST and Chest Xray datasets. They are collected from different sources and carried a variety of resolutions, styles or conditions.

## B. Implementation Detail

1) *Baselines*: We compare our methods with different methods, i.e., FedAvg, FedProx, FedBN, FedROD and FedPCL. While most implementation details are taken from the initial parameter sets in original papers, we also tune suggested parameters and report the results that give the best values. For FedROD, the results are reported for the hyper-parameter  $\mu$  of 1. We also tried other values in the set 1, 5, 10, 20 and found that the results are very similar. For FedProx, we tuned the parameter  $\mu$  with the candidates of 0.001, 0.01, 0.1, 1 and reported the value of 0.01, 1, 0.01 for the three datasets, Chest-Xray, FEMNIST, MNIST, respectively.

2) *Federated Learning Classification Model*: We use shallow Convolutional Neural Networks (CNN) for the classification of image datasets. The models are constructed by two 5x5 convolutional layers (32 and 32 channels for Chest-Xray, 128 and 128 channels for FEMNIST, 16 and 16 channels for MNIST). Each convolutional layer is followed by 2x2 max pooling and batch normalization layers. A fully connected layer with 16 neurons is added on the top of the models. The input and output sizes are designed to fit each dataset scenario (i.e., image size and the number of unique labels). We use stochastic gradient descent (SGD) with a learning rate of 0.01 for the optimizers. Local iterations are set to 2 for all datasets. Global iterations are set to 1500 for FEMNIST and MNIST, and 500 for Chest-Xray.

3) *Density Estimation Model (MADE)*: Density estimation models (MADE) are constructed by neural networks and the hyper-parameters are taken directly from the initial setting in the original work [32]. Several experiments were conducted to select the optimal set of parameters which yield lowest loss value. The networks include input, output and one hidden layer. The number of neurons in the hidden layer is tuned from a value set of {30, 50, 100, 200, 300, 400}. The final selected number of neurons in the hidden layer are

50, 400, 30 for XRAY, FEMNIST and MNIST datasets, respectively. We noticed that using more neurons than numbers above did not significantly decrease the validation loss, thus they are the optimal settings. The model's input and output sizes are set to the flattened size of images. Specifically, the input and output size for MNIST and FEMNIST datasets is 784 with image sizes of 28x28 pixels. This number is 1024 (32x32 pixels) for Chest-Xray dataset. The maximum training iteration is set to 500, and the training process is stopped when the validation loss starts increasing. Other hyper-parameters are taken directly from [32].

4) *Sample Weight Approximation*: In order to compute the sample weight  $\alpha$ , we use a shallow, fully connected neural network to discriminate the density estimation output vectors coming from which of the two distribution density functions  $p(\mathbf{x})$  or  $q(\mathbf{x})$ . The model contains a 100-neuron hidden layer with Relu activation function. The output layer contains two neurons with Softmax activation function. All models applied a learning rate of 0.01, and SGD optimization were used in the training process. The training process is terminated if the loss function is not significantly reduced.

## C. Results

1) *Classification Accuracy*: Figure 3 shows the average of the 100 clients' testing accuracies over training iterations. The shaded regions illustrate the standard deviation over five trials. Overall, FedDisk significantly outperforms other methods in terms of classification accuracy. For example, in Figure 3a for Chest-Xray dataset, FedDisk with an accuracy of 92% outperforms others with the highest accuracy of 90.5%. For FEMNIST dataset (Figure 3b), our method achieved an accuracy of 78% while others only reached the maximum accuracy of 56% (FedROD). For MNIST, FedDisk reached the accuracy of 54.5% while others only obtained the highest accuracy of 51.7%.

Figure 4 shows the descriptive statistical accuracy results of 100 clients on different datasets. The colored rectangles contain 50% of client accuracies. The colored rectangular's lower and upper edges show the middle values in the first and second half of the sorted clients' accuracies (lower quartile and higher quartile). The middle dash is the median value. The upper and lower dashes represent the min and max clients' accuracies. Dots illustrate outliers. Overall, the bars for FedDisk are higher than others, meaning that most clients archive higher accuracy. Dots are also higher (Figure 4a and 4b) for FedDisk, showing that outlier clients are also improved. Especially, the bar for FEMNIST is significantly raised for FedDisk, indicating that the proposed method significantly improved for this dataset. It is clear that the proposed method outperforms compared methods in all experimental datasets, including real-world non-IID and simulated non-IID settings.

