Reprogrammable-FL: Improving Utility-Privacy Tradeoff in Federated Learning via Model Reprogramming

Huzaifa Arif
Electrical and Systems Engineering

Rensselaer Polytechnic Institute
Troy, NY, USA
arifh @ rpi.edu

Alex Gittens
Computer Science

Rensselaer Polytechnic Institute
Troy, NY, USA
gittea @ rpi.edu

Pin-Yu Chen
IBM Thomas J. Watson Research Center

IBM Research
Yorktown Heights, NY, USA
pin-yu.chen @ ibm.com

***Abstract*—Model reprogramming (MR) is an emerging and powerful technique that provides cross-domain machine learning by enabling a model that is well-trained on some source task to be used for a different target task without finetuning the model weights. In this work, we propose Reprogrammable-FL, the first framework adapting MR to the setting of differentially private federated learning (FL), and demonstrate that it significantly improves the utility-privacy tradeoff compared to standard transfer learning methods (full/partial finetuning) and training from scratch in FL. Experimental results on several deep neural networks and datasets show up to over 60% accuracy improvement given the same privacy budget. The code repository can be found at https://github.com/IBM/reprogrammble-FL.**

***Index Terms*—Model Reprogramming, Differential Privacy, Federated Learning, Privacy-Accuracy Tradeoff**

I. Introduction

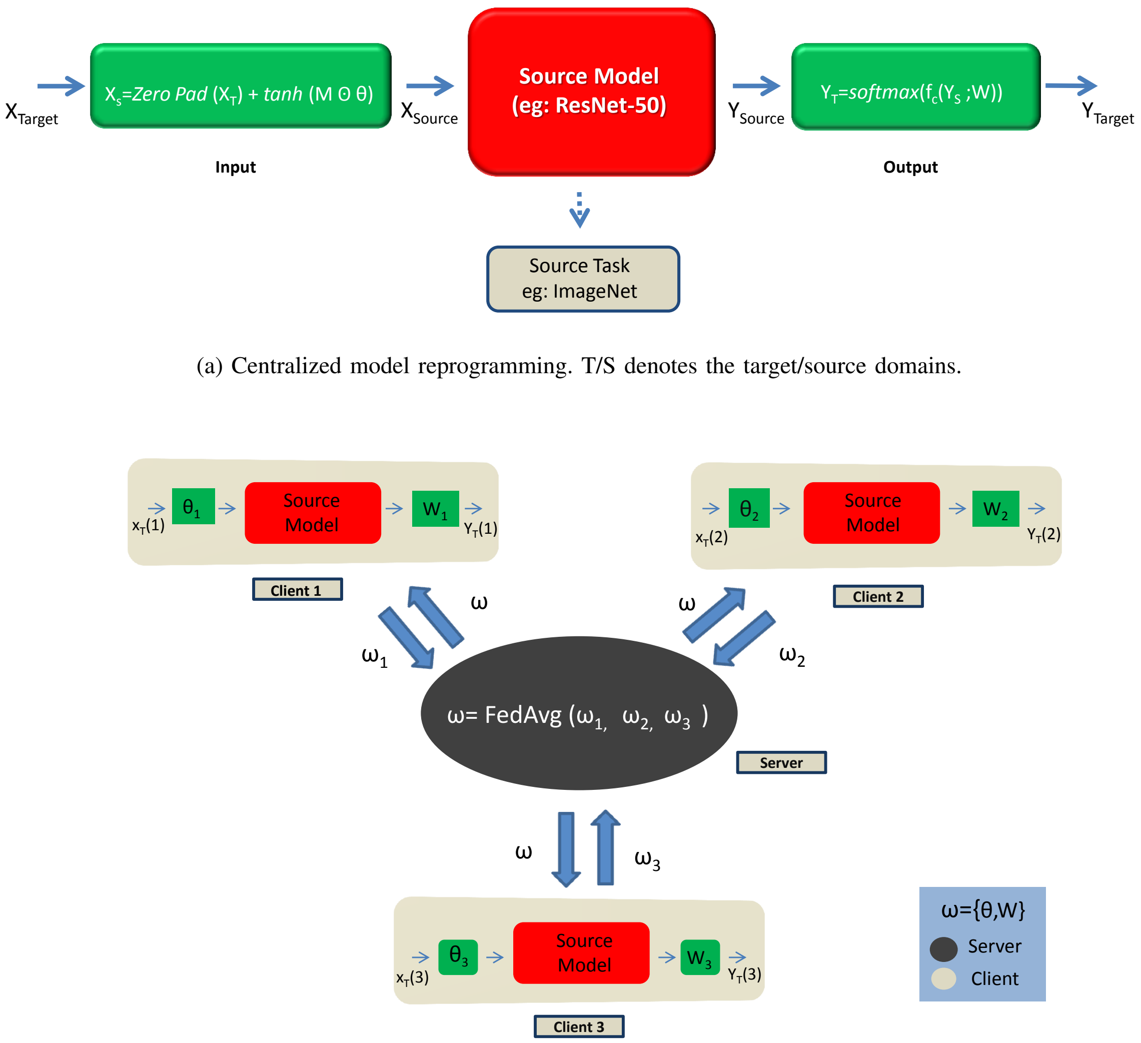
Federated learning (FL) enables learning a global model when the training data is distributed across multiple clients. We define a local version of the model as the model owned by a client and the global model as the model shared by all the clients/devices. Local models are trained locally on private training data and are shared periodically with the server, which averages the local models to obtain the global model. Figure 1 provides an overview of the federated learning framework.

