



## Article

# One-Shot Federated Learning with Label Differential Privacy

Zikang Chen <sup>1</sup>, Changli Zhou <sup>1,2,\*</sup> and Zhenyu Jiang <sup>1</sup>

<sup>1</sup> College of Computer Science and Technology, Huaqiao University, Xiamen 361021, China; czk0620@gmail.com (Z.C.); jiangzy@stu.hqu.edu.cn (Z.J.)

<sup>2</sup> Xiamen Key Laboratory of Data Security and Blockchain Technology, Huaqiao University, Xiamen 361021, China

\* Correspondence: zcl@hqu.edu.cn

**Abstract:** Federated learning (FL) has emerged as an extremely effective strategy for dismantling data silos and has attracted significant interest from both industry and academia in recent years. However, existing iterative FL approaches often require a large number of communication rounds and struggle to perform well on unbalanced datasets. Furthermore, the increased complexity of networks makes the application of traditional differential privacy to protect client privacy expensive. In this context, the authors introduce FedGM: a method designed to reduce communication overhead and achieve outstanding results in non-IID scenarios. FedGM is capable of achieving considerable accuracy, even with a small privacy budget. Specifically, the authors devise a method to extract knowledge from each client's data by creating a scaled-down but highly effective synthesized dataset that can perform similarly to the original data. Additionally, the authors propose an innovative approach to applying label differential privacy to protect the synthesized dataset. The authors demonstrate the superiority of the approach over traditional methods by requiring only one communication round and by testing using four classification datasets for evaluation. Furthermore, when comparing the model performance for clients using their method against traditional solutions, the authors find that the approach achieves significant accuracy and better privacy.

**Keywords:** federated learning; differential privacy; dataset distillation; edge computing



**Citation:** Chen, Z.; Zhou, C.; Jiang, Z. One-Shot Federated Learning with Label Differential Privacy. *Electronics* **2024**, *13*, 1815. <https://doi.org/10.3390/electronics13101815>

Academic Editor: Aryya Gangopadhyay

Received: 16 March 2024

Revised: 23 April 2024

Accepted: 6 May 2024

Published: 8 May 2024



**Copyright:** © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Federated learning (FL) is a powerful approach designed to leverage decentralized computing and data resources to overcome data silos [1]. Despite its potential, FL encounters significant challenges when applied to modern neural networks, which often consist of hundreds of millions of parameters [2,3]. These challenges are compounded by the traditional FL approach, which requires numerous communication rounds, exacerbating the issues of scalability and communication load. This dual burden of excessive parameters and frequent communications significantly hinders the efficiency of FL methods and can impede the widespread deployment of FL models, especially in environments where communication resources are limited [4,5]. Additionally, the uneven distribution of data across different clients typically seen in real-world settings makes it difficult to optimize a global model [6–8]. This often results in model biases and suboptimal performance. These issues highlight the need for a new FL framework that not only reduces the communication overhead but also enhances the performance of the global model in scenarios with non-independent identically distributed (non-IID) data [9].

In this aspect, though FL offers the advantage of allowing participants to retain their raw data locally, which inherently addresses some aspects of data privacy [10], the privacy concerns in FL are far from being fully resolved, as both the model parameters exchanged during training and the outputs from the trained models can still be exploited for privacy attacks [10–12]. To counter these vulnerabilities, various strategies have been developed to enhance privacy in FL, including the use of trusted aggregators and complex cryptographic

techniques such as homomorphic encryption [13,14]. Despite these efforts, these methods often do not allow for personalized privacy settings and may not effectively protect against breaches involving high-dimensional parameter vectors, which are particularly challenging to secure [10].

In response to these privacy limitations, the LDP-fed model has been introduced as an innovative FL framework that supports local differential privacy. This model allows each participant to apply their own privacy settings, thus offering tailored privacy protection based on individual preferences [15,16]. However, this approach is weak given the increasing complexity and dimensionality of models used in modern FL applications, such as those based on deep learning architectures like ResNet and BERT [2,3,17]. Traditional privacy protection methods struggle to scale with the size of these models and often perform inadequately under high privacy requirements, necessitating extensive computational resources [8,18]. Therefore, LDP-fed requires innovations in protection mechanisms or greater efficiency and adaptability.

In this paper, a novel federated framework is designed to address issues related to both privacy and communication volumes. Traditional approaches often require each client to maintain and train its own model and send gradients or weights to a central server for aggregation. Instead, this work proposes creating compact yet high-performance surrogate data for each client. These surrogate datasets, which encapsulate knowledge from the original data, are difficult to visually interpret, thus preserving privacy. Meanwhile, only a one-shot communication round is required, achieving great server performance.

These local surrogate datasets are sent to the server, which constructs a global surrogate function to facilitate updates by minimizing this function. The surrogate data are generated through a process inspired by data distillation techniques. Local surrogate functions are crafted by synthesizing datasets that approximate the objectives; this is achieved by aligning gradients of the original and surrogate datasets across different neural network initializations.

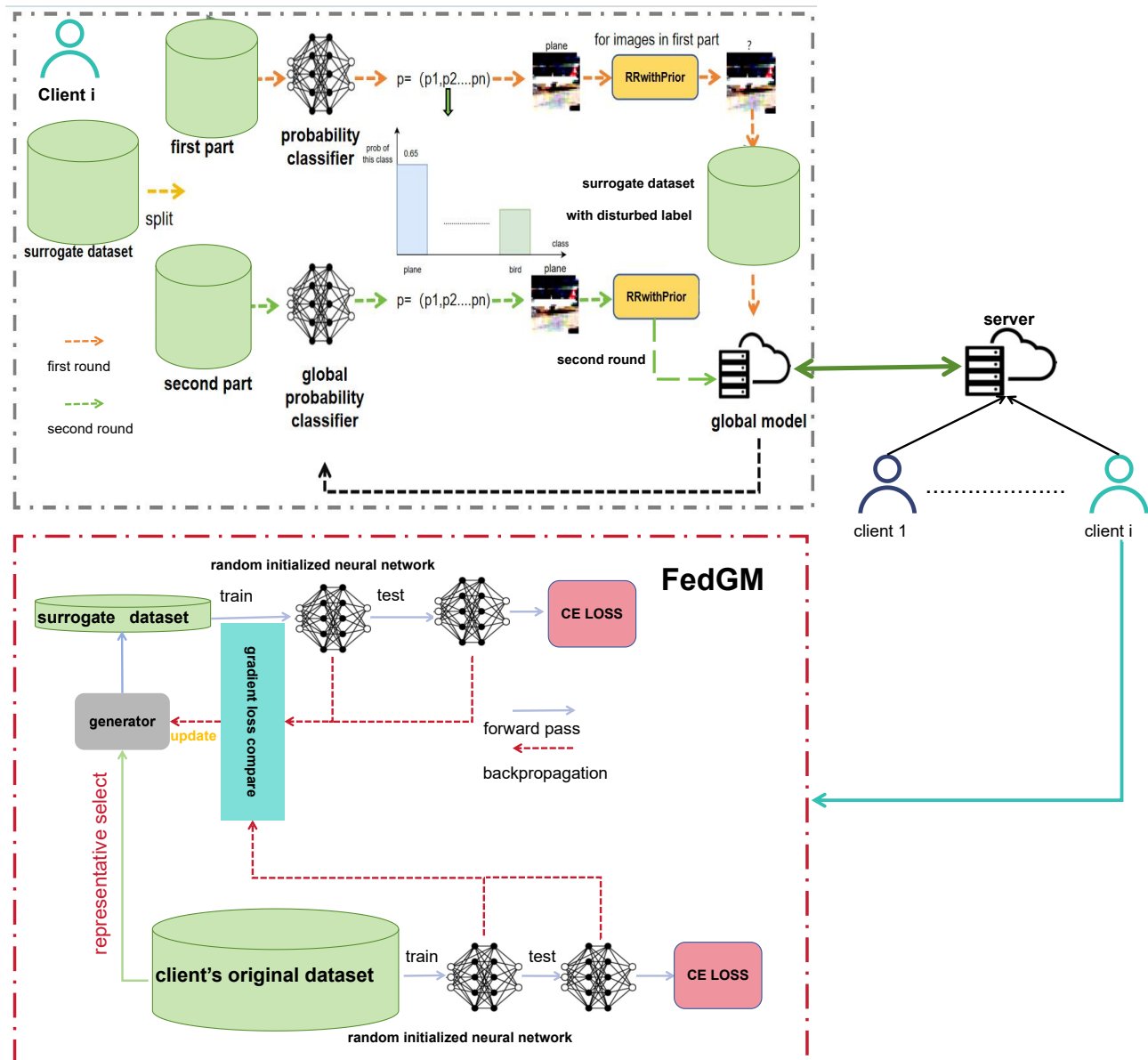
Moreover, the paper explores the challenge of maintaining privacy when surrogate datasets are used. Visual inspection alone does not reveal sensitive information from these datasets; however, using labels could potentially expose privacy-sensitive category information. To address this, an optimized label perturbation method tailored for federated learning (FL) scenarios is developed, striking a balance between data utility and privacy protection.