2) *Effective Communication Rounds*: To have a fair comparison, we use "Effective Communication Rounds" (ECR) to evaluate effective number of communication iterations for each method. On FedDisk, ECR includes the communication rounds for exchanging MADE models and classification models. Figure 5 show the aggregated training loss and validation loss for the global MADE model over communication rounds. The global MADE exchanging process stops when the validation loss starts increasing. For example, the proposed method needs 15 rounds for exchanging global MADE models in the case of the FEMNIST dataset (Figure 5b). The ECR for FedDisk in the classification phase is calculated with the number of iterations the method needs to achieve the highest value among other methods gained. Take the FEMNIST dataset experiment shown in Figure 4b for example, FedDisk only needs 105 rounds to reach FedROD's accuracy at 57% which is the highest accuracy among other experimental methods. Take 15 rounds to exchange MADE model, FedDisk only needs a total of 120 rounds to effectively reach the top comparison method accuracy. Since other methods don't need to exchange extra models, the ECRs are calculated by the number of rounds to exchange classification model until they reach their highest accuracy values.



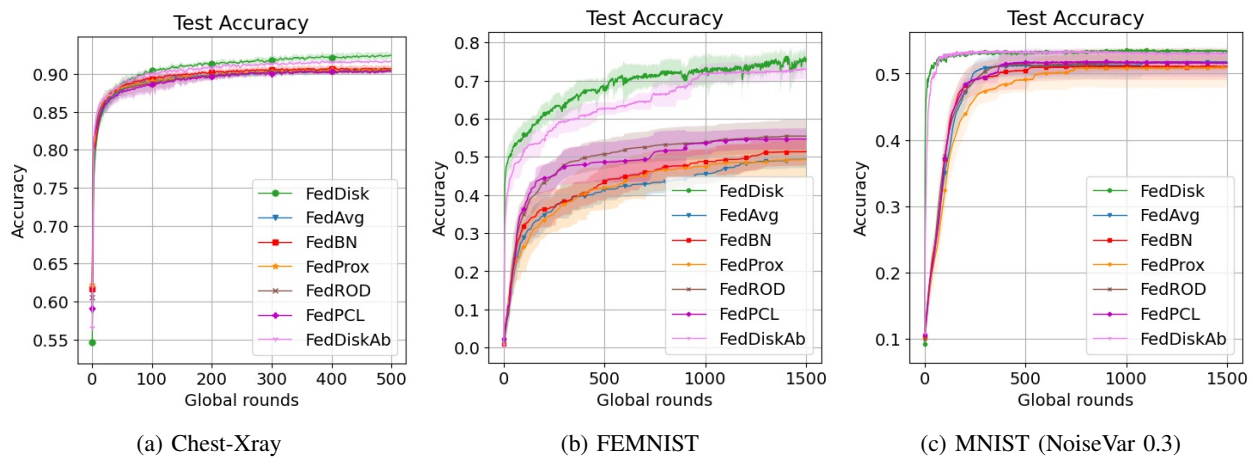


Figure 3: Global model's average test accuracy during aggregation process. For MNIST dataset, clients' data were added noise with the mean of zero and variance of 0.3