Clients involved in a federated learning process are concerned about the privacy of their data shared in the learning process. For instance, if clients are hospitals sharing personal health data of their patients in a federated learning process, leakage of any information—be it a feature of a health record, or worse, a complete health record—could pose ethical and legal risks. Previously, it was believed that since clients only exchange local gradients or model parameters, the privacy of the client data would be well-preserved. However, recent works [1]–[3] show that naïvely trained FL models may not be private and exhibit data leakage risks including vulnerabilities to membership inference attacks and gradient leakage attacks.

Differential privacy (DP) [4] is a popular tool used to preserve privacy during model learning. In particular, the DP-SGD method [5] is a modification of the standard stochastic gradient descent (SGD) that incorporates additional noise to reduce privacy loss during training. However, a model trained using DP-SGD is likely to suffer from degraded accuracy, as noisy gradient estimates lead to poor convergence [6]. This leads to an important challenge in differentially private federated learning, i.e., attaining a better privacy-accuracy (utility) tradeoff. This paper focuses on improving the attainable accuracy given a privacy budget, using the standard DP-SGD algorithm for FL.

Multiple approaches exist to train a global model in a differentially private manner [7]. One straightforward approach is to ensure that each client trains its local model using DP-SGD. However, this approach does not provide a desirable accuracy-privacy tradeoff, as the privacy budget limits the number of training steps allowed for updating the global model, thus leading to sub-optimal accuracy of the global model. A more effective approach is to start with a model that was pre-trained on a public dataset and differentially privately finetune it in a federated manner to improve its accuracy on the dataset whose privacy we wish to ensure. Intuitively, finetuning an already trained model requires less rounds of training on the private dataset to achieve acceptable accuracy levels, and thus higher accuracy can be achieved at a lower privacy budget. Indeed, [8] has shown that even partial finetuning of a pre-trained model on the private dataset can drastically improve the privacy-accuracy tradeoff. In this paper, we use finetuning and transfer learning interchangeably.

In this paper, we show that model reprogramming (MR) [9], a recently introduced alternative to transfer learning [10], exhibits a superior accuracy-privacy tradeoff to fine-tuning approaches in the federated setting. The rationale is that MR is a more efficient approach for leveraging a source pre-trained model to solve a new target task: instead of changing the pre-trained weights as in transfer learning, MR attaches a trainable input transformation layer and an output mapping layer to the source model for reprogramming. Figure 2 illustrates the difference between MR and other approaches that tackle the privacy-accuracy tradeoff. It can be seen that MR keeps the source model unchanged and only modifies the input and output transformation layers during training, thereby efficiently



(a) Centralized model reprogramming. T/S denotes the target/source domains.