The framework, including the processes for creating and utilizing surrogate data, and the method for label perturbation are detailed in Figure 1. This approach significantly reduces communication costs and enhances privacy without compromising the usability of the data in federated learning environments. The contributions of this work are summarized as follows:

- A method called FedGM is introduced, which utilizes iterative gradient matching to learn a surrogate function. This technique involves transmitting synthesized data to the server rather than sending local model updates, significantly enhancing communication efficiency and effectiveness. Additionally, a novel strategy for selecting the original dataset reduces the number of training rounds required while improving the training effectiveness of the distilled dataset.
- Label differential privacy is employed to protect the privacy of approximate datasets for each client. This method is found to be highly capable even with a small privacy budget and outperforms other methods.
- Comprehensive experiments are conducted on three tasks and show that the proposed framework can achieve high performance with just one communication round in scenarios marked by pathological non-IID conditions.

The remainder of this paper is organized as follows: Section 2 reviews the related work of the paper. Section 3.1 presents the difference between the FedGM method and traditional FL. Section 3.2 provides a detailed introduction to the FedGM method, including its advantages and innovative aspects. Section 3.3 experimentally demonstrates that

the surrogate dataset sent by each client to the central authority in the FedGM setup is visually private, and label protection is achieved using an innovative approach to label differential privacy. In Section 4, numerous experiments are conducted to demonstrate the superiority of the work in terms of handling non-independent and identically distributed (non-IID) data, reducing communication overhead, and by comparing it with traditional LDP-fed approaches.



**Figure 1.** Total framework of the paper. Decentralized distilled datasets on federated CIFAR10 are divided into  $i$  clients. Each client has only  $n$  classes of data, and the class combination in each client is individual. Each client distills a surrogate dataset and perturbs it then sends the noise dataset to the server for training.

## 2. Related Works

### 2.1. Federated Learning

FL was first introduced by [1] and has the goal of training models collaboratively. Some well-known FL methods, such as FedAvg [1], involve aggregating models received from individual client devices and updating the global model by averaging the model parameters with appropriate weights. However, due to its reliance on iterative model

averaging, FedAvg encounters challenges related to heterogeneity in the FL system, particularly when dealing with non-IID data partitioning. This leads to a degradation in the overall performance of the global model and contributes to increased communication overhead. To mitigate this issue, several modifications and improvements to FedAvg have been introduced. For example, FedProx [9] adjusts the loss function, while FedNova [19] and SCAFFOLD [20] utilize additional information to address distribution shifts. In addition to enhancing learning algorithms for faster convergence, another approach to enhance efficiency is the explicit reduction in communication costs. However, the majority of current improved algorithms for non-IID scenarios in FL require multiple rounds of gradient and model exchange, resulting in a significant number of communication rounds. At the same time, the privacy of client models and their private data is also challenged by these approaches.

## 2.2. One-Shot Federated Learning

FL with one-shot communication has been explored in various projects. For instance, ref. [21] presents an algorithm for training a support vector machine in a one-shot manner and proposes a framework that not only uploads models but also includes information about the local dataset distribution: a task that is highly inappropriate when dealing with privacy-sensitive datasets. Meanwhile, ref. [22] employs knowledge transfer to distill models within each client, potentially altering the original model structure and incurring significant communication costs when sharing student models. Furthermore, ref. [23] attempts to alter the original optimization objective by sending distilled images to the central server, but it does not take privacy protection into account.

In this work, instead of distilling models, our framework allows each client to create an approximate dataset that mimics the performance of the original dataset and sends it to the central server.

## 2.3. Local Differential Privacy

Differential privacy [24] is a privacy-preserving technique designed to ensure that sensitive information of individuals is not disclosed when analyzing data. This method involves adding noise to the data in such a way that the analysis results do not depend on the data contributions of individual entities, thus preventing re-identification attacks against individuals. It provides a mathematical framework to balance the trade-off between privacy and data utility by controlling the introduction of random noise. Local differential privacy [25] is a variant of differential privacy that focuses on protecting the privacy of individual data rather than the results of the analysis. In local differential privacy (LDP), each individual client adds noise to its data locally before reporting it to the data processor to prevent the data processor from obtaining detailed information about individuals. This approach allows individuals to have autonomous control over the privacy of their data and reduces dependence on trusted centralized data processing. Due to the lack of a trusted central authority in real-world scenarios, recent research on privacy protection in federated learning has primarily focused on LDP. However, with the increasing complexity of models, traditional LDP privacy protection methods [15] in FL inevitably face increased computational requirements. Additionally, as privacy budgets change, users often experience a significant decrease in accuracy.

## 2.4. Dataset Distillation

Dataset distillation [26] has emerged as an essential paradigm in the context of machine learning, particularly when dealing with the challenges posed by the increasing size of training data. This approach focuses on generating a condensed version of a large dataset without significantly compromising the model's performance [27,28], primarily in terms of test accuracy. Initially proposed by [29], dataset distillation has been refined through various methodologies such as meta learning [29,30], gradient matching [31–33], distribution matching [34,35], neural kernels [27,28], and generative priors [36,37]. These

approaches show significant promise when applied to datasets like CIFAR10, yet they encounter difficulties when attempting to scale to larger datasets such as ImageNet.

### 2.5. Dataset Distillation in Federated Learning

Previous research [38] on dataset distillation has mentioned the benefits of dataset distillation in FL. FedD3 [23] has attempted to build a surrogate dataset in each client through KIP, but the authors did not use privacy protection technologies to safeguard the clients' distilled datasets. FedDM [39] tried to apply distribution matching for each client to build a surrogate dataset, and they used differential privacy to protect the distilled dataset. However, in the experiment, the research found that the distilled data itself are not sensitive, and some researches [40,41] have proven that the distilled data can withstand membership inference attacks. Thus, the research believes that privacy protection for distilled datasets should be more focused on the labels of the distilled dataset. Furthermore, both methods need a great deal of time to train at the client. In this work, the research uses a clustering process to select representative original images. This gradient matching target has shifted from randomly sampled original datasets at the outset to intentionally selected cluster datasets after clustering. Through experiments, the research has observed that the approach requires fewer training epochs compared to traditional dataset distillation in FL and achieves better performance more rapidly.

## 3. Methodology

In Section 3.1, the paper demonstrates the differences between the authors' framework and the traditional iterative minimization framework in FL.

In Section 3.2, the paper shows details of the authors' implementation of FedGM. The authors also demonstrate a novel method that can reduce the training time.

In Section 3.3, the paper analyzes the visual privacy of distilled images. Then, the authors proceed to demonstrate how to protect the distilled data using label differential privacy.

### 3.1. Differences between FedGM and the Traditional Iterative Minimization Framework in FL

Traditional FL is a scenario in which a set of clients individually train their own models using their private data, and a central server receives the models from the clients; the authors consider there are data  $D$  distributed among  $K$  clients, and each of the clients has a local private training dataset  $D_k = \{x_k^i, y_k^i\}, 1 \leq i \leq n_k$ , where  $n_k$  represents the number of data points assigned to each client, and  $\sum_{i=1}^k D_k = D$ . Each client has its weight value  $p^k$ , and  $\sum_{i=1}^k p^k = 1$ . In a multi-class classification task under FL, the minimization target is as follows:

$$\min_w \{L(w, D) \triangleq \sum_{k=1}^K p^k L^k(w, D_k)\} \quad (1)$$

where  $L^k(\cdot)$  is a local objective function that is optimized on the  $k$ -th client. The loss function  $L^k(\cdot)$  is formulated as follows:

$$L^k(w, D_k) \triangleq \frac{1}{n^k} \sum_{i=1}^{n^k} l(w, x_i^k, y_i^k) \quad (2)$$

where  $l(\cdot)$  is loss function for client, and  $w$  is model parameters. Most FL algorithms need the clients to transmit the locally trained models to the server multiple times: the communication burden cannot be neglected. Though one-shot FL has been proposed to reduce the cost of communication, its performance is not good. Meanwhile, as each client can only see local data, which can result in the absence of certain categories of information during the training process for each client, local updating is always unable to capture global information. Meanwhile, since the client is trained with biased data, it becomes challenging for the server to improve the joint update direction by taking into account more complex interactions among different clients. Regarding these factors, the authors

have successfully altered the local update strategy. Instead of transmitting the model itself, the client transmits surrogate data  $S_k$  that exhibits similar training performance to the original data  $D_k$ ; thus, the authors change the client local objective to

$$L^k(w) = \frac{1}{n_s^k} \sum_{i=1}^{n_s^k} l(w, \tilde{x}_i^k, \tilde{y}_i^k) \approx L^k(w, S_k) \quad (3)$$

Therefore, the server can train the local model using the surrogate data transmitted by each client. The authors can denote the objective as

$$\min_w L(w, D) \approx \min_w L(w, \sum_{k=1}^K S_k) \quad (4)$$

where  $L(\cdot)$  is a global objective function that is optimized on the server.

### 3.2. FL with Efficient Local Gradient Matching

The authors show how to build a surrogate dataset in each client in Section 3.2.1, and they introduce an improvement in the training efficiency for their method in Section 3.2.2.

#### 3.2.1. Local Gradient Matching

Drawing inspiration from recent advancements in data distillation [31–33], it becomes feasible to train a collection of synthesized data for each client that effectively represents the original data according to the objective function. The methodology of local gradient matching revolves around the concept of synthesizing a compact dataset that captures the essence of the original data distributed across clients in a federated setting. This compact dataset, denoted as  $S^*$ , is obtained by optimizing the following objective:

$$S^* = \arg \min_S C^T(\theta^S) \text{ subject to } \theta^S = \arg \min_{\theta} L^S(\theta). \quad (5)$$

Here,  $L^S(\theta)$  represents the loss function computed over the synthetic dataset  $S$ , and  $C^T(\theta^S)$  denotes the curriculum-based cost function over the course of  $T$  iterations. The process begins by initializing two models with identical architectures and parameters, which are then trained on the client's local original data and the synthetic dataset, respectively.