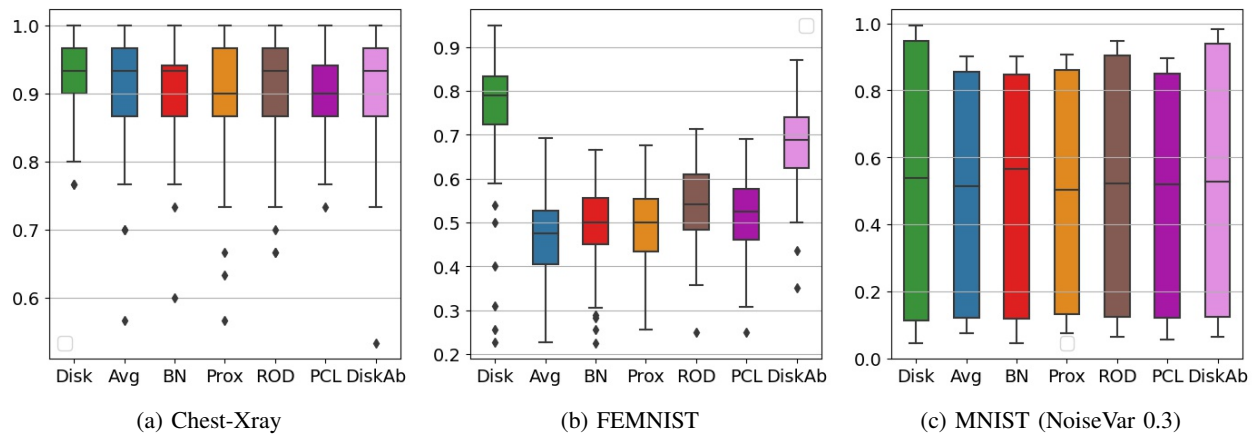


Figure 4: Test accuracy percentiles, min, max and median plot of 100 clients for different datasets and methods.

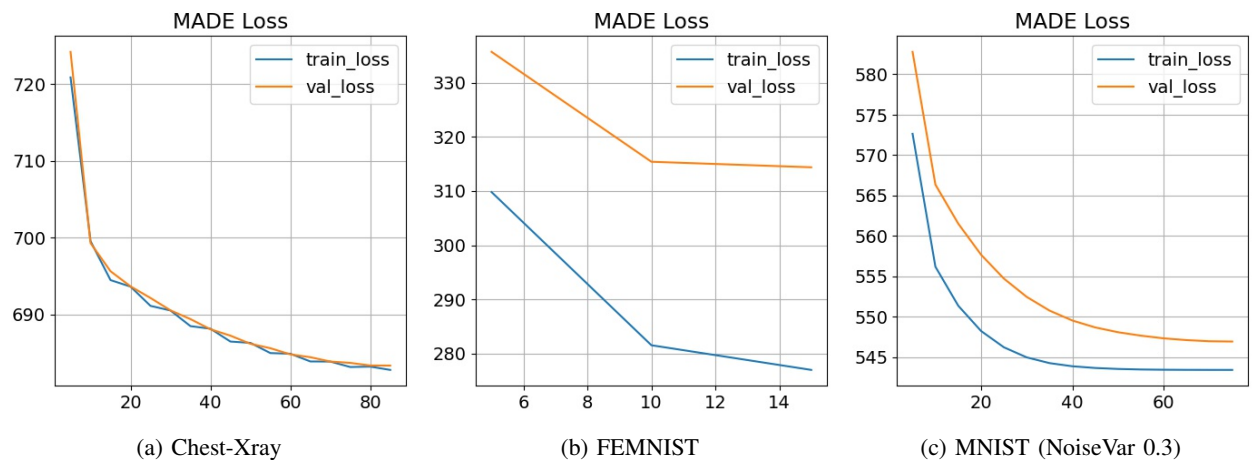


Figure 5: Average validation and train losses during training the global MADE models. The training processes were stopped if the validation loss starts increasing.

Table I shows the summary of “Effective Communication Rounds” for the three experimental datasets. FedDisk mechanism has two phases; one is to transfer MADE models, and the other is to exchange classifiers. The overall FedDisk ECR comprises the communication rounds in the two phases. As shown in the Table, FedDisk is the most effective method as it needs many fewer communication rounds to

reach the highest accuracy among other methods. This is because the proposed method only needs a few number communication rounds for the global MADE model to be converged. Besides, the weight-based adjustment converges the global classification model much faster than others. For example, the FedDisk ECR for FEMNIST is only 120 (15 for MADE model exchange plus 105 for classification model



