One-Shot Federated Learning based on Random Feature Extractor

Luyuan Yang[1] **Shayan Shafaei**[1] **Yiming Liu**[1] **Naeem Shahabi Sani**[1] **Yu Cai**[1] **Jun (Luke) Huan**[2] **Chao Lan**[1]

[1]School of Computer Science, University of Oklahoma, Norman, Oklahoma, USA
[2]AWS AI, USA

Abstract

The transition from multi-rounds federated learning to one-shot federated learning (OFL) markedly alleviates communication burden and represents a major step toward realistic deployment. Most existing OFL approaches require clients to perform local training, imposing a substantial computational burden on client side, while others rely on pre-trained models. In this paper, we propose a novel one-shot federated learning framework based on random feature extractor (FedRFE). Unlike existing approaches, it does not require any local model training or pre-trained model, featuring superior resource efficiency. Through comprehensive experiments, we show FedRFE achieves competitive performance while being robust in challenging scenarios including data heterogeneity and client scalability. Our code is available at`https://github.com/luyuanxyang/FedRFE`

1 INTRODUCTION

Federated learning is a distributed and privacy-preserving machine learning paradigm [Kairouz et al., 2021]. It has broad applications in domains like finance and healthcare, which are further advanced by the large-scale deployments of machine learning on smart devices [Pfeiffer et al., 2023]. Traditional federated learning approaches require repeated communications between clients and server, which not only limits learning efficiency [McMahan et al., 2017] but also raises privacy concerns [Amato et al., 2025]. A compelling solution is one-shot federated learning (OFL) [Guha et al., 2019], which restricts communication to a single round.

Although OFL effectively alleviates communication burden, some challenges remain. In particular, a notable challenge is client-side resource constraints. In a survey of federated learning for computationally constrained devices, [Pfeiffer et al., 2023] compare the memory required to train deep models with the memory available on widely used edge devices, as shown in Table 1. Their analysis indicates that many edge devices lack sufficient memory to train even a single model. Moreover, local training can become prohibitively time-consuming on devices without powerful processors, such as GPUs, as shown in Table 2.

Table 1: Memory Costs and Capacities [Pfeiffer et al., 2023]

Model	Training Memory	Device	RAM
ResNet18	0.8 GB	STM32H5	32 $\sim$ 640 KB
MobileNet	1.4 GB	STM32N6	4.2 MB
LeNet	0.5 GB	Raspberry Pi	256 MB $\sim$ 32 GB

Table 2: Example Training Time of 100 epochs on MNIST with AMD EPYC 7003 CPU and NVIDIA L40 GPU

Models	**Batch Size**	**CPU Time**	**GPU Time**
MobileNet	32	6 h 38 min	1 h 35 min
ResNet-18	32	12 h 36 min	1 h 7 min

Although resource constraint has drawn increasing attentions in federated learning research, it has not been broadly tackled in OFL. Current OFL approaches can be roughly categorized into two lines. One line follows the framework in which clients train models locally and upload the resulting models to the server for aggregation.[Guha et al., 2019, Zeng et al., 2024, Dai et al., 2024, Zhang et al., 2022, Allouah et al., 2024, Zeng et al., Jhunjhunwala et al., 2024]. This line does not account for local resource constraint.

Another line focuses on obfuscating local data before uploading them to the server for model training [Zhou et al., 2020, Zhang et al., 2024, Song et al., 2023, Guan et al., 2025, Beitollahi et al., 2024]. Some approaches still require local training of the obfuscation model such as those employing the distill technique [Zhou et al., 2020, Zhang et al.,

Proceedings of the 42nd Conference on Uncertainty in Artificial Intelligence (UAI 2026), PMLR 337:7737–7750.

2024, Song et al., 2023]. There are approaches that avoid any local training, by presuming the availability of a relevant pre-trained model [Guan et al., 2025, Beitollahi et al., 2024]. Although these approaches generally achieve SOTA performance, their applicability is often limited by the dependence on pre-trained models. In addition, global model may be highly biased when the pre-trained model does not well align with the target task [Liu et al., 2025].

Beyond this issue, other problems also pose significant challenges to the practical deployment of OFL. For example, the *data heterogeneity* problem arises as data across clients are drawn from different distributions, and the *client scalability* becomes a concern when a large number of clients participate. These challenges can degrade model performance, particularly when model training can only rely on a single round of client-server communication.

In this paper, we propose a new OFL framework based on random feature extract (FedRFE). It does not require any local training or pre-trained models. Instead, it only relies on some randomly generated feature extractors, and is thus extremely resource-efficient. Its accuracy is generally comparable to those of the SOTA approaches (which rely on pretrained models), and even higher by a large margin on some data set. FedRFE also turns out to be fairly robust in other challenging scenarios including data heterogeneity and client scalability. We conduct comprehensive experiments to verify its effectiveness and superiority.

2 RELATED WORK

2.1 ONE-SHOT FEDERATED LEARNING

In the pioneering OFL [Guha et al., 2019], clients conduct local model training and upload well-trained local models to the server as the communicated information. Most existing approaches follow this paradigm and can be broadly categorized into three groups: Generative method [Mendieta et al., 2025, Yang et al.], Ensemble method [Zeng et al., 2024, Dai et al., 2024, Zhang et al., 2022, Allouah et al., 2024, Zeng et al.] and Bayesian method [Jhunjhunwala et al., 2024, Liu et al., 2024, Hasan et al., 2024]. Although this paradigm has drawn considerable research interest, their reliance on substantial local training imposes a heavy computational burden on clients, especially resource-constrained ones. Additionally, the global model may suffer from performance degradation as the number of clients or the degree of data heterogeneity increases.

Besides this model-transmission paradigm, some OFL frameworks explore alternative forms of information. A set of studies propose to transmit distilled synthetic data to avoid performance degradation [Zhou et al., 2020, Zhang et al., 2024, Song et al., 2023]. However, distillation brings extra computation costs to clients and thus fails to alleviate the computational burden. Another set of studies choose to upload descriptions of local data [Chen et al., 2025, Yang et al., 2024a, Zaland et al., 2025, Yang et al., 2024b]. In these works, the server generates synthetic samples from the uploaded descriptions and uses them to train a global classifier. Although these methods avoid fully training process, some still require optimization of descriptions and computation of large pre-trained generative models, which still suffer from high computation costs. Moreover, the high-fidelity synthetic data produced by the server raises extra privacy concerns Carlini et al. [2023].

Recently, [Beitollahi et al., 2024, Guan et al., 2025] explored uploading statistics of features extracted by pretrained feature extractors. [Beitollahi et al., 2024] propose to formulate the global model as a concatenation of a pre-trained feature extractor and a head classifier. To be specific, each client uploads class-conditional approximation of extracted features, and the server reconstructs the original features to train the head classifier. [Guan et al., 2025] follows this design, but eliminates the approximation and reconstruction step. Specifically, clients compute local feature statistics from extracted features and upload them to the server. The server then aggregates them into global feature statistics and uses them to configure a Bayesian classifier without training. These two frameworks provide a promising solution to above challenges, yet are built upon pre-trained extractors.

2.2 CNNS WITH RANDOM WEIGHTS

The effectiveness of CNNs with random weights has been validated from both empirical and theoretical perspectives.