The optimization is carried out by matching the gradients of the synthetic model parameters ( $\theta^S$ ) with the target model parameters ( $\theta^T$ ) at each iteration. This is described by the equation:

$$\min_S E_{\theta_0 \sim P_{\theta_0}} \left[ \sum_{t=0}^{T-1} D(\theta_t^S, \theta_t^T) \right] \quad (6)$$

$$\text{subject to } \theta_{t+1}^S = \text{opt-alg}(\theta_t^S, \zeta_S) \text{ and } \theta_{t+1}^T = \text{opt-alg}(\theta_t^T, \zeta_T) \quad (7)$$

where  $D$  is a distance measure between the model parameters at each step, opt-alg denotes the optimization algorithm, and  $\zeta$  indicates the number of optimization steps. The primary goal is to generate synthetic samples that can result in gradient trajectories similar to those obtained from the original dataset.

In practice, this method leverages an approximate form of the backpropagation algorithm, which allows for a single backpropagation pass per step without the need for multiple levels of nesting, thereby simplifying the optimization process:

$$\min_S E_{\theta_0 \sim P_{\theta_0}} \left[ \sum_{t=0}^{T-1} D(\nabla_{\theta} L^S(\theta_t), \nabla_{\theta} L^T(\theta_t)) \right] \quad (8)$$

This approach ensures that the optimization is both memory-efficient and computationally feasible for deployment on edge devices within an FL framework.



### 3.2.2. Local Gradient Matching with Training Efficiency

The efficiency of training processes in dataset distillation is highly contingent on the choice of original images. Traditional methods rely on random sampling to select original images for matching, which often leads to a significant waste of training time. However, this framework enhances efficiency by employing a strategic sampling technique that not only reduces the training time of each client but also improves the effectiveness of the distilled images compared with random sampling.

The authors adopt a novel strategy that clusters samples of the same class from the client into different regions. A previous study [42] shows that samples closer to the class distribution center provide smaller backward gradients and more effective supervision, whereas samples near the boundary offer diverse conditions. This method allows us to extract surrogate datasets in each client that more accurately reflect the global data distribution, thereby improving the gradient matching process.

Gradient matching is then performed by aligning the synthetic images with the representative samples, which avoids the chaotic targeting that degrades performance in traditional methods. Furthermore, this approach mitigates the imbalance in mini-batch coverage of the original class sample distribution, addressing the instabilities in random sampling and optimization observed in previous approaches.

Inspired by [42], to get representative samples, the authors need to divide sub-clusters. So the authors utilize the k-means [43,44] algorithm. This clustering occurs within each class and generates sub-clusters that cover the entire sample space of the class: ensuring an even distribution and providing the algorithm with a diverse set of samples that accurately represent the sample density.

This clustering leads to a more balanced gradient distribution across different samples, aligning perfectly with the principle of maintaining diversity while ensuring an even distribution. The complete FL training pipeline is then followed, wherein each client distills the data locally and transmits the distilled dataset to the server. The server leverages the synthesized data to update the global model, significantly enhancing the training efficiency and effectiveness of the FL system.

The overall framework is shown in Algorithm 1. Each client distills surrogate data and transmits them to the server; then, the server updates the global model with the surrogate data from all the clients.

### 3.3. One-Shot FL with Label Differential Privacy

The previous study of FL with dataset distillation seldom mentioned data privacy protection, while some studies that mention privacy protection only focus on the distilled images themselves. In this part, the authors analyze the visual privacy of distilled images and find the distilled images themselves are not sensitive when the authors try to protect privacy. Thus, the authors consider protecting the labels with label differential privacy.

**Algorithm 1** Federated learning with efficient local gradient matching

---

```

1: Input: Client training set  $D_k$ , set of synthetic samples from client  $S_k$ , deep neural
   network parameterized with  $w$ , probability distribution over parameters  $P_w$ , global
   learning rate  $\eta_g$ , classes of original data  $c$ , size of minibatch gradient matching  $m$ ,
   number of steps for updating weights  $\zeta_\theta$  and synthetic samples  $\zeta_S$  in each inner-loop
   step, learning rates for updating weights  $\eta_\theta$  and synthetic samples  $\eta_S$ 
2: Server executes:
3: for  $k = 1, \dots, K$  do
4:   Transmit set of synthetic samples from client  $S_k$  to the server
5: end for
6: Aggregate synthesized dataset from each client and build the surrogate function update
   weights to  $w_{r+1}$  on  $N$  by SGD with the learning rate  $\eta_g$ 
7: Each client distills data:
8: for  $e = 1, \dots, E$  do
9:   Initialize  $\theta_0 \sim P_{\theta_0}$ 
10:  for  $t = 0, \dots, T - 1$  do
11:    for  $c = 0, \dots, C - 1$  do
12:      Get embeddings for each sample in  $D_k$  using  $\theta_0$ 
13:      Compute L2 distance among samples' embeddings and using k-means to
      cluster them into  $m$  subsets
14:      Sample a minibatch pair  $B_c^{D_k} \sim D_k$  from the center of clusters
15:      Sample a minibatch pair  $B_c^{S_k} \sim S_k$ 
16:       $B_c^{T_k}$  and  $B_c^{S_k}$  are of the same class  $c$ 
17:      Compute  $L_T^c = \frac{1}{|B_c^{D_k}|} \sum_{(x,y) \in B_c^{D_k}} \ell(\phi_{\theta_t}(x), y)$  and  $L_S^c = \frac{1}{|B_c^{S_k}|} \sum_{(s,y) \in B_c^{S_k}} \ell(\phi_{\theta_t}(s), y)$ 
18:      Update  $S_c \leftarrow opt - alg_{S_k}(D(\nabla_\theta L_{S_k}^c(\theta_t), \nabla_\theta L_{D_k}^c(\theta_t)), \zeta_S, \eta_{S_k})$ 
19:      Update  $\theta_{t+1} \leftarrow opt - alg_\theta(L_S(\theta_t), \zeta_\theta, \eta_\theta)$ 
20:    end for
21:  end for
22: end for
23: Output the set of synthetic samples from client  $S_k$ 
24: Transmit synthetic samples from client  $S_k$  to the server

```

---

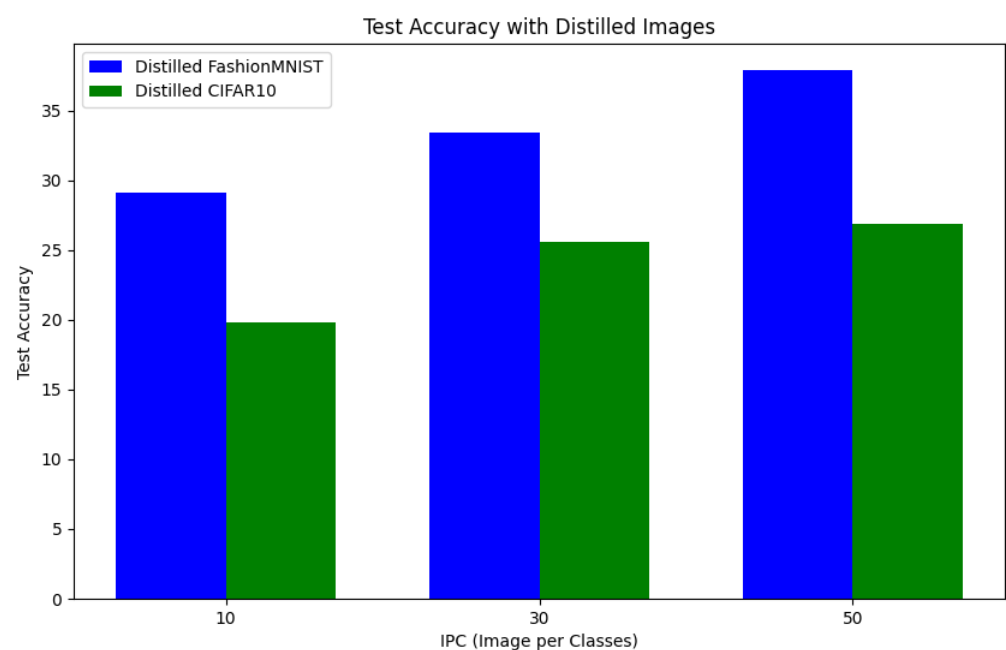
**3.3.1. Visual Privacy of Distilled Images**

Assume the authors have a malicious server. It can visualize synthetic data directly and make comparisons with the target sample to deduce membership. The authors use VGG [45] as the backbone to recognize the synthetic images and measure the similarity between the synthetic images and the original images using the L2 distance. Figure 2 shows examples of clients' distilled images and the their (top-two) most similar real images: images with the lowest L2 relative to the synthetic images at the tops of the columns [40]. The authors observe that the real data share similar contour patterns with the synthetic images, but the authors cannot observe more fine-grained features from the synthetic images. Meanwhile, the authors use ResNet [17] as the backbone to classify the distilled images; the authors input the distilled images as the test set into the network. Figure 3 shows the test accuracy. The authors find that few of the distilled images can be classified correctly, which means the distilled images themselves are not sensitive. An experiment is also conducted using the MMAFEDB facial expression recognition dataset provided by Kaggle to compare a dataset initialized with real images, as shown in Figure 4, and a dataset synthesized through knowledge distillation performed locally on the client side before being uploaded to the central party, as shown in Figure 5. It is found that the synthesized distilled dataset makes it difficult to observe the characteristics of the original data, as facial expressions and contours are distorted; thereby, the original facial information is protected. Thus, the authors try to protect the labels of distilled images.