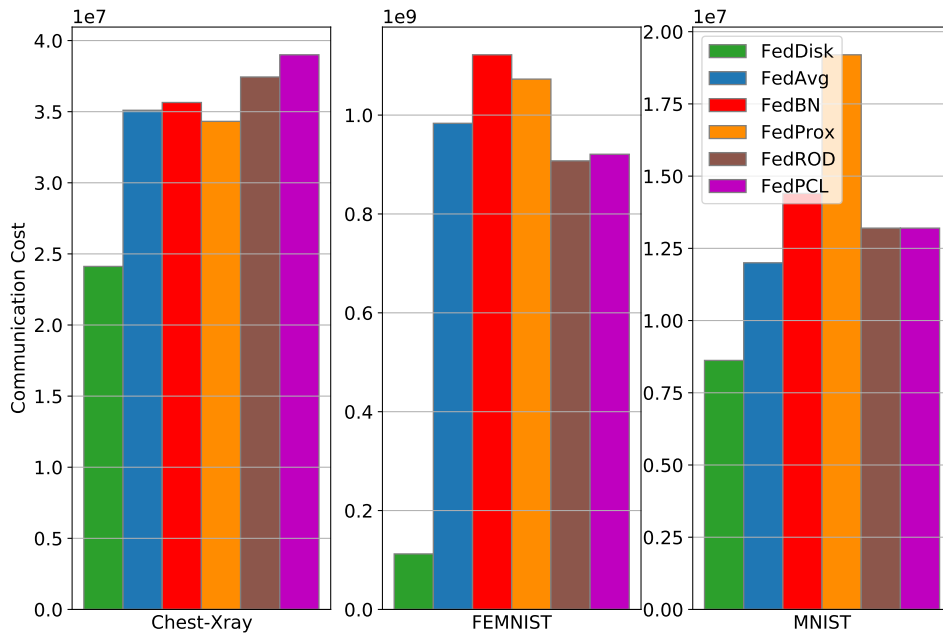


Figure 6: Summary of “Effective Communication Cost” over 3 datasets. The Figure shows that FedDisk is much more efficient in number of communication cost.

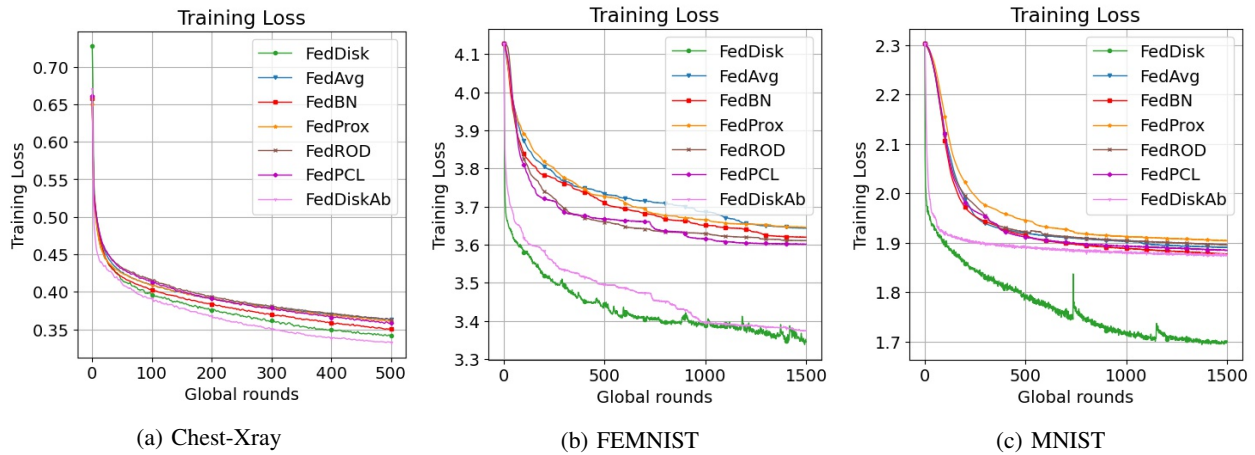


Figure 7: Global model’s losses over communication rounds on the three datasets during training. FedDisk loss reduced much faster than other methods.