[Jarrett et al., 2009] empirically showed that a two layer convolutional architecture with random filters and a supervised classifier can achieve strong performance for image classification. Later, [Saxe et al., 2011] theoretically proves the frequency selectivity and translation invariance properties of the random convolutional pooling architecture. Moreover, it also suggests the superiority of certain methods may primarily originate from the architectural design. [Gilbert et al., 2017] also confirmed the competitive performance of CNN with two layers of gaussian random convolution and a supervised classifier on CIFAR-10. The paper further indicates that the extracted features by CNNs with random filters can provide informative representations of the input data and do not contradict those extracted by CNNs with learned filters. Notably, [Cao and Wu, 2022] discovers the surprising phenomenon "Tobias" that a randomly initialized CNN without learning has superior localization ability, and ReLU activation and network depth are important contributors to this ability. With the popularity of lottery ticket hypothesis [Frankle and Carbin, 2018] : a randomly initialized neural network contains a sub-network such that, when both networks trained in isolation for similar numbers of iterations, their performance is comparable, [da Cunha et al.,

2022] extends it to convolutional neural networks. It derives theoretical and experimental results to show that it is feasible to approximate any target CNN with high probability by pruning a randomly initialized CNN of logarithmically larger size and twice the depth.

3 METHODOLOGY

3.1 PROBLEM SETTING

Our problem setting is regular and focuses on image data. Let $\mathcal{X} \subseteq \mathbb{R}^{M \times D}$ be the data space of images, where each image is described by M channels and D features (i.e., the image is flattened into a vector of dimension D). Suppose there are c classes and let $\mathcal{Y} = \{1, 2, \ldots, c\}$ be the label set.

Suppose there are k clients, and client i possesses a private data set $D_i = \{(x_j, y_j) \in \mathcal{X} \times \mathcal{Y}\}_{j=1}^{N_i}$ of N_i points. Suppose there is a server that can communicate with all clients and learn models. The general goal of OFL is to learn a global model at the server based on one round of client-server communication that preserves client data privacy.

We aim to design an OFL framework that avoids any local training or use of pretrained model. Our design is inspired by the success of random CNN technique [Jarrett et al., 2009, Cao and Wu, 2022].

Our basic idea is to broadcast the random CNNs to clients, who then use them to generate obfuscated data that are uploaded to the server for training the downstream head classifier. The key of this design is the broadcast CNNs, which we consider as random feature extractors. In the following, we discuss the lessons learned from random CNN which leads to our design of these extractors. At the end, we fill in other details and explain the complete algorithm.

3.2 LESSONS FROM RANDOM CNN

It is known that random CNN can extract useful features for building strong classifiers. However, the extracted features also retain information that may allow one to reasonably reconstruct input data, which raises privacy and security concern. This concern is formally studied in [Gilbert et al., 2017], which analyzes the invertibility of random CNN in terms of the reconstruction error. In this section, we review the result and discuss related insights that lead to our design.

Consider the following Restricted Isometry Property (RIP) defined for any vector set and associated distortion factor.

Definition 3.1. *For any $\mathcal{M} \subseteq \mathbb{R}^q$, a matrix $\Phi \in \mathbb{R}^{p \times q}$ is said to satisfy $\mathcal{M}$-RIP if there is a constant $\delta > 0$ such that*

$$(1-\delta)||z||^2 \leq ||\Phi z||^2 \leq (1+\delta)||z||^2, \tag{1}$$

for all $z \in \mathcal{M}$, where $||\cdot||$ is ℓ_2 norm. The smallest δ that satisfies (1) is called the distortion factor of $\mathcal{M}$.

Let $\mathcal{M}_k \subseteq \mathbb{R}^p$ be the set of vectors with exactly k non-zero entries, and $\mathcal{M}_k^2 \subseteq \mathbb{R}^p$ be the set of vectors where every member can be expressed as $x = \alpha_1 x_1 + \alpha_2 x_2$ for some $x_1, x_2 \in \mathcal{M}_k$ and $\alpha_1, \alpha_2 \in \mathbb{R}$. Here, we omit p in set notations to indicate it is flexible e.g., if we write $y \in \mathcal{M}_k$ for $y \in \mathbb{R}^{p'}$, then $p = p'$. By design there is $\mathcal{M}_k \subseteq \mathcal{M}_{2k}$, which further implies the following.

Remark 3.2. *If a matrix $\Phi \in \mathbb{R}^{p \times q}$ satisfies $\mathcal{M}_k^2$-RIP with a distortion factor δ_{2k}, then the matrix also satisfies $\mathcal{M}_k$-RIP with a distortion factor δ_k and $\delta_k \leq \delta_{2k} < 1$.*

Proof. The proof just follows definitions, and we only elaborate the case with upper bound. Suppose Φ satisfies $\mathcal{M}_k^2$-RIP, then for any $z \in \mathcal{M}_k^2$, there is $||\Phi z||^2 \leq (1+\delta_{2k})||z||^2$. Since $\mathcal{M}_k \subseteq \mathcal{M}_k^2$, the above also holds for all $z \in M_k$. This means Φ also satisfies $\mathcal{M}_k$-RIP with some distortion factor δ_k. For this implied RIP, since δ_k is the smallest value that supports the uniform bound (whereas δ_{2k} is only one possible value), we have $\delta_k \leq \delta_{2k}$. □

Let $\mathcal{C}(W)$ be a single-layer CNN model with the following structure. Its input vector $x \in \mathbb{R}^{MD}$ contains data from M channels, each of dimension D. It has a single convolution layer that uses K filters, each of dimension ℓ, and stride length t so the number of filter shifts is $n = (D-\ell)/t + 1$. Let $W \in \mathbb{R}^{Kn \times MD}$ be its weight matrix for its filters, which maps x to a convolution output $z(x) = Wx \in \mathbb{R}^{Kn}$, followed by a ReLU activation function and then a pooling function $\mathbb{M}$ that outputs a vector $\hat{z}(x) = \mathbb{M}(z(x), k) \in \mathbb{R}^{Kn} \cap \mathcal{M}_k$. It is verified in prior study that $\mathbb{M}$ includes max-pooling. Finally, a regular MLP classifier is applied.

[Gilbert et al., 2017] present two results for model $\mathcal{C}(W)$. First, if the model uses Gaussian random filters, then W^T satisfies $\mathcal{M}_k^2$-RIP with distortion factor δ_{2k} with high probability, providing (Theorem 3.1 & Corollary 3.2)

$$\frac{\ln(Kn)}{\delta_{2k}^2} \leq g(M, D, \ell, k, n), \tag{2}$$

where g is a function of other model hyper-parameters. Note this implies $\delta_{2k} = \Omega(\sqrt{\ln(Kn)})$.

Second, if W is $\mathcal{M}_k^2$-RIP with distortion factor δ_{2k}, then for any input x with $||x|| \neq 0$, its reconstructed $\hat{x} = W^T \hat{z}(x)$ based on hidden unit $\hat{z}(x)$ satisfies (Theorem 3.3)

$$||x - \hat{x}|| \leq \frac{5\delta_{2k}}{1-\delta_k} \frac{\sqrt{1+\delta_{2k}}}{\sqrt{1-\delta_{2k}}} ||x||, \tag{3}$$

where δ_k is the distortion factor of the implied RIP in Remark 3.2. Combining this with (2), it can be verified that when other hyper-parameters are fixed and input has a bounded norm, the reconstruction error satisfies

$$||x - \hat{x}|| = O(1/\sqrt{\ln(Kn)}), \tag{4}$$

for small δ_{2k}.