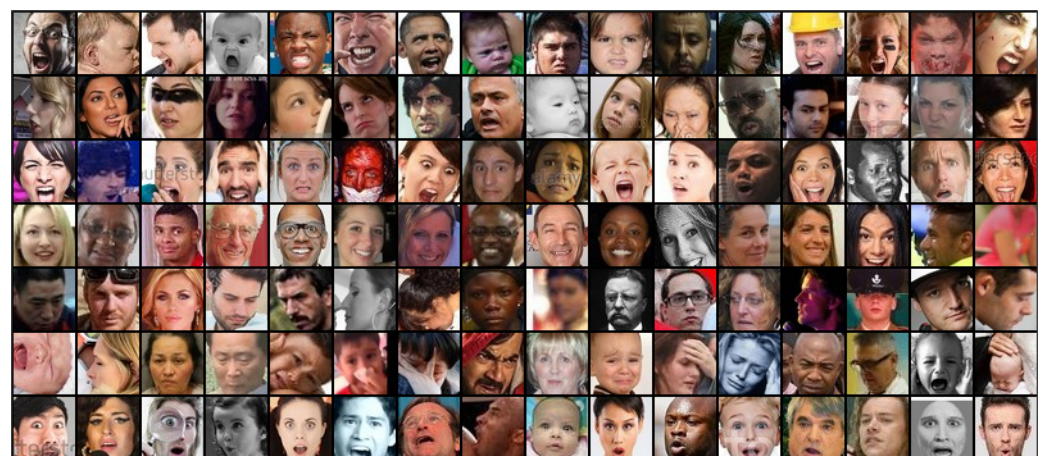




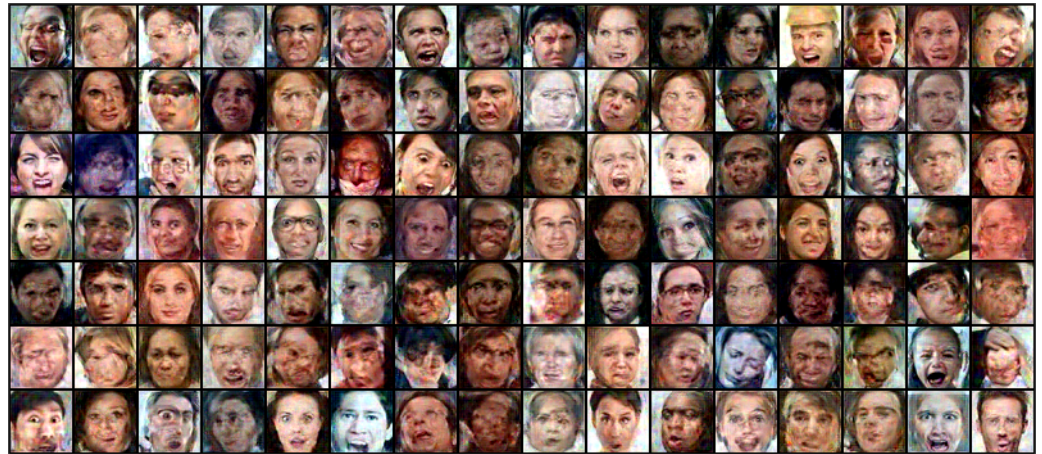
**Figure 2.** Examples of synthetic images that are most similar to synthetic data generated by clients with representative initialization. The value above each image is the L2 distance between the image and the synthetic data (first row). Lower distance indicates higher similarity. Even though these real images have outline information, the color or more detail are difficult for the adversary to infer.



**Figure 3.** Using synthetic data generated from CIFAR10 and Fashionmnist as test sets.



**Figure 4.** Synthetic data initialized by real images.



**Figure 5.** Synthetic data trained over 1000 iterations within the client.

### 3.3.2. Implementation of Label Differential Privacy

To protect the labels of the distilled dataset, the authors apply label differential privacy. Before analyzing their method, the authors first review some fundamentals of differential privacy.

**Definition 1** (Randomized Response [24]). *Given a positive integer  $K$ , let  $[K] = \{1, \dots, K\}$ . The randomized response RR is defined as follows: for a parameter  $\epsilon \geq 0$  and a true value  $y \in [K]$  known to RR, when an observer queries the value of  $y$ , the RR mechanism provides a random response  $\tilde{y}$  according to the following probability distribution:*

$$\Pr(\tilde{y} = y) = \begin{cases} \frac{e^\epsilon}{e^\epsilon + K - 1} & \text{if } \tilde{y} = y, \\ \frac{1}{e^\epsilon + K - 1} & \text{if } \tilde{y} \neq y. \end{cases} \quad (9)$$

**Definition 2** (Differential Privacy [24]). *Let  $\epsilon, \delta \in \mathbb{R}_{\geq 0}$ . A randomized algorithm  $A$  taking as input a dataset is said to be  $(\epsilon, \delta)$ -differentially private  $(\epsilon, \delta)$ -DP if for any two neighboring datasets  $D$  and  $D'$  and for any subset  $S$  of outputs of  $A$ , it is the case that  $\Pr[A(D) \in S] \leq e^\epsilon \cdot \Pr[A(D') \in S] + \delta$ . If  $\delta = 0$ , then  $A$  is said to be  $\epsilon$ -differentially private ( $\epsilon$ -DP).*

**Definition 3** (Label Differential Privacy [18]). *Let  $\epsilon, \delta \in \mathbb{R}_{\geq 0}$ . A randomized training algorithm  $A$  taking as input a dataset is said to be  $(\epsilon, \delta)$ -label differentially private  $(\epsilon, \delta)$ -Label DP if for any two training datasets  $D$  and  $D'$  that differ in the label of a single example and for any subset  $S$  of outputs of  $A$ , it is the case that  $\Pr[A(D) \in S] \leq e^\epsilon \cdot \Pr[A(D') \in S] + \delta$ . If  $\delta = 0$ , then  $A$  is said to be  $\epsilon$ -label differentially private ( $\epsilon$ -Label DP).*

Traditional label differential privacy employs a randomized response algorithm on the label of each data point equally. However, this approach can significantly impact the training effectiveness. Motivated by [18], the authors employ RRTop- $k$  and RRwithPrior, which is shown in Algorithms 2 and 3 and is a method that, by leveraging prior probability knowledge of labels, achieves better label perturbation, striking a balance between privacy and data utility.

In detail, the authors consider the global model  $M_t$  as a probabilistic classifier to each client. Given an unlabeled sample  $x$ , this classifier enables the assignment of a probability  $p_y$  for each class  $y \in [K]$ , and each client partitions its distilled data into  $T$  non-overlapping subsets. The probabilistic classifier initially outputs an identical probability distribution across all classes in the data.

**Algorithm 2** RRTop- $k$ **Require:** A label  $y \in [K]$   $k \in [K]$ , prior  $p = (p_1, \dots, p_K)$ 

Let  $Y_k$  be the set of  $k$  labels with maximum prior probability (with ties broken arbitrarily)

2: **if**  $y \in Y_k$  **then**  
     Then output  $y$  with probability  $\frac{\epsilon}{\epsilon+k-1}$  and output  $y' \in Y_k \setminus \{y\}$  with probability  $\frac{1}{\epsilon+k-1}$

4: **else**  
     Output an element from  $Y_k$  uniformly at random

6: **end if**

**Algorithm 3** RRWithPrior**Require:** A label  $y \in [K]$  prior  $p = (p_1, \dots, p_K)$ 

**for**  $k \in [K]$  **do**  
     Compute  $w_k := \frac{\epsilon}{\epsilon+k-1} \cdot \left( \sum_{j \notin Y_k} p_j \right)$ , where  $Y_k$  is the set of  $k$  labels with maximum prior probability (with ties broken arbitrarily)

3: **end for**  
     Let  $k^* = \arg \max_{k \in [K]} w_k$   
     Return an output of RRTop- $k$  ( $y$ ) with  $k = k^*$

Using the probabilistic classifier obtained from the previous round of training, the authors assign corresponding probabilities  $p = (p_1, \dots, p_K)$  for each sample  $x_i$  in the subset. Utilizing these probabilities,  $p$  helps to refine the randomized response (RR): specifically, through the RRWithPrior algorithm. The perturbed labels obtained through RRWithPrior contribute to the formation of the perturbed dataset. All the data subjected to RRWithPrior are combined into a larger noise set and sent to the central server for training. Then, the server learns the prior knowledge of the dataset and sends the global model  $M_t$  to clients as the probabilistic classifier. The authors explain the entire process in Algorithm 4.