Model	Chest-Xray	FEMNIST	MNIST
FedDisk MADE	80	15	70
FedDisk Classifier	100	105	85
FedDisk Total	180	120	155
FedAvg Classifier	450	1100	500
FedBN Classifier	457	1255	600
FedProx Classifier	440	1200	800
FedROD Classifier	480	1015	550
FedPCL Classifier	500	1030	550

Table I: The average of “Effective Communication Rounds” for exchanging model weights between each client and the aggregator.

Model	Chest-Xray	FEMNIST	MNIST
FedDisk MADE	102000	614000	47000
FedDisk Classifier	39000	447000	12000
FedAvg Classifier	39000	447000	12000
FedBN Classifier	39000	447000	12000
FedProx Classifier	39000	447000	12000
FedROD Classifier	39000	447000	12000
FedPCL Classifier	39000	447000	12000

Table II: Communication overhead each round per client.

exchange), whereas others take more than 1000 iterations.

3) *Effective Communication Cost*: To have a comprehensive comparison, the communication cost is estimated for each method.

The cost is comprised of communication rounds and overhead, where the overhead is measured by the size of transferred data each round between a client and the aggregator. Since the transferred data mainly contains model weights, our study uses the number of model weights to estimate the overhead size. Table II shows the number of model weights in different scenarios. Note that a client must consume two costs each communication round; one is for transferring its current

model weight, and another is for receiving the updated weight from the aggregator. Generally, the "effective communication cost" is then computed as follows.

$$C = 2 * S_{cls} * ECR_{cls}, \quad (18)$$

where  $C$  is the overall communication cost for one client,  $S_{cls}$  is the overhead (number of transferred classification model weights), and  $ECR_{cls}$  is the number of effective communication rounds. For FedDisk, as it needs to exchange extra MADE in the first phase, the computation is adjusted as follows.

$$C_{FedDisk} = 2 * (S_{MADE} * ECR_{MADE} + S_{cls} * ECR_{cls}), \quad (19)$$

where  $S_{MADE}$  and  $S_{cls}$  are model sizes needed to be transferred in phase 1 (MADE models) and phase 2 (classification models), respectively.  $ECR_{MADE}$  and  $ECR_{cls}$  are the corresponding effective communication rounds in the two phases. Note that for FedDisk,  $ECR_{MADE} + ECR_{cls} = ECR$ , and  $ECR_{cls} = ECR$  for other methods.

Figure 6 summarizes effective communication cost. Noticeably, the communication cost reduced significantly for the FEMNIST dataset under the proposed method, 8 times, from the second lowest cost method of 907,000,000 (FedROD, brown column) to 112,000,000 (FedDisk, green column). The communication cost reduction trend is also applied to other experimental datasets, Chest-Xray (1.4 times) and MNIST (1.6 times). Thus, the proposed method improves accuracy and dramatically reduces communication costs, one of the most critical concerns in Federated Learning. This is because the loss function were reduced faster as we adjusted using the sample weights. Figure 7 demonstrates the global loss values during training. For FedDisk, the sample weights effectively affect the optimization function, and the loss reduces much faster, and the accuracies proportionally increase faster.

#### D. Discussion

The proposed method offers a holistic improvement over existing federated learning methods. Its combination of enhanced accuracy and reduced communication costs signifies its effectiveness across a variety of datasets and scenarios. For example, the accuracy can be increased by 22% and the communication time reduced by 8 times for FEMNIST dataset. While FedDisk exhibits remarkable performance across various metrics in federated learning scenarios, it's important to acknowledge a drawback associated with its larger model size compared to some other methods as described in Table II. The increased model size leads to higher space occupation on client devices, which can have implications for light weight devices with limited storage capacity. Hence, the suggested approach could be well-suited for the cross-silo scenario, typically characterized by clients having ample data and adequate computational capabilities. Overall, the proposed approach provides an avenue to address critical challenges in federated learning, making it a promising option for real-world applications.

### VIII. PRIVACY LEAKAGE ANALYSIS

In this section, we discuss the privacy leakage of our method compared to the conventional FL. Similar to many other works, we utilize additional information to alleviate the negative impact of non-IID data, i.e., parameters of MADE models. However, these parameters might contain distribution information of clients' data. However, we prove that the more clients are involved in the FL training process, the less our extra information is leaked. The detail is described as follows.