(b) Federated learning with model reprogramming (Reprogrammable-FL)

Fig. 1: The framework of federated learning using model reprogramming (Reprogrammable-FL). The corresponding mathematical notation is introduced in Section II. (a) Model reprogramming in a centralized setting. (b) Model reprogramming in a federated setting. In both cases only the parameters associated with the input and output layers attached to the pre-trained source model are altered during reprogramming; this differs from standard transfer learning approaches that finetune the parameters of the source model itself.

utilizing the pretrained model's capabilities.

The organization of the remainder of the paper is as follows. A concise summary of the paper's contributions and findings is followed by a brief presentation used to familiarize the reader with the notions and notations of Model Reprogramming (MR), Federated Learning (FL), and Differential Privacy (DP). Next, the proposed methodology of MR in the federated setting, named *Reprogrammable-FL*, is introduced. Finally, the results of several numerical experiments are presented to demonstrate the superiority of Reprogrammable-FL over existing methods for achieving better privacy-accuracy tradeoffs in DP-FL settings.

A. Our Contributions

- We propose *Reprogrammable-FL*, a novel framework for achieving model reprogramming in the differentially private federated learning setting, which achieves an improved utility-privacy tradeoff. This framework uses an unchanged pre-trained source model, attaches a pair

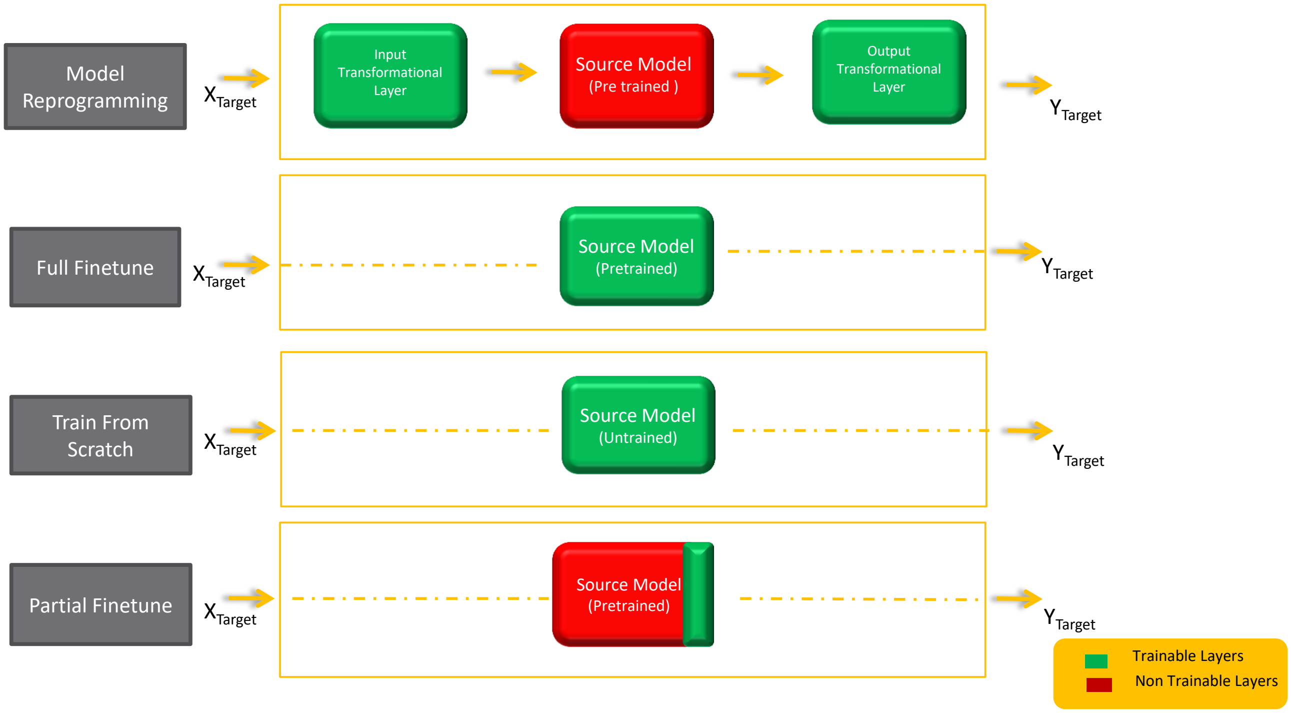


Fig. 2: Illustration of different machine learning principles in transfer learning settings. Top to Bottom: Model Reprogramming, Full-Finetune, Train from Scratch, and Partial Finetune. The colors on the right panel specify the trainable and non-trainable layers of pretrained/untrained neural networks in each method.

of learnable input and output transform layers to the source model, and learns their parameters in the federated setting.

- We show that, across datasets and models, the novel use of model reprogramming consistently and massively outperforms existing approaches that aim to improve the accuracy-privacy tradeoff. For instance, on CIFAR-10, for a given privacy budget of $\epsilon = 1.04$, Reprogrammable-FL exhibits up to a 61.4% increase in classification accuracy over other approaches.
- Our ablation study shows that more accurate (usually deeper) source models give a better accuracy-privacy tradeoff for a given budget in Reprogrammable-FL, while competing approaches may not have such benefits.

II. Background and Related Work

A. Model Reprogramming