**Algorithm 4** One-Shot Federated Learning with Label Differential Privacy

**Input:** Client synthetic dataset  $S_k$ , deep neural network parameterized with  $w$ , global learning rate  $\eta_g$ , global model  $M$

**Output:** Noise set for central server training

**for** each client  $k$  **do**

4:   Partition the distilled data into  $T$  non-overlapping subsets  
     **for**  $t = 1, \dots, T$  **do**  
         Use the classifier  $M_t$  from the previous round to assign probabilities  $p = (p_1, \dots, p_K)$   
         Refine randomized response using the RRWithPrior algorithm

8:   Form the perturbed dataset with the obtained labels  
     **end for**  
     Combine all perturbed data into a noise set  
     Send the noise set to the central server

12: **end for**  
     Server executes:  
     **for**  $k = 1, \dots, K$  **do**  
         Transmit set of synthetic samples from client  $S_k$  to the server

16: **end for**  
     Aggregate synthesized dataset from each client and update weights to  $w_{r+1}$  on client noise set by SGD with the learning rate  $\eta_g$  and send the global model  $M_t$  as classifier to clients

### 3.3.3. Proof of Differential Privacy

To demonstrate that this algorithm conforms to the definition of differential privacy, it is imperative to demonstrate that the RRWithPrior algorithm satisfies differential privacy. The authors begin by scrutinizing the differential privacy properties of the RRTop- $k$  mechanism shown in Algorithm 2. Consider two arbitrary inputs  $y$  and  $y'$  from the set  $[K]$  and an arbitrary possible outcome  $\hat{y}$  from the set  $Y_k$ . The authors observe that the probability  $\Pr[\text{RRTop-}k(y) = \hat{y}]$  is at its maximum when  $y$  is the same as  $\hat{y}$ , while  $\Pr[\text{RRTop-}k(y') = \hat{y}]$  is minimized when  $y'$  is any element of  $Y_k \setminus \{\hat{y}\}$ . This leads to the following relationship:

$$\frac{\Pr[\text{RRTop-}k(y) = \hat{y}]}{\Pr[\text{RRTop-}k(y') = \hat{y}]} = \frac{\frac{\epsilon}{\epsilon+k-1}}{\frac{1}{\epsilon+k-1}} = \epsilon. \quad (10)$$

Therefore, RRTop- $k$  satisfies  $\epsilon$ -differential privacy.

Moreover, since RRWithPrior can be conceptualized as the RRTop- $k$  algorithm, which calculates the threshold  $k$  maximizing  $\Pr[\text{RRTop-}k(y) = y]$ , it follows that RRWithPrior also satisfies  $\epsilon$ -differential privacy.

To further substantiate that this algorithm is compliant with differential privacy, the authors consider any  $\epsilon > 0$  and posit that if RRWithPrior is  $\epsilon$ -DP, then this one-shot federated learning with differential privacy (OSFLDP) is  $\epsilon$ -LabelDP. The algorithm maintains  $\epsilon$ -DP even when disclosing all  $T$  models  $M^{(1)}, \dots, M^{(T)}$  and all randomized labels  $y_1, \dots, y_n$ . The joint probability for any conceivable set of output models  $m^{(1)}, \dots, m^{(T)}$  and corresponding output labels  $z_1, \dots, z_n$  can be partitioned into a product of individual probabilities for each model and a product of conditional probabilities for each label given by the trained global model.

When examining two datasets  $D$  and  $D'$  that are identical except for the label of a single user, the probability terms for  $D$  and  $D'$  are congruent for all but one term related to the differing user's label. This particular term is indicative of the probability that RRTop- $k$  assigns a specific label. As RRWithPrior guarantees  $\epsilon$ -DP, it ensures that the discrepancy in probabilities caused by changing one user's label is bounded by  $e^\epsilon$ . Hence, this algorithm satisfies  $\epsilon$ -differential privacy, thereby fulfilling  $\epsilon$ -LabelDP as desired.

## 4. Experiments

### 4.1. Experimental Settings

In this work, the authors conduct experiments on MNIST [46], FashionMNIST [47], CIFAR10 [48], and SVHN [49]. In the experiments, the authors use a three-layer convolutional network (ConvNet-3) [50] with 128 filters and instance normalization for each client to employ local gradient matching and to serve as a probabilistic classifier in each client. The authors select the batch size as 128 for representative real images, and the authors conduct 300 matching iterations in each client, inside each of which 100 inner loops are conducted. SGD [51] is set as the optimizer for each clients' distilled image generator with a learning rate of 0.01. The clustering process utilizes a matching model for extracting features. The interval between each clustering iteration, referred to as  $D_{int}$ , is established at 10 iterations. The training efficiency driven by this clustering process is examined in Section 4.5 of the analysis. The authors tune the number of images per class (ipc) within [3, 5, 10] for each client under the IID setting, and the number of clients is 10, but when it comes to the non-IID setting, the authors consider a thornier scene, which is called a label distribution skew case. A previous study [52] showed that among different non-IID data settings, all studied FL algorithms performed worse on the label distribution skew case, and for the label distribution skew setting, the algorithms have the worst accuracy when each party only has data from a single label, which is also called pathological non-IID; thus, the authors focus on this scene to show the performance of the method. The authors design a pathological non-IID scene wherein each client only has one or two classes of data. In this non-IID setting, the authors tune the ipc within [30, 50, 100], and each client's synthetic images are initialized as representative sampled real images. For the baseline method, the



authors choose a batch size of 128 for local training. The authors tune the learning rate at the client from [0.001, 0.01, 0.1] and at the server from [0.01, 0.1, 1], and the local epoch is from [1, 2, 5, 10, 15, 20]. In particular, the authors tune  $\mu$  for FedProx in [0.001, 0.01, 0.1, 1].

The authors compare FedGM with four representative model-averaging-based methods. The federated methods are: FedAvg [1], FedProx [9], FedNova [19], and SCAFFOLD [20]; the four federated learning methods and their comparative results are shown in Table 1.

**Table 1.** Comparison of results of various methods. For non-IID settings, ipc = 50; for IID settings, ipc = 10. C represents the number of data classes a client has. The authors also include a Dirichlet distribution scenario with  $\alpha = 0.5$ . The number of clients is 10.

Dataset	Partitioning	FedAvg	FedProx	SCAFFOLD	FedNova	FedGM
MNIST	$p_k \sim \text{Dir}(0.5)$	98.9% $\pm$ 0.1%	98.9% $\pm$ 0.1%	99.0% $\pm$ 0.1%	98.9% $\pm$ 0.1%	96.1% $\pm$ 0.1%
	#C = 1	29.8% $\pm$ 7.9%	40.9% $\pm$ 23.1%	39.9% $\pm$ 0.2%	39.2% $\pm$ 22.1%	99.3% $\pm$ 0.1%
	#C = 2	97.0% $\pm$ 0.4%	96.4% $\pm$ 0.3%	95.9% $\pm$ 0.2%	94.5% $\pm$ 1.5%	99.2% $\pm$ 0.1%
	IID	99.1% $\pm$ 0.1%	99.1% $\pm$ 0.1%	99.1% $\pm$ 0.2%	99.1% $\pm$ 0.2%	99.2% $\pm$ 0.1%
FMNIST	$p_k \sim \text{Dir}(0.5)$	88.1% $\pm$ 0.1%	88.1% $\pm$ 0.4%	88.1% $\pm$ 0.5%	88.6% $\pm$ 0.3%	89.6% $\pm$ 0.3%
	#C = 1	10.2% $\pm$ 0.1	28.8% $\pm$ 0.3%	11.8% $\pm$ 0.2%	15.8% $\pm$ 0.2%	85.7% $\pm$ 0.1%
	#C = 2	74.3% $\pm$ 0.1%	71.3% $\pm$ 0.1%	42.8% $\pm$ 28.7%	69.5% $\pm$ 5.7%	85.3% $\pm$ 0.1%
	IID	89.3% $\pm$ 0.1%	89.1% $\pm$ 0.2%	89.8% $\pm$ 0.1%	89.4% $\pm$ 0.3%	87% $\pm$ 0.0%
CIFAR-10	$p_k \sim \text{Dir}(0.5)$	68.4% $\pm$ 0.3%	65.9% $\pm$ 0.5%	65.8% $\pm$ 0.9%	65.3% $\pm$ 1.5%	61.4% $\pm$ 0.3%
	#C = 1	10.3% $\pm$ 0.5%	12.3% $\pm$ 2.0%	10.0% $\pm$ 0.0%	10.0% $\pm$ 0.0%	74.5% $\pm$ 0.1%
	#C = 2	49.8% $\pm$ 3.3%	50.7% $\pm$ 1.7%	49.1% $\pm$ 1.7%	46.5% $\pm$ 3.5%	75.5% $\pm$ 0.2%
	IID	72.4% $\pm$ 0.2%	70.2% $\pm$ 0.1%	71.5% $\pm$ 0.3%	69.5% $\pm$ 1.0%	62.4% $\pm$ 0.4%
SVHN	$p_k \sim \text{Dir}(0.5)$	86.1% $\pm$ 0.7%	86.6% $\pm$ 0.9%	86.8% $\pm$ 0.3%	86.4% $\pm$ 0.6%	86.5% $\pm$ 0.3%
	#C = 1	11.1% $\pm$ 0.1%	18.6% $\pm$ 0.2%	6.8% $\pm$ 0.0%	10.6% $\pm$ 0.2%	85.3% $\pm$ 0.2%
	#C = 2	79.2% $\pm$ 0.5%	80.3% $\pm$ 0.5%	64% $\pm$ 10.6%	72.4% $\pm$ 3.8%	85.7% $\pm$ 0.4%
	IID	87.5% $\pm$ 0.3%	85.5% $\pm$ 0.7%	88.5% $\pm$ 0.2%	87.4% $\pm$ 0.4%	87.9% $\pm$ 0.4%

#### 4.2. Comparison with Other One-Shot FL Methods

In this section, the authors compare this new approach with the relatively novel one-shot FL methods. The novel one-shot FL methods are FedDM [39], DENSE [53], FedD3 [23], and FedDF [54]. In comparison with the latest one-shot FL methods, FedDM [39] is essentially a multi-round FL approach. However, due to its methodological approximations, it has been adapted for evaluation in a one-shot environment. The authors configured the system with 10 clients and compared the performance of this new method against these baseline methods on the MNIST [46] and CIFAR-10 datasets [48]. The comparisons were conducted under both IID settings and a scenario with a Dirichlet distribution with  $\alpha = 0.5$ . Through this comparison, the authors found that FedGM can achieve outstanding results, as shown in Table 2.