Assume each client samples their own data point  $\hat{Z}_k \sim Q_k$  independently, and let  $\Theta$  be a random variable taking values in  $\llbracket 1, K \rrbracket$  with  $\mathcal{P}[\Theta = k] = \kappa_k$  and independent of  $\hat{Z}_k$  for each  $k \in \llbracket 1, K \rrbracket$ . Note that  $\hat{Z}_\Theta \sim P$ , and one may quantify the privacy leakage of

client  $k$ 's data through the knowledge of  $P$  by the mutual information between  $\hat{Z}_k$  and  $\hat{Z}_\Theta$ , as given by

$$\begin{aligned} I(\hat{Z}_k; \hat{Z}_\Theta) &\leq I(\hat{Z}_k; \hat{Z}_\Theta, \Theta) = I(\hat{Z}_k; \Theta) + I(\hat{Z}_k; \hat{Z}_\Theta | \Theta) \\ &= I(\hat{Z}_k; \hat{Z}_\Theta | \Theta) = \sum_{i=1}^K \mathcal{P}[\Theta = i] I(\hat{Z}_k; \hat{Z}_i) \\ &= \kappa_k H(\hat{Z}_k). \end{aligned} \quad (20)$$

In other words, the privacy leakage is proportional to  $\kappa_k$ , which decreases to 0 as long as  $\kappa_k = O(1/K)$  and  $K \rightarrow \infty$ .

### IX. CONCLUSION

In this work, we have proposed an FL method to tackle the issue of data distribution skewness. The technique utilizes an FL framework and a neural network-based density estimation model to derive training sample weights. This helps to adjust the individual distribution without revealing clients' raw data. Thus, the global model loss is converged faster and more accurately. The experimental results show that the proposed method improves FL accuracy and significantly reduces communication costs. We also provide a privacy analysis for the extra information used in FedDisk (i.e., the parameters of MADE models) and prove that the leakage information becomes less important when the number of clients increases. To advance our research, we intend to fine-tune the parameters of the MADE model and investigate various distribution models in order to potentially achieve improved outcomes and have a greater impact on real-world applications. More analysis on privacy vulnerabilities and potential attacks will also be our future research.

### 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 *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, A. Singh and J. Zhu, Eds., vol. 54, PMLR, 20–22 Apr 2017, pp. 1273–1282. [Online]. Available: <https://proceedings.mlr.press/v54/mcmahan17a.html>
- [2] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *ArXiv*, vol. abs/1806.00582, 2018.
- [3] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-10 (canadian institute for advanced research)." [Online]. Available: <http://www.cs.toronto.edu/~kriz/cifar.html>
- [4] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," 2018.
- [5] A. Hard, K. Rao, R. Mathews, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage, "Federated learning for mobile keyboard prediction," *CoRR*, vol. abs/1811.03604, 2018. [Online]. Available: <http://arxiv.org/abs/1811.03604>
- [6] S. P. Yadav, B. S. Bhati, D. P. Mahato, S. Kumar, and SpringerLink (Online service), *Federated Learning for IoT Applications*. Cham: Springer International Publishing Imprint Springer, 2022, oCLC: 1295622946.
- [7] I. Feki, S. Ammar, Y. Kessentini, and K. Muhammad, "Federated learning for COVID-19 screening from Chest X-ray images," *Applied Soft Computing*, vol. 106, p. 107330, Jul. 2021. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S1568494621002532>
- [8] S. Zhai, X. Jin, L. Wei, H. Luo, and M. Cao, "Dynamic Federated Learning for GMEC With Time-Varying Wireless Link," *IEEE Access*, vol. 9, pp. 10400–10412, 2021. [Online]. Available: <https://ieeexplore.ieee.org/document/9317862/>
- [9] A. K. Singh, D. Saxena, J. Kumar, and V. Gupta, "A quantum approach towards the adaptive prediction of cloud workloads," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 12, pp. 2893–2905, 2021.
- [10] W. Y. B. Lim, J. S. Ng, Z. Xiong, J. Jin, Y. Zhang, D. Niyato, C. Leung, and C. Miao, "Decentralized edge intelligence: A dynamic resource allocation framework for hierarchical federated learning," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 3, pp. 536–550, 2022.
- [11] H. Zhu, J. Xu, S. Liu, and Y. Jin, "Federated learning on non-iid data: A survey," *Neurocomputing*, vol. 465, pp. 371–390, 2021. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0925231221013254>