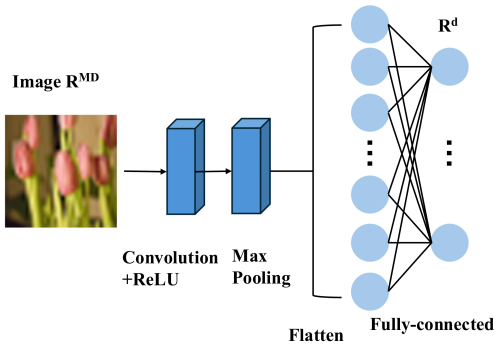


Figure 1: **Architecture of a Feature Extractor** ϕ_i

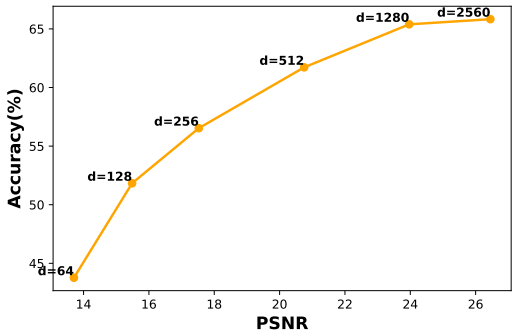


Figure 2: Accuracy-Privacy Tradeoff based on d

3.3 RANDOM FEATURE EXTRACTOR DESIGN

Our feature extractor is a subnetwork of random CNN that contains both the random convolution layer and one linear layer. This linear layer also adopts random weights to map the hidden unit $\hat{z}$ from $\mathbb{R}^{Kn}$ dimensional space to a lower d-dimensional space. Figure 1 illustrates a feature extractor.

We examine invertibility of the designed extractor. Let $R \in \mathbb{R}^{d\times Kn}$ be the projection matrix at the linear layer, whose entries are sampled i.i.d. from $O(0, 1/d)$. Let $\hat{v}(x) = R\hat{z}(x) \in \mathbb{R}^d$ be the projected vector and $\tilde{x} = R^T\hat{v}(x)$ be the reconstructed vector. Our observation is as follows.

Remark 3.3. *For $\mathcal{C}(W)$, if the model uses Gaussian random filters and both W and $\hat{z}(x)$ have bounded operator norms, then with high probability,*

$$\min_{U\in\mathbb{R}^{MD\times d}} ||x - U\hat{v}(x)|| = \tilde{O}(Kn/d). \tag{5}$$

Proof. Begin with a decomposition:

$$\min_U ||x - U\hat{v}(x)|| \le ||x - \hat{x}|| + \min_U ||\hat{x} - U\hat{v}(x)||. \tag{6}$$

The first term can be bounded by (4). The second term can be bounded as follows:

$$\begin{aligned}\min_U ||\hat{x} - U\hat{v}(x)|| &= \min_U ||W^T\hat{z}(x) - UR\hat{z}(x)|| \\ &\le ||W^T\hat{z}(x) - W^TR^TR\hat{z}(x)|| \\ &\le ||W||\,||\hat{z}(x)||\cdot||I - R^TR|| \\ &= O(\max_i |\sigma_i^2(R) - 1|),\end{aligned} \tag{7}$$

where σ_i is the i_{th} largest singular value of R (including ties); the first inequality holds with $U = W^TR^T$ and the last line holds for bounded operator norms of W and $\hat{z}(x)$. Since R has entries i.i.d. from $N(0, 1/d)$, classic random matrix theory [Vershynin, 2012, Corollary 5.35] implies

$$\sigma_i(R)^2 \le Kn/d + 1 + o(1) \tag{8}$$

with high probability. Plugging this back to the above shows

$$\min_U ||\hat{x} - U\hat{v}(x)|| = O(Kn/d). \tag{9}$$

Plugging this back to (6) and noting that the first term has a lower-order bound yields the remark. □

We have some interesting observations from the remark.

First, it suggests the invertibility of our feature extractor is dominated by the linear layer, since the convolution layer introduces way smaller error $O(\sqrt{\ln(Kn)})$ that is overruled by the $O(Kn/d)$ error introduced at the linear layer. This aligns with a common sense that, the convolution layer extracts useful information while linear layer mainly preserves the extracted information. For this reason, in practice we suggest fix a reasonable Kn and focus on tuning d.

Second, it suggests our feature extractor is less invertible when d is small or Kn is large. It is well-known that smaller d typically implies lower model accuracy. Thus, the remark establishes an accuracy-privacy tradeoff balanced by d. Figure 2 demonstrates this trade-off on real-world data, with detailed setup explained in Section 5. Note the x-axis shows reconstruction accuracy (PSNR), and smaller PSNR means more privacy. As d varies, we see the model either gains more privacy but less accuracy (e.g. $d = 64$) or higher accuracy but less privacy (e.g., $d = 1280$). This confirms the existence of the accuracy-privacy tradeoff.

Figure 3 provides a more intuitive result of the tradeoff. It shows an original image and its reconstructions from the extractor output $\tilde{x}$; reconstructed images are obtained using a very popular feature inversion technique [Ulyanov et al., 2018], with detailed configuration explained in Section 5. We see smaller d provides less meaningful reconstructions, and $d = 256$ seems like a good threshold in both figures.

The impact of Kn on input data reconstruction is twofold. For the convolution layer that maps MD-dimensional input data into Kn-dimensional data, larger Kn implies that more information is preserved and hence reconstruction is easier (lower error in (4)). For the linear layer that further maps Kn-dimensional data into d-dimensional data, larger KN implies that more information are compressed and hence reconstruction is harder (higher error in (9)). Overall, error incurred at the linear layer dominates so the reconstruction

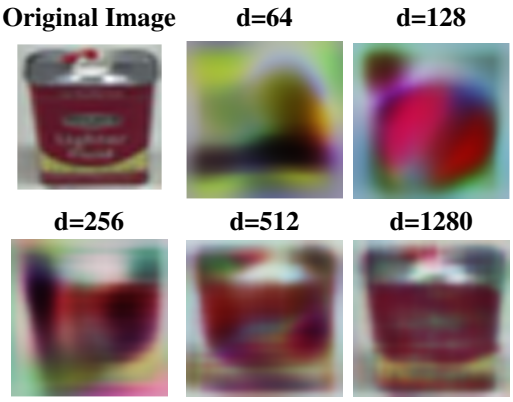


Figure 3: Reconstruction Attack Results on CIFAR100

Algorithm 1 One-Shot Federated Learning based on Random Feature Extractor (FedRFE)

1: Server: randomly sample a set of random feature extractors $\phi_1, \phi_2, \ldots, \phi_m$, each mapping from the input domain $\mathbb{R}^{MD}$ to the hidden unit domain $\mathbb{R}^d$.
2: Server: broadcast set $\{\phi_i\}$ to each of k clients.
3: **for** $i = 1, 2, \ldots, k$ in parallel **do**:
Client: client i runs algorithm *Extraction(D_i)* and uploads algorithm output $\tilde{D}_i$ to the server.
4: **end for**
5: Server: train a head classifier g based on $\tilde{D}_i$'s.
6: **return** Global model $f = g \circ (\phi_1, \phi_2, \ldots, \phi_m)$.

error increases as Kn increases.

3.4 FRAMEWORK DESIGN

In this section, we describe the complete design of FedRFE and its learning algorithm.

Let $\phi : \mathbb{R}^{MD} \to \mathbb{R}^d$ be a random feature extractor designed in Section 3.3. Recall it has a random convolution layer plus a max-pooling layer, denoted as $\mathcal{C}(W) : \mathbb{R}^{MD} \to \mathbb{R}^{Kn}$; then, $\mathcal{C}$ is followed by a linear mapping from $\mathbb{R}^{Kn}$ to $\mathbb{R}^d$.