Model Reprogramming (MR) [9] is an approach toward resource-efficient cross-domain machine learning originally introduced in [11] as an alternative to model finetuning approaches. In MR (see Figure 1a), a pre-trained source model is adapted for use in a new domain (i.e., a target domain) by the addition of input and output transformation layers. The parameters of the source model are frozen after training on the source task in the source domain, then the parameters of the input and output transformations are learned to map, respectively, inputs in the target domain to inputs in the source domain and outputs in the source domain to outputs in the target domain. The seminal work adapted image classifiers for several image-based tasks, but subsequent work has demonstrated the success of MR in cross-domain model adaptation, including reprogramming language recognition models to learn molecular representations [12], reprogramming acoustic models for time series classification [13], and reprogramming general image classifiers for data-limited bio-medical measurements [14].

1) Transformation Layers: Without loss of generality, we assume that the inputs and outputs for the source and target tasks are vectors of dimension $d_{\mathcal{S}}$ and $d_{\mathcal{T}}$ respectively. To facilitate the MR approach, we require $d_{\mathcal{T}} \leq d_{\mathcal{S}}$. We also assume that the two tasks under consideration are classification tasks, with $K_{\mathcal{T}}$ target classes and $K_{\mathcal{S}}$ source classes; and that $K_{\mathcal{T}} \leq K_{\mathcal{S}}$.

Input Transformation Layer. The input transformation layer maps a sample $x_{\mathcal{T}}$ in the input space of the target task to a point $\widetilde{x}_{\mathcal{T}}$ in the input space of the source task, while including learned parameters in $\widetilde{x}_{\mathcal{T}}$ that assist in partially adapting the source model to the target task. In the case of scaled input range $[-1, 1]$ in each input dimension, as considered in this work, a standard choice of input transformation [11] is given by

$$\widetilde{x}_{\mathcal{T}} = \text{ZeroPadding}(x_{\mathcal{T}}) + \tanh(M \odot \Theta).$$

The operator ZeroPadding adds borders of zeros around the input image to result in an image of size $d_{\mathcal{S}}$, and M is a binary mask that equals zero where ZeroPadding$(x_{\mathcal{T}})$ is equal to $x_{\mathcal{T}}$ and equals one on the border of ZeroPadding$(x_{\mathcal{T}})$. The only learnable parameter, Θ, is an input-independent perturbation of the padded image that helps adapt the source model to the target task, and $\tanh$ is the element-wise hyperbolic tangent function that ensures each dimension of $\widetilde{x}_{\mathcal{T}}$ stays in the range $[-1, 1]$.