- [12] A. K. Sahu, T. Li, M. Sanjabi, M. Zaheer, A. S. Talwalkar, and V. Smith, "On the convergence of federated optimization in heterogeneous networks," *ArXiv*, vol. abs/1812.06127, 2018.
- [13] S. Itahara, T. Nishio, Y. Koda, M. Morikura, and K. Yamamoto, "Distillation-based semi-supervised federated learning for communication-efficient collaborative training with non-iid private data," *IEEE Transactions on Mobile Computing*, no. 01, pp. 1–1, mar 2021.
- [14] L. Liu, J. Zhang, S. Song, and K. B. Letaief, "Edge-assisted hierarchical federated learning with non-iid data," *CoRR*, vol. abs/1905.06641, 2019.
- [15] T. Shen, J. Zhang, X. Jia, F. Zhang, G. Huang, P. Zhou, F. Wu, and C. Wu, "Federated mutual learning," *ArXiv*, vol. abs/2006.16765, 2020.
- [16] H. Wang, Z. Kaplan, D. Niu, and B. Li, "Optimizing federated learning on non-iid data with reinforcement learning," in *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, 2020, pp. 1698–1707.
- [17] Q. Li, Y. Diao, Q. Chen, and B. He, "Federated learning on non-iid data silos: An experimental study," *CoRR*, vol. abs/2102.02079, 2021. [Online]. Available: <https://arxiv.org/abs/2102.02079>
- [18] X. Zhang, M. Hong, S. V. Dhople, W. Yin, and Y. Liu, "Fedpd: A federated learning framework with optimal rates and adaptivity to non-iid data," *CoRR*, vol. abs/2005.11418, 2020. [Online]. Available: <https://arxiv.org/abs/2005.11418>
- [19] M. Luo, F. Chen, D. Hu, Y. Zhang, J. Liang, and J. Feng, "No fear of heterogeneity: Classifier calibration for federated learning with non-iid data," in *Advances in Neural Information Processing Systems*, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., 2021.
- [20] Z. Zhu, J. Hong, and J. Zhou, "Data-free knowledge distillation for heterogeneous federated learning," 2021.
- [21] X. Li, M. JIANG, X. Zhang, M. Kamp, and Q. Dou, "FedBN: Federated learning on non-IID features via local batch normalization," in *International Conference on Learning Representations*, 2021. [Online]. Available: <https://openreview.net/forum?id=6YEQUn0QICQ>
- [22] A. K. Sahu, T. Li, M. Sanjabi, M. Zaheer, A. Talwalkar, and V. Smith, "On the convergence of federated optimization in heterogeneous networks," *CoRR*, vol. abs/1812.06127, 2018. [Online]. Available: <http://arxiv.org/abs/1812.06127>
- [23] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. Vincent Poor, "tackling the objective inconsistency problem in heterogeneous federated optimization," *Advances in Neural Information Processing Systems*, vol. "2020-December", "2020".
- [24] H. Wang, M. Yurochkin, Y. Sun, D. Papailiopoulos, and Y. Khazaeni, "Federated learning with matched averaging," in *International Conference on Learning Representations*, 2020. [Online]. Available: <https://openreview.net/forum?id=BkluqISFDS>
- [25] M. Mohri, G. Sivek, and A. T. Suresh, "Agnostic federated learning," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 4615–4625. [Online]. Available: <https://proceedings.mlr.press/v97/mohri19a.html>
- [26] M. Yurochkin, M. Agarwal, S. Ghosh, K. Greenwald, N. Hoang, and Y. Khazaeni, "Bayesian nonparametric federated learning of neural networks," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 7252–7261. [Online]. Available: <https://proceedings.mlr.press/v97/yurochkin19a.html>
- [27] H.-Y. Chen and W.-L. Chao, "On bridging generic and personalized federated learning for image classification," in *International Conference on Learning Representations*, 2022. [Online]. Available: <https://openreview.net/forum?id=I1hQbx10Kxn>
- [28] Y. Tan, G. Long, J. Ma, L. Liu, T. Zhou, and J. Jiang, "Federated learning from pre-trained models: A contrastive learning approach," 2022.
- [29] M. Wang, J. Guo, and W. Jia, "Fedcl: Federated multi-phase curriculum learning to synchronously correlate user heterogeneity," 2023.
- [30] J.-H. Duan, W. Li, and S. Lu, *FedDNA: Federated Learning with Decoupled Normalization-Layer Aggregation for Non-IID Data*, 09 2021, pp. 722–737.
- [31] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *MLSys*, 2020. [Online]. Available: <https://proceedings.mlsys.org/book/316.pdf>
- [32] M. Germain, K. Gregor, I. Murray, and H. Larochelle, "Made: Masked autoencoder for distribution estimation," in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 07–09 Jul 2015, pp. 881–889. [Online]. Available: <https://proceedings.mlr.press/v37/germain15.html>
- [33] C. Jin, L. T. Liu, R. Ge, and M. I. Jordan, "On the local minima of the empirical risk," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018.
- [34] A. Menon and C. S. Ong, "Linking losses for density ratio and class-probability estimation," in *Proceedings of The 33rd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. F. Balcan and K. Q. Weinberger, Eds., vol. 48. New York, New York, USA: PMLR, 20–22 Jun 2016, pp. 304–313. [Online]. Available: <https://proceedings.mlr.press/v48/menon16.html>
- [35] L. Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.
- [36] M. E. H. Chowdhury, T. Rahman, A. Khandakar, R. Mazhar, M. A. Kadir, Z. B. Mahbub, K. R. Islam, M. S. Khan, A. Iqbal, N. Al-Emadi, and M. B. I. Reaz, "Can AI help in screening viral and COVID-19 pneumonia?" *CoRR*, vol. abs/2003.13145, 2020. [Online]. Available: <https://arxiv.org/abs/2003.13145>
- [37] S. Jaeger, S. Candemir, S. Antani, Y.-X. Wang, P.-X. Lu, and G. Thoma, "Two public chest x-ray datasets for computer-aided screening of pulmonary diseases," *Quantitative imaging in medicine and surgery*, vol. 4, pp. 475–7, 12 2014.
- [38] D. S. Kermany, K. Zhang, and M. H. Goldbaum, "Labeled optical coherence tomography (oct) and chest x-ray images for classification," 2018.
- [39] Z. Wang, J. Qiu, Y. Zhou, Y. Shi, L. Fu, W. Chen, and K. B. Letaief, "Federated learning via intelligent reflecting surface," *IEEE Transactions on Wireless Communications*, vol. 21, no. 2, pp. 808–822, 2022.