Since a single feature extractor may provide limited information for classification, we propose the server broadcast a collection of m extractors $\{\phi_1, \phi_2, \ldots \phi_m\}$ to the clients to collect more extracted information for boosting model performance in an ensemble fashion [Dietterich, 2000, Zhou, 2025]. For each data point x, its features are generated by these extractors and then stacked into a higher dimensional vector $[\phi_1(x); \ldots; \phi_m(x)] \in \mathbb{R}^{md}$. Finally, after the server receives uploaded data from all clients, it standardly learns an MLP classifier as the head classifier $g : \mathbb{R}^{md} \to \mathbb{R}^c$. When learning is done, the head classifier is attached to the random feature extractors to form the global model $f = g \circ (\phi_1, \ldots, \phi_m)$. The complete FedRFE learning framework is provided in Algorithm 1 2.

Algorithm 2 The $Extraction(D_i)$ Protocol

1: **for** each point $(x_j, y_j) \in D_i$ **do**:
2: **for** $l = 1, 2, \ldots, m$ **do**: Compute $\phi_l(x_j) \in \mathbb{R}^d$
3: **end for**
4: $s_j \leftarrow [\phi_1(x_j); \phi_2(x_j); \ldots; \phi_m(x_j)] \in \mathbb{R}^{md}$
5: **end for**
6: **return** $\tilde{D}_i = \{(s_j, y_j)\}_{j=1}^{N_i}$.

4 EXPERIMENT

4.1 SETUP

Datasets. We experiment on three real-world datasets that are commonly used in federated learning research i.e., CIFAR10 (C10) [Krizhevsky et al., 2009], CIFAR100 (C100) [Krizhevsky et al., 2009] and SVHN [Netzer et al., 2011].

Data Heterogeneity. We follow a common practice Zeng et al., 2024], Li et al. [2022] to simulate data heterogeneity in reality. The basic idea is that data sets at different clients have different label distributions independently sampled from a Dirichlet distribution. More specifically, for each class c, the posterior probability p_c is sampled from $Dir(\alpha)$ independently for each client. In general, smaller α yields higher degree of heterogeneity across clients.

Baselines. This study focuses on resource-constrained environment for one-shot federated learning. Directly related baselines that address the same problem are FedPFT [Beitollahi et al., 2024] and FedCGS [Guan et al., 2025], which assumes pre-trained generative models are available. For comprehensiveness, we also compare with the SOTA techniques that adopt local model training, which are FAFI Zeng et al. and IntactOFL [Zeng et al., 2024] and Co-Boosting[Dai et al., 2024]. Finally, the vallina FedAvg [McMahan et al., 2017, Guha et al., 2019] is also included as a baseline. Like all above studies, for fair comparison we exclude the techniques built on pre-trained generative models [Zeng et al., 2024, Guan et al., 2025, Zeng et al.].

General Configuration. We evaluate a broad range of client number k, from 5 to 120, and report the main comparison results based on $k = 20$. To simulate a resource-constrained local environment, in our main results, all baselines requiring local model training are restricted to exactly 100 epochs of local model updates with the standard 0.01 learning rate [Dai et al., 2024]. At the end, we also compare with these baselines without such restriction. Finally, we choose the most common ResNet-18 as base model for all baselines; for FedPFT and FedCGS, the pre-trained ResNet-18 is used as their pretrained models. All reported results are averaged over three random seeds.

Configuration of FedRFE.

We set the hyper-parameters for FedRFE as follows.

Table 3: Accuracy (%) of FedFRE based on Different m and Convolution Layer Number.

Dataset	# Conv.Layers	0	1	2
C10	(m=1,d=256)	48.65 ± 0.3	56.53 ± 0.1	54.21 ± 0.6
	(m=5,d=256)	55.34 ± 0.1	65.33 ± 0.2	63.57 ± 0.1
	(m=10,d=256)	55.45 ± 0.1	66.65 ± 0.6	$\mathbf{65.96 \pm 0.1}$
	(m=15,d=256)	54.19 ± 0.9	$\mathbf{66.81 \pm 0.3}$	64.56 ± 0.0
	(m=20,d=256)	53.83 ± 0.4	64.36 ± 0.39	64.85 ± 0.1
	(m=1,d=2560)	$\mathbf{55.58 \pm 0.3}$	65.83 ± 0.31	63.35 ± 0.8
C100	(m=1,d=256)	23.47 ± 0.4	28.80 ± 0.6	25.31 ± 0.1
	(m=5,d=256)	28.93 ± 0.2	37.36 ± 0.5	35.53 ± 0.2
	(m=10,d=256)	$\mathbf{29.19 \pm 0.3}$	$\mathbf{39.80 \pm 0.0}$	37.74 ± 0.3
	(m=15,d=256)	28.84 ± 0.2	38.88 ± 0.8	$\mathbf{39.02 \pm 0.2}$
	(m=20,d=256)	27.59 ± 0.4	39.00 ± 0.4	38.98 ± 0.4
	(m=1,d=2560)	28.87 ± 0.2	37.89 ± 0.4	36.07 ± 0.3

Table 4: Accuracy (%) of FedFRE based on Head Classifiers g with Different Number of Hidden Layers

Dataset	classifier	Accuracy
CIFAR10	1-layer MLP	61.65 ± 0.07
	2-layer MLP	$\mathbf{66.65 \pm 0.62}$
	3-layer MLP	62.12 ± 1.52

Figure 2 presents the relationship between reconstruction similarity PSNR and the test accuracy of global model consisting of a single feature extractor ϕ_l and a 2-layer MLP with varying output dimension d of ϕ_l. Then, Figure 3 shows the exact reconstructed image of ϕ_l with changing d. From both images, we can conduct the optimal output dimension of single random feature extractor is $d = 256$ for balancing accuracy and privacy. We also conduct experiments to find the best structure for FedRFE by varying convolution Layer number and extractor number in table 3.Number of feature extractors in the ensemble is set as $m = 10$ based on Table 3 (where $\alpha = 0.3$ and $k = 20$) as it generally leads to optimal performance. Number of convolution layer is set as 1 based on Table 3, for it yields optimal performance in most cases. In this table, the top row shows three candidate layer numbers 0, 1, 2. Number 0 means no convolution is performed, so the feature extractor is simply a linear random projection. Number of layers for the head MLP classifier is set as 2 based on Table 4 (based on $\alpha = 0.3$ and $k = 20$). We set the number of output channels to 64, vary the kernel size among 3 , 5 and 7 , and vary the stride between 1 and 2. The head MLP classifier is trained with cross-entropy loss.

4.2 MAIN COMPARISON RESULTS

Table 5 presents the classification performance of all methods with 20 clients. We have several observations.

First, our FedRFE achieves the highest or second highest accuracy in most cases, especially when clients have high data heterogeneity. This demonstrates its superior and robust performance as a new one-shot federated learning framework.

Second, all methods that upload obfuscated data (FedPFT, FedCGS, FedRFE) are fairly robust across all heterogeneity degrees, compared to baselines that upload local models (FAFI, IntactORF, etc). This aligns with the observation in prior studies [Guan et al., 2025] that presents feature-based methods as a promising direction to tackle data heterogeneity in one-shot federated learning.

Third, compared with SOTA feature-based methods FedCGS and FedPFT, FedRFE achieves higher accuracy on most data sets, especially by a large margin on SVHN. Even on the third data set it performs closely to the optimum. Most importantly, FedFRE excels without using any (expensive) pre-trained model like the other two methods. These establishes FedRFE as a new SOTA method for one-shot federated learning that features extreme resource efficiency.