**Table 2.** Results compared with other one-shot FL methods. For non-IID settings, ipc = 50; for IID settings, ipc = 10. The authors set ipc = 10 for each method.

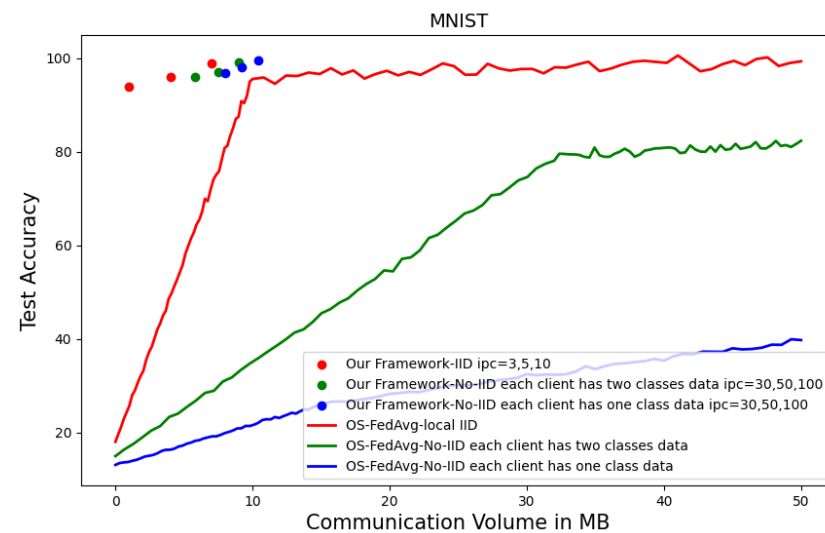
Dataset	Partitioning	FedGM (Ours)	FedDM	FedDF	FedD3	DENSE
MNIST	IID	99.21%	98.01%	96.13%	94.71%	99.13%
	$p_k \sim \text{Dir}(0.5)$	96.13%	99.01%	92.23%	94.1%	95.82%
CIFAR10	IID	62.41%	58.12%	54.12%	49.31%	64.1%
	$p_k \sim \text{Dir}(0.5)$	62.93%	61.83%	53.56%	40.15%	62.19%

#### 4.3. Efficient and Robust Training on Heterogeneous Data

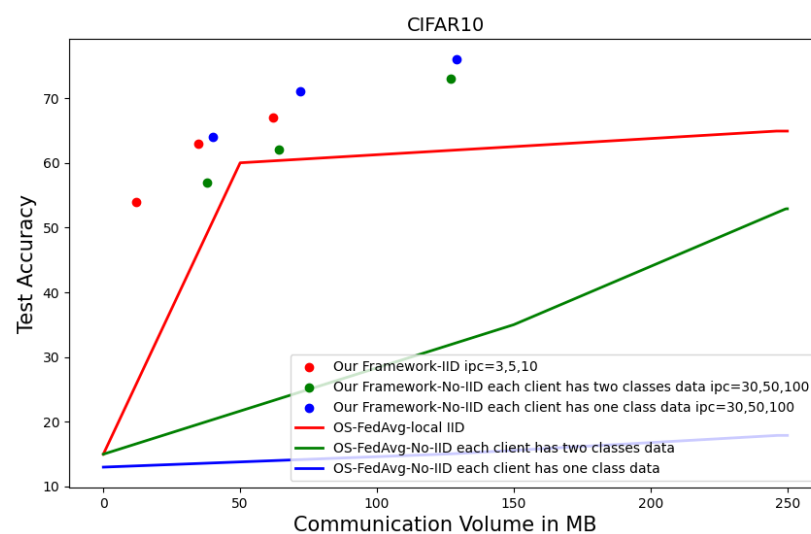
The evaluation of FedGM focuses on two key aspects: the communication volume required to achieve optimal accuracy and the performance under data heterogeneity; the authors use one-shot FedAvg as the baseline for comparison.

In FedGM, communication overhead is generated by transmitting condensed images (using  $1 \times 8$  or  $3 \times 8$  bits per grayscale or RGB pixel, respectively), whereas in traditional federated learning approaches, this overhead is due to the transmission of model parameters and requires 32 bits per parameter.

In Figures 6 and 7, the authors find that with unlimited communication rounds, FedAvg outperforms the new method under the IID scenario, but the best performance of one-shot FedAvg (OS-FedAvg) can be reached by using FedGM with only around one-third of the communication volume on average. However, when the data heterogeneity across clients increases, the accuracy of the new method remains unchanged, while the prediction accuracy of FedAvg decreases notably, which demonstrates that the new method is more efficient under conditions of limited communication resources and significant data heterogeneity.



**Figure 6.** MNIST: test accuracy changes with increasing total communication volume in Mb required in each method on MNIST. The training results are compared with OS-FedAvg under non-IID setting and IID setting. For non-IID scenario, the ipc is set within [30,50,100], and each client only has one or two classes of data.

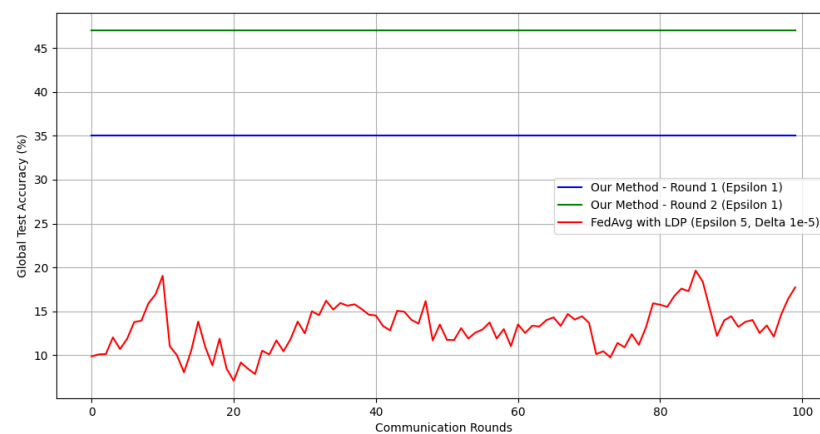


**Figure 7.** CIFAR10: test accuracy changes with increasing total communication volume in Mb required in each method on CIFAR-10. The training results are compared with OS-FedAvg under non-IID setting and IID setting. For IID scenario, ipc is set within [3, 5, 10].

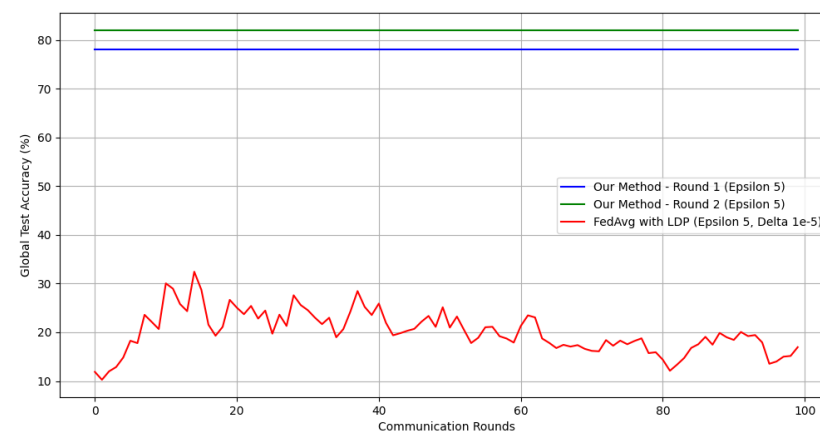


#### 4.4. Performance with DP Guarantee

As demonstrated in Section 3.3, the labels of the client's distilled dataset are protected with label differential privacy when being transmitted to server, and the label differential privacy in FedGM can satisfy  $\epsilon$ -differential privacy. In this approach, each client is pre-configured to divide its distilled dataset into two non-overlapping datasets. One half is used directly in the RRwithprior process with a probability classifier that operates without learned prior knowledge, and this portion sends noise data to the server for learning prior knowledge. The other half utilizes the existing prior knowledge to obtain improved perturbed labels. These labels are then sent to the server for second rounds of training. The authors compare FedGM with traditional LDP-FedAvg and find FedGM can achieve higher accuracy compared with the baseline, as shown in Figures 8–10. The traditional LDP-FedAvg undergoes multiple rounds of communication with the central party for training. In contrast, the one-shot method requires only two rounds of communication: the first round is to acquire prior knowledge, and the second round is for further optimization training with label differential privacy utilizing the acquired prior knowledge from the global model.



**Figure 8.**  $\epsilon = 1$ : In the context of IID scenarios with the MNIST dataset employing Gaussian differential privacy with delta parameter set to  $\times 10^{-5}$  and exploring an epsilon value of 1.



**Figure 9.**  $\epsilon = 5$ : In the context of IID scenarios with the MNIST dataset employing Gaussian differential privacy with delta parameter set to  $\times 10^{-5}$  and exploring an epsilon value of 5.