Output Transformation Layer. The output transformation layer maps K_S classes to K_T classes through a trainable fully connected layer parametrized by W. This mapping between the logit outputs of the source model, $\hat{y}_{\mathcal{S}} = f_{\mathcal{S}}(\widetilde{x}_{\mathcal{T}})$, and the logits of the target task, $\hat{y}_{\mathcal{T}}$, is denoted by $\hat{y}_{\mathcal{T}} = F_c(W; \hat{y}_{\mathcal{S}})$. The final prediction on the target task is given by the softmax output:

$$\hat{y}_{\mathcal{T}} = \text{softmax}[F_c(W; \hat{y}_{\mathcal{S}})]$$

2) Training: Given a pretrained source model $f_{\mathcal{S}}$ and a target training set $\{x^i_{\mathcal{T}}, y^i_{\mathcal{T}}\}_{i=1}^n$, the reprogramming parameters Θ and W are learned as minimizers of

$$f(\Theta, W) = \frac{1}{n} \sum_{i=1}^{n} \ell(\hat{y}^i_{\mathcal{T}}(f_{\mathcal{S}}(\tilde{x}^i_{\mathcal{T}})), y^i_{\mathcal{T}}),$$

where ℓ is a measure of discrepancy between the true target and the predicted target.

As proved in [13], the target loss of MR is upper bounded by the summation of two terms: the source loss and the representation alignment loss between the source data and the reprogrammed target data. MR can fully leverage the pretrained source model when reprogramming suffices to align the representations of the target data with those of the source data. In this case, it is possible to use the classification model encoded in the source model to solve the target task.

B. Federated Learning

To date, MR has been studied exclusively in a shared memory (centralized machine learning) setting. This work is the first to study the performance of MR in distributed learning settings, and specifically investigate its privacy-accuracy trade-offs in the setting of differentially private federated learning.

Federated learning is a machine learning paradigm that allows multiple clients to collaboratively train a model in a distributed manner without sharing training data. It was initially proposed in [15] and has gained popularity due to its demonstrated ability to efficiently use data distributed across a large number of clients. There is a rich body of literature addressing problems in federated learning such as data heterogeneity [16], [17], expensive communication [18], [19], and privacy guarantees [20]. The survey paper in [21] provides an overview of the field.

Let $f_i(\omega)$ measure the loss of the shared model ω on the ith client, then FL attempts to minimize the weighted global loss

$$F(\omega) = \sum\nolimits_{i=1}^{m} \alpha_i f_i(\omega)$$

while minimizing communication and data exposure between the clients [15]. Typically f_i measures the performance of the model on the training data set local to the ith client. In this paper we take

$$f_i(\omega) = \sum\nolimits_{j=1}^{n_i} \ell(\omega; (\mathbf{x}^i_j, \mathbf{y}^i_j)),$$

where $\ell(\omega; (\mathbf{x}^i_j, \mathbf{y}^i_j))$ measures the loss of the model ω on the jth training example $(\mathbf{x}^i_j, \mathbf{y}^i_j)$ on client i. We take $\alpha_i = n_i/N$ to be the fraction of the overall training data present on the ith client, where N is the total number of samples used in the training process.

To achieve the goals of minimizing communication and data exposure, FL algorithms alternate between aggregating local models to form a global model and locally updating these global models on the clients to form more accurate local models. Algorithms vary in their choice of methods to aggregate the local models, methods used to update the local models, choice of clients to participate in each round, and in additional steps that may be introduced for various reasons: e.g., to increase privacy, decrease communication costs, mitigate the effects of non-participating clients, or alleviate the impact of heterogenous data distributions across the clients.

The widely employed federated averaging (FedAvg) algorithm forms the tth global model as an average over the local models $\omega_t = \sum_{i=1}^m \alpha_i \omega^i_t$. The local models ω^i_t are obtained by using multiple steps of SGD on each worker to update the previous global model ω_{t-1} to minimize f_i [15]. This process continues until model convergence.

C. Differential Privacy and DP-SGD

The process of fitting ML models potentially discloses sensitive information about the training data set, even after seemingly clever anonymization techniques are used on the data set. This concern is exacerbated in the federated setting, where the communication of the sequence of global models between the clients potentially enables information leakage.