4.3 IMPACT OF RESOURCE CONSTRAINT

To further evaluate the impact of resource constraint on different methods, we gradually relax the restriction on local model training epochs. Figure 4 shows classification accuracy with five clients. In each figure, the x-axis shows the number of epochs used for model training. The limit is set to 200, which is reported by the latest FAFI as a sufficient number for convergence.

We see as the epoch number increases, methods that rely on local model training generally gain higher accuracies. Among them, the latest FAFI seems to gain the fastest improvement rate but still falls behind feature-based methods under high data heterogeneity e.g., $\alpha = 0.01$ or 0.05. In general, the result suggests that existing local-training methods are extremely sensitive to resource constraints and feature-based methods are not only naturally robust but can often provide competitive performance especially when data heterogeneity is large.

We also evaluate the computation time at each client. Results are shown in Table 6. We see FedPFT, FedCGS and our method all consume significantly less time. This substantial time savings make these three methods the better choice for realistic deployment, comparing with the hours of training process of other methods.

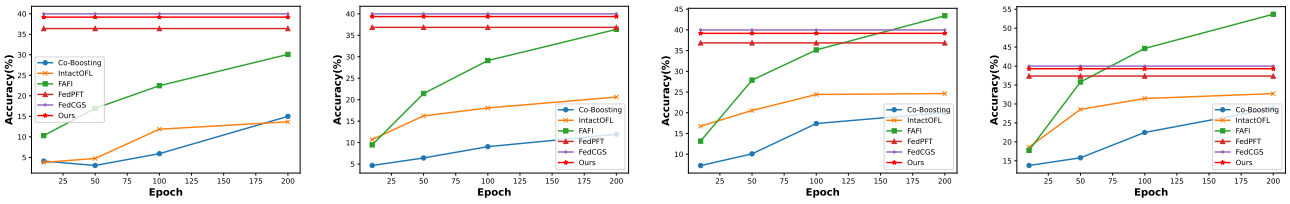
In the rest experiment, we focus on the comparative performance of FedRFE with unconstrained local resources. To this end, local-training epoch number is set to 200.

4.4 IMPACT OF HETEROGENEITY

In this section, we focus on evaluating the impact of data heterogeneity from two perspectives. First, we rerun the comparison experiment with increased local training epochs.

Table 5: Classification Accuracy (%) of All Methods

Dataset	α	FedAvg (one-shot)	Co-Boosting	IntactOFL	FAFI	FedPFT	FedCGS	Ours
CIFAR10	0.01	15.72 ± 0.87	37.03 ± 0.21	13.87 ± 0.35	38.51 ± 0.88	56.07 ± 0.27	63.95 ± 0	**66.51 ± 0.09**
	0.05	15.30 ± 0.42	50.32 ± 0.93	49.01 ± 0.74	57.22 ± 0.37	56.63 ± 0.47	63.95 ± 0	**66.94 ± 0.41**
	0.10	31.86 ± 2.81	65.32 ± 1.40	57.75 ± 0.34	**67.62 ± 1.61**	56.62 ± 0.09	63.95 ± 0	66.99 ± 0.14
	0.30	50.80 ± 3.15	68.12 ± 1.68	65.41 ± 0.82	**79.83 ± 0.41**	57.71 ± 0.21	63.95 ± 0	66.65 ± 0.62
CIFAR100	0.01	4.00 ± 0.26	14.08 ± 0.91	11.57 ± 0.85	20.54± 1.71	36.67 ± 0.16	**39.95 ± 0**	39.10 ± 0.04
	0.05	7.98 ± 0.43	21.33 ± 0.85	18.61 ± 1.15	26.57 ± 0.83	37.50 ± 0.13	**39.95 ± 0**	39.22 ± 0.13
	0.10	10.62 ± 0.87	32.62 ± 1.05	23.35 ± 1.17	30.79 ± 1.09	37.83 ± 0.08	**39.95 ± 0**	39.40 ± 0.14
	0.30	18.44 ± 0.36	38.74 ± 1.21	31.98 ± 0.18	**40.19 ± 0.68**	38.92 ± 0.36	39.95 ± 0	39.80 ± 0.03
SVHN	0.01	15.93 ± 1.89	37.20 ± 2.40	22.04 ± 0.69	36.46 ± 0.64	42.58 ± 0.34	57.77 ± 0	**81.01 ± 0.34**
	0.05	18.32 ± 0.92	64.05 ± 2.33	65.69 ± 1.33	74.72 ± 0.95	43.17 ± 0.23	57.76 ± 0	**80.53 ± 0.10**
	0.10	45.30 ± 1.20	67.26 ± 0.29	**81.94 ± 0.81**	76.22 ± 1.28	43.64 ± 0.56	57.77 ± 0	80.77 ± 0.53
	0.30	50.32 ± 0.63	73.97 ± 1.13	**88.25 ± 0.43**	85.70 ± 1.45	44.87 ± 0.42	57.76 ± 0	80.35 ± 0.24



(a) α=0.01 (b) α=0.05 (c) α=0.1 (d) α=0.3

Figure 4: Impact of the Number of Training Epochs on Classification Performance on CIFAR-100.

Table 6: Computation Time at Each Client on CIFAR-100 based on with AMD EPYC 7003 and NVIDIA L40

Methods	**Models**	**Batch Size**	**Local Training Epoch**	**CPU Time**	**GPU Time**
FedAvg (one-shot)	ResNet-18	256	100	4 h 13 min	7 min
Co-Boosting	ResNet-18	256	100	48 min	13 min
IntactOFL	ResNet-18	256	100	54 min	7 min
FAFI	ResNet-18	256	100	20 h 19 min	32 min
FedPFT	ResNet-18	256	Non	11 s	6 s
FedCGS	ResNet-18	256	Non	10 s	3 s
Ours	Random CNN	256	Non	1 min 40 s	7 s

Results are shown in Table 7. We observe a similar pattern i.e., feature-based methods remain robust across all heterogeneity degrees, and FedRFE offers competitive performance. For local-training methods like FAFI, we see its performance gains significant improvements e.g., accuracy is about 10% higher than in resource-constrained environment (Table 5).

Next, we evaluate all methods under another type of data heterogeneity based on quantity-based label partition [Li et al., 2022, Dai et al., 2024]. In this setting, each client only possesses data from C number of classes out of total c classes, which makes global learning very challenging. None of our SOTA baseline methods have tested this scenario. Our experimental results are shown in Table 8. First, we observe all performance improves as C increases, which is somewhat expected. Surprisingly, the proposed FedRFE outperforms all SOTA baselines (including feature-based methods) by a large margin and across all choices of C, featuring its outstanding robustness under extreme heterogeneity.

Table 7: Accuracy (%) with Five Clients on CIFAR100

	IntactOFL	FAFI	FedPFT	FedCGS	Ours
α=0.01	13.67	30.10	36.40	**39.98**	39.18
α=0.05	20.60	36.38	36.84	**39.99**	39.36
α=0.1	24.64	**43.40**	36.86	39.98	39.16
α=0.3	32.73	**53.68**	37.36	39.98	39.30

Table 8: Accuracy (%) with Five Clients on SVHN

	IntactOFL	FAFI	FedPFT	FedCGS	Ours
C=2	43.69	31.70	33.44	19.59	**70.78**
C=3	65.49	53.89	37.90	51.65	**75.97**
C=5	67.08	53.85	43.13	57.34	**76.14**

4.5 LARGE NUMBER OF CLIENTS

In this section, we evaluate the performance of all methods as client number increases. This is an important scenario for deploying federated learning techniques, as many real-world problems include a large number (e.g. millions) of clients [Kairouz et al., 2021, Caldas et al., 2018]. The scenario becomes particularly challenging with data heterogeneity, which would rapidly accumulate over massive clients causing substantial performance degeneration.