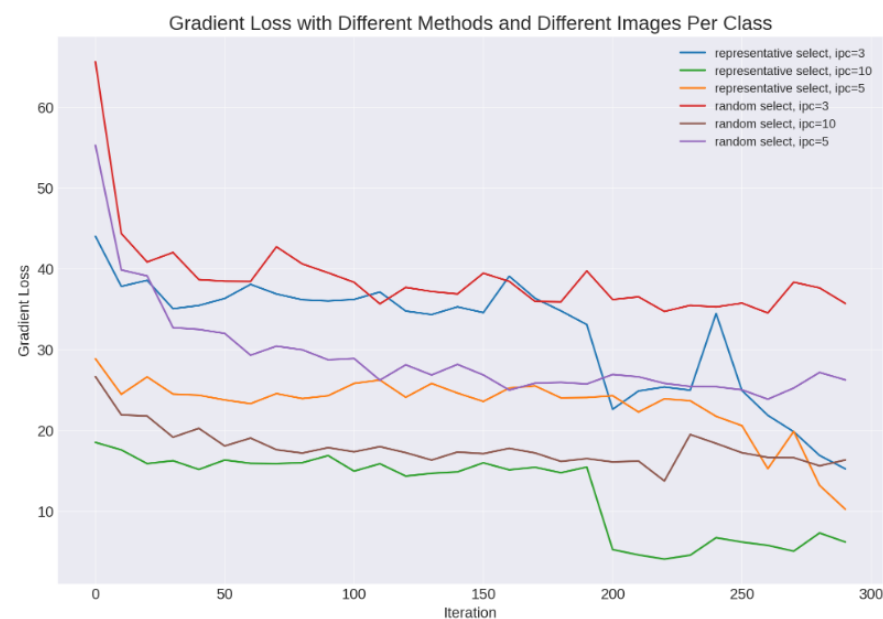
**Figure 10.**  $\epsilon = 10$ : In the context of IID scenarios with the MNIST dataset employing Gaussian differential privacy with delta parameter set to  $\times 10^{-5}$  and exploring an epsilon value of 10.

#### 4.5. Analyzing the Efficiency Brought by Matching Representative Images

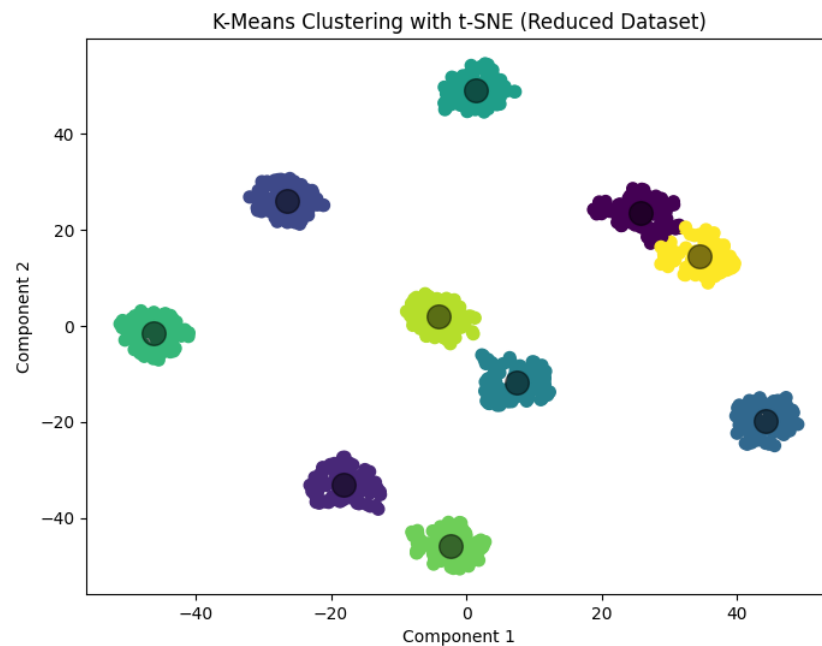
The authors show the training efficiency curve of matching the synthetic images with only the random or representative samples in Figure 11; different numbers of images per class are also considered in Figure 11. The authors have observed that by matching synthetic images with representative samples, the gradient loss between the original and synthetic datasets decreases more rapidly and significantly. This suggests that the synthetic dataset exhibits similar training performance to the original dataset. Additionally, it implies reduced training time and enhanced performance for client-side local private data. Moreover, the authors also note that a larger ipc correlates with fewer required training rounds and smaller gradient loss.

Example clustering and sub-cluster center results of matching representative images are also shown in Figure 12.

In conclusion, the authors find that with matching representative images, each client needs fewer epochs to train synthetic data, and the training accuracy with synthetic data is obviously improved.



**Figure 11.** The authors focus on the variation of client-side gradient loss within an IID distribution scenario, specifically using the FashionMNIST dataset. This analysis includes the impact of different ipc values and the effects of two distinct matching selection methods.



**Figure 12.** K-means is used for dividing sub-clusters within each class, creating  $N$  sub-clusters that mirror the density of samples. The value of  $N$  is predetermined and serves as the hyper-parameter for the mini-batch size of actual images. This approach ensures that the clustering is specifically tailored to the characteristics and distribution of the samples within each class.

#### 4.6. Comparison with Transmitting Real Images

The authors' approach is pitted against REAL, which transmits authentic images with dimensions identical to those of FedGM ( $\text{ipc} = 10$ ) under the IID setting. To elaborate, REAL attains a test accuracy of on CIFAR10 using the default configuration. However, it falls short of surpassing FedGM, which achieves a test accuracy of  $67.5\% \pm 0.2\%$ . This suggests that the acquired synthetic dataset is capable of encapsulating a more extensive range of information from the entire dataset as opposed to merely using a handful of images.

## 5. Conclusions and Limitation

In the endeavor to advance the domain of federated learning, FedGM is introduced; it is a pioneering one-shot federated learning framework engineered to diminish the necessity for recurrent communication rounds, which frequently encumber traditional federated learning systems. This innovative approach not only optimizes communication efficiency but also significantly enhances the training effectiveness of the distilled dataset. By capitalizing on the unique strategy of leveraging synthetic datasets that mimic the original data's distribution, FedGM achieves remarkable training accuracy and privacy protection while demonstrating an exceptional balance between performance and efficiency.

FedGM's methodology of employing label differential privacy further exemplifies its novel contribution to ensuring data privacy while maintaining high utility in training models. This strategy, which is meticulously designed to protect the labels of the pseudo datasets, showcases a sophisticated understanding of the challenges inherent in federated learning environments. The empirical results, which are corroborated by extensive experiments, underscore the superior performance of FedGM over traditional methods, especially in the context of non-IID data scenarios and under stringent privacy constraints.

Moreover, the research illuminates the significance of visual privacy in federated learning, which is a novel area that has received limited attention in prior studies. The findings reveal that, while the distilled images themselves may not contain sensitive information, the labels attached to them possess a higher degree of sensitivity, necessitating the innovative approach to label privacy adopted by FedGM.

However, the study also acknowledges its limitations: particularly concerning the scalability of the synthetic dataset with an increasing number of clients. This recognition of potential constraints not only underscores the authenticity and originality of the work but also paves the way for future research to explore innovative solutions to these challenges.

**Author Contributions:** Z.C. was responsible for the coding and algorithm design as well as the overall writing of the manuscript. C.Z. was responsible for providing resources and funding assistance. Z.J. was responsible for conducting part of the comparative experiments. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported in part by the National Natural Science Foundation of China under grants (61802134, 61872154, 61972166, and 61972168), the Fundamental Research Funds for the Central Universities (ZQN-811), and the Natural Science Foundation of Fujian Province of China under grant 2020J05059.

**Data Availability Statement:** <https://github.com/baldcodeman/One-Shot-Federated-Learning-with-Label-Differential-Privacy> (accessed on 15 March 2024).