**Hung Nguyen** received his Ph.D. degree in Department of Electrical Engineering, University of South Florida, FL, USA, in 2023. His current research interests include machine learning, artificial intelligence, federated learning, cyber security, privacy enhancing technologies. Hung is a member of IEEE.



**Pei-Yuan Wu** (Member, IEEE) was born in Taipei, Taiwan, in 1987. He received the B.S.E. degree in electrical engineering from National Taiwan University, Taipei, in 2009, and the M.A. and Ph.D. degrees in electrical engineering from Princeton University, Princeton, NJ, USA, in 2012 and 2015, respectively.



**J. Morris Chang** (SM'08) is a professor in the Department of Electrical Engineering at the University of South Florida. He received his Ph.D. degree from the North Carolina State University. His past industrial experiences include positions at Texas Instruments, Microelectronic Center of North Carolina and AT&T Bell Labs. He received the University Excellence in Teaching Award at Illinois Institute of Technology in 1999. His research interests include: cyber-security, wireless networks, and energy efficient computer systems. In the last six years, his research projects on cyber-security have been funded by DARPA. Currently, he is leading a DARPA project under Brandeis program focusing on privacy-preserving computation over Internet. He is a handling editor of Journal of Microprocessors and Microsystems and an editor of IEEE IT Professional. He is a senior member of IEEE.