Results are shown in Table 9. We see feature-based methods remain robust as client number increases, and FedRFE offers near-optimal performance compared to the leading methods. This suggests them a better option for large-scale one-shot federated learning. Comparatively, local-training methods have rapid degenerating performance.

4.6 EFFECTS OF PRE-TRAINED MODELS

Our experiments evaluates the dependence of data-uploading methods FedCGS and FedPFT on pre-trained models. To this end, we replace their pre-trained models with a random model (created by replacing the ResNet-18 feature extractor trained on ImageNet with a randomly initialized ResNet-18). The replaced models are called FedPFT.r and FedCGS.r accordingly.

Table 10 shows the results. We see FedCGS and FedPFT performance substantially degrade after pre-trained models are replaced with random ones. This indicate these methods rely heavily on pre-trained models, which limit their applicability in some real-world scenarios. For example, the performance may deteriorate significantly when the pre-train dataset does not align with the target task in specific industries[Liu et al., 2025]. Comparatively, the proposed FedRFE relies solely on random models and thus is much cheaper and more broadly applicable.

5 DISCUSSION

We discuss the feature privacy of our framework. We consider two scenarios that server acts as an attacker: 1). server uses all functions $\{\phi_1, \phi_2, \ldots, \phi_{10}\}$ to reconstruct the private data with uploaded features $[s_1, \ldots, s_{10}]$, 2). server correctly locates the feature outputs and corresponding extractors and uses one ϕ_i to reconstruct the private data with corresponding features s_i. We utilize the popular feature inversion technology [Ulyanov et al., 2018] to simulate above attacks. For comparison, we use the first layer of pre-trained ResNet-50 and a flatten layer to get the flattened raw feature of input data and present the reconstruction image.

Table 9: Accuracy (%) with $\alpha = 0.3$ on CIFAR-100

k	IntactOFL	FAFI	FedPFT	FedCGS	Ours
5	32.73	**53.68**	37.31	39.96	39.09
20	37.29	**52.25**	38.76	39.96	39.80
50	33.03	**44.66**	39.91	39.96	39.21
100	28.8	36.56	39.83	**39.97**	39.64
120	27.3	34.13	39.56	**39.99**	39.29

Table 10: Accuracy (%) with 20 Clients on CIFAR-10

α	FedPFT.r	FedPFT	FedCGS.r	FedCGS	Ours
0.01	10.57	56.07	17.48	63.95	**66.51**
0.05	11.88	56.63	17.91	63.95	**66.94**
0.1	10.00	56.62	18.39	63.95	**66.99**
0.3	10.33	57.71	17.88	63.95	**66.65**

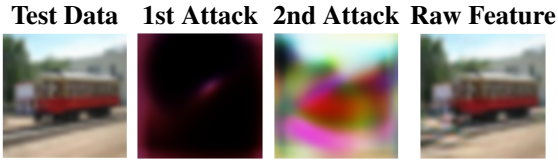


Figure 5: Results of reconstruction attacks on CIFAR100.

We randomly select 10 images in CIFAR100 and report the reconstruction images 6 7 and mean PSNR value 11 of all images for raw features and 2nd Attack reconstruction. For first attack, as the stacked vector destroyed the original data structure, it's impossible to reconstruct any useful information from the whole feature vector. For second attack, it is hard to visually identify the original image's class or content. The PSNR value further validates this. This demonstrates that FedRFE can satisfy strict privacy requirements.

6 CONCLUSION

In this paper, we propose a novel one-shot federated learning framework based on random feature extractor (FedRFE). It is the first framework that does not require local model training or pretrained model, thus featuring extreme resource-efficient. Through comprehensive experiments, we show FedRFE achieves competitive performance while being fairly robust in challenging scenarios including data heterogeneity and scalability.

References

Youssef Allouah, Akash Dhasade, Rachid Guerraoui, Nirupam Gupta, Anne-Marie Kermarrec, Rafael Pinot, Rafael Pires, and Rishi Sharma. Revisiting ensembling in one-shot federated learning. *Advances in Neural Information Processing Systems*, 37:68500–68527, 2024.

Flora Amato, Lingyu Qiu, Mohammad Tanveer, Salvatore Cuomo, Fabio Giampaolo, and Francesco Piccialli. Towards one-shot federated learning: Advances, challenges, and future directions. *arXiv preprint arXiv:2505.02426*, 2025.

Mahdi Beitollahi, Alex Bie, Sobhan Hemati, Leo Maxime Brunswic, Xu Li, Xi Chen, and Guojun Zhang. Parametric feature transfer: One-shot federated learning with foundation models. *arXiv preprint arXiv:2402.01862*, 2024.

Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečnỳ, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

Yun-Hao Cao and Jianxin Wu. A random cnn sees objects: One inductive bias of cnn and its applications. In *Proceedings Of The AAAI conference on artificial intelligence*, volume 36, pages 194–202, 2022.

Nicolas Carlini, Jamie Hayes, Milad Nasr, Matthew Jagielski, Vikash Sehwag, Florian Tramer, Borja Balle, Daphne Ippolito, and Eric Wallace. Extracting training data from diffusion models. In *32nd USENIX security symposium (USENIX Security 23)*, pages 5253–5270, 2023.

Haokun Chen, Hang Li, Yao Zhang, Jinhe Bi, Gengyuan Zhang, Yueqi Zhang, Philip Torr, Jindong Gu, Denis Krompass, and Volker Tresp. Fedbip: Heterogeneous one-shot federated learning with personalized latent diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 30440–30450, 2025.

Arthur da Cunha, Emanuele Natale, and Laurent Viennot. Proving the strong lottery ticket hypothesis for convolutional neural networks. In *ICLR 2022-10th International Conference on Learning Representations*, 2022.

Rong Dai, Yonggang Zhang, Ang Li, Tongliang Liu, Xun Yang, and Bo Han. Enhancing one-shot federated learning through data and ensemble co-boosting. *arXiv preprint arXiv:2402.15070*, 2024.

Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

Badih Ghazi, Noah Golowich, Ravi Kumar, Pasin Manurangsi, and Chiyuan Zhang. Deep learning with label differential privacy. *Advances in neural information processing systems*, 34:27131–27145, 2021.

Anna C Gilbert, Yi Zhang, Kibok Lee, Yuting Zhang, and Honglak Lee. Towards understanding the invertibility of convolutional neural networks. *arXiv preprint arXiv:1705.08664*, 2017.

Zenghao Guan, Yucan Zhou, and Xiaoyan Gu. Capture global feature statistics for one-shot federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 16942–16950, 2025.

Neel Guha, Ameet Talwalkar, and Virginia Smith. One-shot federated learning. *arXiv preprint arXiv:1902.11175*, 2019.

Mohsin Hasan, Guojun Zhang, Kaiyang Guo, Xi Chen, and Pascal Poupart. Calibrated one round federated learning with bayesian inference in the predictive space. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 12313–12321, 2024.

Clare Elizabeth Heinbaugh, Emilio Luz-Ricca, and Huajie Shao. Data-free one-shot federated learning under very high statistical heterogeneity. In *The Eleventh International Conference on Learning Representations*, 2023.