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

- McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; y Arcas, B.A. Communication-efficient learning of deep networks from decentralized data. In Proceedings of the Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA, 20–22 April 2017; pp. 1273–1282.
- Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv* **2020**, arXiv:2010.11929.
- Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
- Zeadally, S.; Javed, M.A.; Hamida, E.B. Vehicular communications for ITS: Standardization and challenges. *IEEE Commun. Stand. Mag.* **2020**, *4*, 11–17. [[CrossRef](#)]
- Karlstetter, R.; Raoofy, A.; Radev, M.; Trinitis, C.; Hermann, J.; Schulz, M. Living on the edge: Efficient handling of large scale sensor data. In Proceedings of the 2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid), Melbourne, Australia, 10–13 May 2021; pp. 1–10.
- Kairouz, P.; McMahan, H.B.; Avent, B.; Bellet, A.; Bennis, M.; Bhagoji, A.N.; Bonawitz, K.; Charles, Z.; Cormode, G.; Cummings, R.; et al. Advances and open problems in federated learning. *Found. Trends® Mach. Learn.* **2021**, *14*, 1–210. [[CrossRef](#)]
- Konečný, J.; McMahan, H.B.; Yu, F.X.; Richtárik, P.; Suresh, A.T.; Bacon, D. Federated learning: Strategies for improving communication efficiency. *arXiv* **2016**, arXiv:1610.05492.
- Li, Q.; Wen, Z.; Wu, Z.; Hu, S.; Wang, N.; Li, Y.; Liu, X.; He, B. A survey on federated learning systems: Vision, hype and reality for data privacy and protection. *IEEE Trans. Knowl. Data Eng.* **2021**, *35*, 3347–3366. [[CrossRef](#)]
- Li, T.; Sahu, A.K.; Zaheer, M.; Sanjabi, M.; Talwalkar, A.; Smith, V. Federated optimization in heterogeneous networks. *Proc. Mach. Learn. Syst.* **2020**, *2*, 429–450.
- Mothukuri, V.; Parizi, R.M.; Pouriyeh, S.; Huang, Y.; Dehghantanha, A.; Srivastava, G. A survey on security and privacy of federated learning. *Future Gener. Comput. Syst.* **2021**, *115*, 619–640. [[CrossRef](#)]
- Zhu, L.; Liu, Z.; Han, S. Deep leakage from gradients. *arXiv* **2019**, arXiv:1906.08935.
- Wei, W.; Liu, L.; Loper, M.; Chow, K.H.; Gursoy, M.E.; Truex, S.; Wu, Y. A framework for evaluating gradient leakage attacks in federated learning. *arXiv* **2020**, arXiv:2004.10397.
- Bonawitz, K.; Ivanov, V.; Kreuter, B.; Marcedone, A.; McMahan, H.B.; Patel, S.; Ramage, D.; Segal, A.; Seth, K. Practical secure aggregation for privacy-preserving machine learning. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, Dallas, TX, USA, 30 October–3 November 2017; pp. 1175–1191.
- Zhang, C.; Li, S.; Xia, J.; Wang, W.; Yan, F.; Liu, Y. {BatchCrypt}: Efficient homomorphic encryption for {Cross-Silo} federated learning. In Proceedings of the 2020 USENIX annual technical conference (USENIX ATC 20), Boston, MA, USA, 15–17 July 2020; pp. 493–506.
- Truex, S.; Liu, L.; Chow, K.H.; Gursoy, M.E.; Wei, W. LDP-Fed: Federated learning with local differential privacy. In Proceedings of the Third ACM International Workshop on Edge Systems, Analytics and Networking, Heraklion, Greece, 27 April 2020; pp. 61–66.
- Hu, R.; Guo, Y.; Li, H.; Pei, Q.; Gong, Y. Personalized federated learning with differential privacy. *IEEE Internet Things J.* **2020**, *7*, 9530–9539. [[CrossRef](#)]
- He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 1–26 June 2016; pp. 770–778.

18. Ghazi, B.; Golowich, N.; Kumar, R.; Manurangsi, P.; Zhang, C. Deep learning with label differential privacy. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 27131–27145.
19. Wang, J.; Liu, Q.; Liang, H.; Joshi, G.; Poor, H.V. Tackling the objective inconsistency problem in heterogeneous federated optimization. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 7611–7623.
20. Huang, X.; Li, P.; Li, X. Stochastic Controlled Averaging for Federated Learning with Communication Compression. *arXiv* **2023**, arXiv:2308.08165.
21. Guha, N.; Talwalkar, A.; Smith, V. One-shot federated learning. *arXiv* **2019**, arXiv:1902.11175.
22. Kasturi, A.; Ellore, A.R.; Hota, C. Fusion learning: A one shot federated learning. In Proceedings of the Computational Science–ICCS 2020: 20th International Conference, Amsterdam, The Netherlands, 3–5 June 2020; Proceedings, Part III 20; Springer: Berlin, Germany, 2020; pp. 424–436.
23. Song, R.; Liu, D.; Chen, D.Z.; Festag, A.; Trinitis, C.; Schulz, M.; Knoll, A. Federated learning via decentralized dataset distillation in resource-constrained edge environments. In Proceedings of the 2023 International Joint Conference on Neural Networks (IJCNN), Gold Coast, Australia, 18–23 June 2023; pp. 1–10.
24. Dwork, C. Differential privacy. In Proceedings of the International Colloquium on Automata, Languages, and Programming, Venice, Italy, 10–14 July 2006; pp. 1–12.
25. Arachchige, P.C.M.; Bertok, P.; Khalil, I.; Liu, D.; Camtepe, S.; Atiquzzaman, M. Local differential privacy for deep learning. *IEEE Internet Things J.* **2019**, *7*, 5827–5842. [[CrossRef](#)]
26. Sachdeva, N.; McAuley, J. Data distillation: A survey. *arXiv* **2023**, arXiv:2301.04272.
27. Nguyen, T.; Chen, Z.; Lee, J. Dataset meta-learning from kernel ridge-regression. *arXiv* **2020**, arXiv:2011.00050.
28. Nguyen, T.; Novak, R.; Xiao, L.; Lee, J. Dataset distillation with infinitely wide convolutional networks. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 5186–5198.
29. Wang, T.; Zhu, J.Y.; Torralba, A.; Efros, A.A. Dataset distillation. *arXiv* **2018**, arXiv:1811.10959.
30. Sucholutsky, I.; Schonlau, M. Soft-label dataset distillation and text dataset distillation. In Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN), Shenzhen, China, 18–22 July 2021; pp. 1–8.
31. Zhao, B.; Bilen, H. Dataset Condensation with Gradient Matching. In Proceedings of the International Conference on Learning Representations (ICLR), Virtual Event, Austria, 3–7 May 2021.
32. Guo, Z.; Wang, K.; Cazenavette, G.; Li, H.; Zhang, K.; You, Y. Towards Lossless Dataset Distillation via Difficulty-Aligned Trajectory Matching. *arXiv* **2023**, arXiv:2310.05773.
33. Du, J.; Shi, Q.; Zhou, J.T. Sequential Subset Matching for Dataset Distillation. In Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), New Orleans, LA, USA, 10–16 December 2023.
34. Sajedi, A.; Khaki, S.; Amjadian, E.; Liu, L.Z.; Lawryshyn, Y.A.; Plataniotis, K.N. DataDAM: Efficient Dataset Distillation with Attention Matching. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 1–6 October 2023; pp. 17097–17107.
35. Zhao, G.; Li, G.; Qin, Y.; Yu, Y. Improved Distribution Matching for Dataset Condensation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 17 June–24 June 2023; pp. 7856–7865.
36. Cazenavette, G.; Wang, T.; Torralba, A.; Efros, A.A.; Zhu, J.Y. Generalizing Dataset Distillation via Deep Generative Prior. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 17 June–24 June 2023; pp. 3739–3748.
37. Wang, K.; Gu, J.; Zhou, D.; Zhu, Z.; Jiang, W.; You, Y. DiM: Distilling Dataset into Generative Model. *arXiv* **2023**, arXiv:2303.04707.
38. Yu, R.; Liu, S.; Wang, X. A Comprehensive Survey to Dataset Distillation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *46*, 17–32.
39. Xiong, Y.; Wang, R.; Cheng, M.; Yu, F.; Hsieh, C.J. Feddm: Iterative distribution matching for communication-efficient federated learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17 June–24 June 2023; pp. 16323–16332.
40. Dong, T.; Zhao, B.; Liu, L. Privacy for Free: How does Dataset Condensation Help Privacy? In Proceedings of the International Conference on Machine Learning (ICML), Baltimore, MD, USA, 17–23 July 2022; pp. 5378–5396.
41. Tsilivis, N.; Su, J.; Kempe, J. Can we achieve robustness from data alone? In Proceedings of the International Conference on Machine Learning (ICML), Workshop, Baltimore, MD, USA, 17–23 July 2022.
42. Liu, Y.; Gu, J.; Wang, K.; Zhu, Z.; Jiang, W.; You, Y. DREAM: Efficient Dataset Distillation by Representative Matching. *arXiv* **2023**, arXiv:2302.14416.
43. Bohdal, O.; Yang, Y.; Hospedales, T. Flexible dataset distillation: Learn labels instead of images. *arXiv* **2020**, arXiv:2006.08572.
44. Forgy, E.W. Cluster analysis of multivariate data: Efficiency versus interpretability of classifications. *Biometrics* **1965**, *21*, 768–769.
45. Sengupta, A.; Ye, Y.; Wang, R.; Liu, C.; Roy, K. Going deeper in spiking neural networks: VGG and residual architectures. *Front. Neurosci.* **2019**, *13*, 95. [[CrossRef](#)]
46. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [[CrossRef](#)]
47. Xiao, H.; Rasul, K.; Vollgraf, R. Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms. *arXiv* **2017**, arXiv:1708.07747.
48. Krizhevsky, A.; Hinton, G. *Learning Multiple Layers of Features from Tiny Images*; University of Toronto: Toronto, ON, Canada, 2009.

49. Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; Ng, A.Y. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*; NIPS: San Diego, CA, USA, 2011.
50. Gidaris, S.; Komodakis, N. Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4367–4375.
51. Amari, S.i. Backpropagation and stochastic gradient descent method. *Neurocomputing* **1993**, *5*, 185–196. [[CrossRef](#)]
52. Li, Q.; Diao, Y.; Chen, Q.; He, B. Federated learning on non-iid data silos: An experimental study. In *Proceedings of the 2022 IEEE 38th International Conference on Data Engineering (ICDE)*, Kuala Lumpur, Malaysia, 9–12 May 2022; pp. 965–978.
53. Zhang, J.; Chen, C.; Li, B.; Lyu, L.; Wu, S.; Ding, S.; Shen, C.; Wu, C. Dense: Data-free one-shot federated learning. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 21414–21428.
54. Lin, T.; Kong, L.; Stich, S.U.; Jaggi, M. Ensemble distillation for robust model fusion in federated learning. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 2351–2363.

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.