Kevin Jarrett, Koray Kavukcuoglu, Marc'Aurelio Ranzato, and Yann LeCun. What is the best multi-stage architecture for object recognition? In *2009 IEEE 12th international conference on computer vision*, pages 2146–2153. IEEE, 2009.

Divyansh Jhunjhunwala, Shiqiang Wang, and Gauri Joshi. Fedfisher: Leveraging fisher information for one-shot federated learning. In *International Conference on Artificial Intelligence and Statistics*, pages 1612–1620. PMLR, 2024.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and trends® in machine learning*, 14(1–2):1–210, 2021.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Qinbin Li, Yiqun Diao, Quan Chen, and Bingsheng He. Federated learning on non-iid data silos: An experimental study. In *2022 IEEE 38th international conference on data engineering (ICDE)*, pages 965–978. IEEE, 2022.

Xuhui Li, Zhengquan Luo, Zihui Cui, Xin Cao, and Zhiqiang Xu. Ofeded: One-shot federated learning with model ensemble and dataset distillation. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pages 1706–1715, 2025.

Xiang Liu, Liangxi Liu, Feiyang Ye, Yunheng Shen, Xia Li, Linshan Jiang, and Jialin Li. Fedlpa: One-shot federated learning with layer-wise posterior aggregation. *Advances in Neural Information Processing Systems*, 37:81510–81548, 2024.

Xiang Liu, Zhenheng Tang, Xia Li, Yijun Song, Sijie Ji, Zemin Liu, Bo Han, Linshan Jiang, and Jialin Li. One-shot federated learning methods: A practical guide. *arXiv preprint arXiv:2502.09104*, 2025.

Mi Luo, Fei Chen, Dapeng Hu, Yifan Zhang, Jian Liang, and Jiashi Feng. No fear of heterogeneity: Classifier calibration for federated learning with non-iid data. *Advances in Neural Information Processing Systems*, 34:5972–5984, 2021.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

Matias Mendieta, Guangyu Sun, and Chen Chen. Navigating heterogeneity and privacy in one-shot federated learning with diffusion models. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2601–2610. IEEE, 2025.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 4. Granada, 2011.

Kilian Pfeiffer, Martin Rapp, Ramin Khalili, and Jörg Henkel. Federated learning for computationally constrained heterogeneous devices: A survey. *ACM Computing Surveys*, 55(14s):1–27, 2023.

Andrew M Saxe, Pang Wei Koh, Zhenghao Chen, Maneesh Bhand, Bipin Suresh, and Andrew Y Ng. On random weights and unsupervised feature learning. In *Icml*, volume 2, page 6, 2011.

MyungJae Shin, Chihoon Hwang, Joongheon Kim, Jihong Park, Mehdi Bennis, and Seong-Lyun Kim. Xor mixup: Privacy-preserving data augmentation for one-shot federated learning. *arXiv preprint arXiv:2006.05148*, 2020.

Rui Song, Dai Liu, Dave Zhenyu Chen, Andreas Festag, Carsten Trinitis, Martin Schulz, and Alois Knoll. Federated learning via decentralized dataset distillation in resource-constrained edge environments. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pages 1–10. IEEE, 2023.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9446–9454, 2018.

Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices., 2012.

Mingzhao Yang, Shangchao Su, Bin Li, and Xiangyang Xue. One-shot heterogeneous federated learning with local model-guided diffusion models. In *Forty-second International Conference on Machine Learning*.

Mingzhao Yang, Shangchao Su, Bin Li, and Xiangyang Xue. Exploring one-shot semi-supervised federated learning with pre-trained diffusion models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 16325–16333, 2024a.

Mingzhao Yang, Shangchao Su, Bin Li, and Xiangyang Xue. Feddeo: Description-enhanced one-shot federated learning with diffusion models. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 6666–6675, 2024b.

Obaidullah Zaland, Shutong Jin, Florian T Pokorny, and Monowar Bhuyan. One-shot federated learning with classifier-free diffusion models. *arXiv preprint arXiv:2502.08488*, 2025.

Hui Zeng, Wenke Huang, Tongqing Zhou, Xinyi Wu, Guancheng Wan, Yingwen Chen, and Zhiping Cai. Does one-shot give the best shot? mitigating model inconsistency in one-shot federated learning. In *Forty-second International Conference on Machine Learning*.

Hui Zeng, Minrui Xu, Tongqing Zhou, Xinyi Wu, Jiawen Kang, Zhiping Cai, and Dusit Niyato. One-shot-but-not-degraded federated learning. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 11070–11079, 2024.

Jie Zhang, Chen Chen, Bo Li, Lingjuan Lyu, Shuang Wu, Shouhong Ding, Chunhua Shen, and Chao Wu. Dense: Data-free one-shot federated learning. *Advances in Neural Information Processing Systems*, 35:21414–21428, 2022.

Junyuan Zhang, Songhua Liu, and Xinchao Wang. One-shot federated learning via synthetic distiller-distillate communication. *Advances in Neural Information Processing Systems*, 37:102611–102633, 2024.

Yanlin Zhou, George Pu, Xiyao Ma, Xiaolin Li, and Dapeng Wu. Distilled one-shot federated learning. *arXiv preprint arXiv:2009.07999*, 2020.

Zhi-Hua Zhou. *Ensemble methods: foundations and algorithms*. CRC press, 2025.

Supplementary Material

Luyuan Yang[1] **Shayan Shafaei**[1] **Yiming Liu**[1] **Naeem Shahabi Sani**[1] **Yu Cai**[1] **Jun (Luke) Huan**[2] **Chao Lan**[1]

[1]School of Computer Science, University of Oklahoma, Norman, Oklahoma, USA
[2]AWS AI, USA

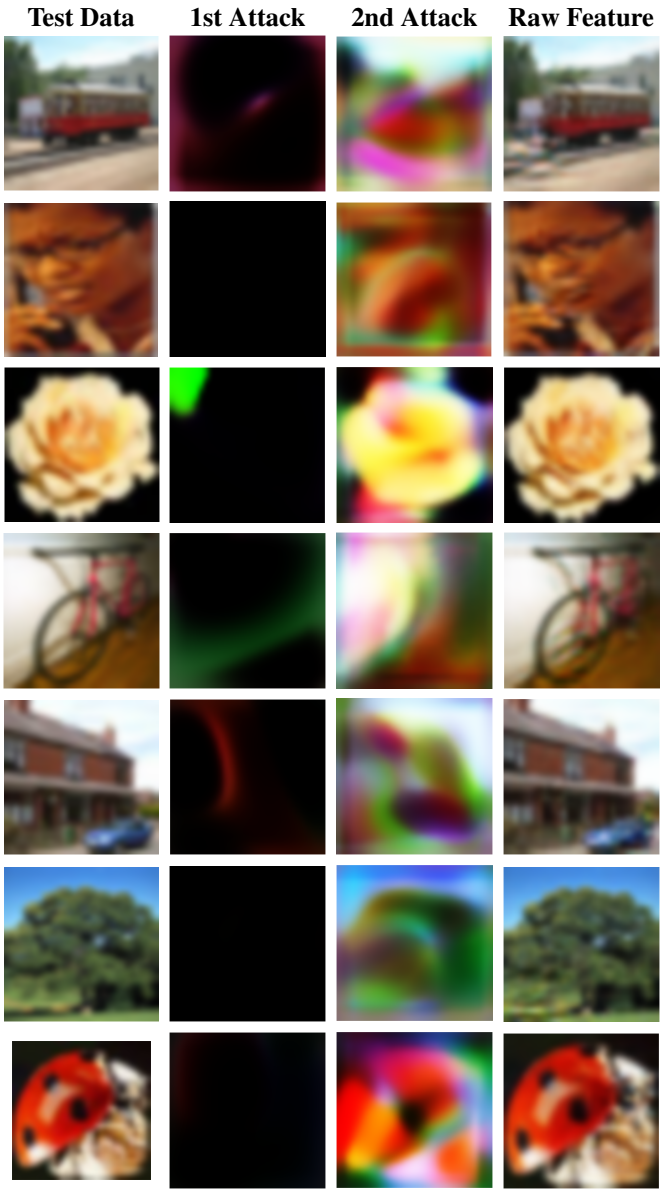


Figure 6: Results of reconstruction attacks on CIFAR100.

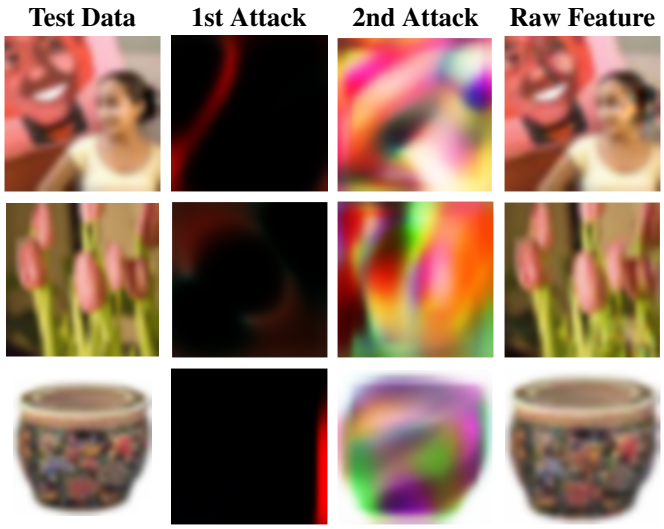


Figure 7: Results of reconstruction attacks on CIFAR100.

A RECONSTRUCTION ATTACK

We consider two scenarios that server acts as an attacker: 1). server uses all functions $\{\phi_1, \phi_2, \ldots, \phi_{10}\}$ to reconstruct the private data with uploaded features $[s_1, \ldots, s_{10}]$, 2). server correctly locates the feature outputs and corresponding extractors and uses one ϕ_i to reconstruct the private data with corresponding features s_i. We utilize the popular feature inversion technology Ulyanov et al. [2018] to simulate above attacks. Intuitively, this mechanism learns a function f_θ that can generate images whose mappings (by the FedFRE random feature extractor) match the mappings uploaded from clients. To be specific, for Attack 1, it solves

$$\min_\theta ||[\phi_1(f_\theta(z)), \ldots, \phi_{10}(f_\theta(z))] - [s_1, \ldots, s_{10}]||^2,$$

where z is a randomly sampled noise, s_i's are client uploaded features, ϕ_i is the random feature extractor of FedFRE and f_θ has a U-Net-like encoder–decoder architecture. For Attack 2, it solves

$$\min_\theta ||\phi_i(f_\theta(z)) - s_i||^2.$$

We report all reconstruction images in 6 7 and their average similarity measure in 11. For attack one, it's impossible to recover any useful information due to the stacked feature vector. For attack two, it's difficult to identify the class or content of original image. Moreover, the mean PSNR value is below 15, suggesting the reconstructed images are heavily distorted and the visual information leakage is very limited.

Table 11: Similarity measure for Reconstructed Images.

Reconstruction source	**PSNR**
Raw features	27.01± 1.72
2nd Attack	14.78 ± 1.55

B COMMUNICATION COST

As mentioned above, FedRFE requires uploading a $\mathbb{R}^{2560}$ vector for every sample, which may bring concerns about the total volume of data transmission. For the majority of prior methods, the standard transmitted local model is ResNet-18, which is equivalent to approximately 4500 vectors in terms of the number of parameters 12. Therefore, in realistic settings where each client typically holds at most thousands of samples [Caldas et al., 2018], the total communication volume of our method remains no greater than that of directly transmitting the local model.

Table 12: Comparison of Communication Costs (by parameters).

Message	**Memory Cost (per client)**
ResNet-18	11.7 M
4500 vectors $\mathbf{s_j} \in \mathbb{R}^{2560}$	11.5 M

C LABEL PRIVACY

Uploading numerical label of samples has precedent in federated learning. [Chen et al., 2025, Shin et al., 2020, Zhou et al., 2020, Zhang et al., 2024, Song et al., 2023, Li et al., 2025] require uploading the labels of real-world samples or synthetic samples. Another line of approaches require clients to provide the statistical information of labels, specifically the total number of samples belonging to each class [Heinbaugh et al., 2023, Guan et al., 2025, Luo et al., 2021]. This information reveals the local label distribution and is therefore closely related to direct label transmission.

Meanwhile, we provide a variant of FedRFE to meet stringent privacy requirements in real-world deployments.

C.1 FEDRFE-F

To protect label differential privacy (LDP), we provide a variant, called FedRFE-F. In addition to the original algorithm, each client also applies a popular LDP mechanism Ghazi et al. [2021] to obfuscate labels before sending them to the server. This mechanism flips labels with certain probability, and is guaranteed to preserve LDP. Results are shown in the following Table 13, where ϵ is the DP budget and the associated number is the percentage of flipped labels. It can be observed that the accuracy remains well preserved even when a portion of the labels is flipped.

Table 13: Classification Accuracy (%) of FedRFE-F

Dataset	α	$\epsilon = 2$ (55 % flip)	$\epsilon = 3$ (31 % flip)	$\epsilon = 4$ (14 % flip)	FedRFE
CIFAR10	0.01	55.19	61.07	65.00	66.51
	0.05	53.76	63.56	64.78	66.94
	0.10	61.18	64.82	66.03	66.99
	0.30	62.43	66.50	68.25	66.65
SVHN	0.01	67.05	73.65	76.32	81.01
	0.05	74.74	74.37	78.75	80.53
	0.10	75.27	77.85	79.49	80.77
	0.30	75.47	79.23	80.92	80.35
	α	$\epsilon = 4$ (64 % flip)	$\epsilon = 5$ (40 % flip)	$\epsilon = 6$ (20 % flip)	FedRFE
CIFAR100	0.01	23.72	30.89	34.27	39.10
	0.05	29.50	33.98	36.49	39.22
	0.10	31.16	36.48	37.87	39.40
	0.30	33.26	38.29	39.43	39.80

D MEMBERSHIP INFERENCE ATTACK

We also evaluate robustness of FedFRE against membership inference attack, based on the setting in Chen et al. [2025]. Results with $\alpha = 0.3$ and $k = 5$ are shown below. Metric Entropy is defined as the absolute difference between prediction entropy on testing data and prediction entropy on training data. Metric Loss is defined as the absolute difference between average cross-entropy loss on testing data and average cross-entropy loss on training data. Lower entropy and loss imply the method is more robust. We see the performance of FedFRE is comparable to some previous methods.

Table 14: Membership Inference Attack (MIA) analysis.

Dataset	**MIA Metric**	**FedAvg (one-shot)**	**FedPFT**	**FedCGS**	**FedRFE**
SVHN	Entropy	0.02	0.09	0.03	0.06
	Loss	0.04	0.78	0.01	0.22
CIFAR-10	Entropy	0.01	0.03	0.00	0.06
	Loss	0.06	0.30	0.06	0.35
CIFAR-100	Entropy	0.00	0.04	0.03	0.11
	Loss	0.19	0.61	0.58	0.66