Differential privacy (DP) provides strong privacy guarantees and has become the standard framework for privacy-preserving ML [4]. A randomized algorithm $\mathcal{A}$ is said to be (ε, δ)-DP if it guarantees that for any two training data sets $\mathcal{D}$ and $\mathcal{D}'$ that differ by the inclusion or exclusion of a single training example, and any set S in the output space,

$$\mathcal{P}\left[\mathcal{A}(\mathcal{D}) \in S\right] \leq \exp(\varepsilon)\, \mathcal{P}\left[\mathcal{A}(\mathcal{D}') \in S\right] + \delta.$$

When $\mathcal{D}$ is the training data set and $\mathcal{A}$ is the algorithm used to learn an ML model, this guarantee ensures that even if all the other data points utilized in fitting a model are known, one cannot infer the presence or absence of a particular individual data point from the learned model because the models $\mathcal{A}(\mathcal{D})$ and $\mathcal{A}(\mathcal{D}')$ are very likely to be similar. Smaller values of ε and δ give stronger privacy guarantees.

By far the most popular and ubiquitous approach to privacy-preserving model learning is differentially private SGD (DP-SGD), which modifies the standard SGD algorithm by using the Gaussian mechanism to lower disclosure risk [5]. Let g be a deterministic vector-valued query function that takes a dataset as input, and define its sensitivity S_g as the maximum of $\|g(\mathcal{D}) - g(\mathcal{D}')\|_2$ over adjacent datasets. The Gaussian mechanism uses

$$g(\mathcal{D}) + \mathcal{N}(0, S_g^2\sigma^2\mathbf{I})$$

as a more private proxy for $g(\mathcal{D})$; here $\mathcal{N}(0, S_g^2\sigma^2\mathbf{I})$ denotes zero-mean Gaussian noise with the given covariance matrix. Intuitively, the addition of noise calibrated to the sensitivity level of the query function hides the influence of any one particular data point; more precisely, given $\varepsilon \in (0,1)$ and $\delta \in (0,1)$, it suffices to take $\sigma > 2\varepsilon^{-1}\ln(1.25\delta^{-1})$ for this proxy for g to be (ε, δ)-DP [4].

In the application of the Gaussian mechanism to DP-SGD, the query function g is the SGD gradient estimator evaluated on the training dataset, and its sensitivity is naturally bounded by the ℓ_2 norm of the largest gradient on any of the training data points. This quantity is unknown, changes over the course of training, and can be prohibitively large, so in DP-SGD the sensitivity of the gradient estimator is fixed at a hyperparameter C by passing it through the Clip operator:

$$\text{Clip}(x) = \frac{x}{\max\{1, \|x\|_2/C\}}.$$

The moments accounting method of [5] is used to track the evolution of the privacy parameters ε and δ during DP-SGD training.

In practice, DP-SGD is used to update the local models in FedAvg to ensure that the model is learned privately; this variant of FedAvg is called DP-FedAvg. Our Algorithm 2 in Section III uses DP-FedAvg to implement Model Reprogramming (MR) in the federated setting. In our study, DP-FedAvg is used as the algorithmic framework for FL because it comprises the essential components of FL – model averaging, multiple rounds of local model updating, and privacy-preserving learning – without additional algorithmic enhancements whose presence may complicate the interpretation of the impact of MR on FL.

For clarity, in our proposed Reprogrammable-FL framework, we note that the privacy guarantee is specific to the private data for FL, not the public data to train the source model.

III. Methodology of Reprogrammable-FL

We consider the setting of MR on FL (i.e., federated MR), as illustrated in Figure 1b. The client-server architecture follows [15] and the DP-SGD methodology follows [5] and [22]. This work is the first to study model reprogramming in the context of differentially private federated learning, and establish that it exhibits superior privacy-utility tradeoffs compared to fine-tuning.

The client- and server-side algorithms of our proposed Reprogrammable-FL framework are given in Algorithms 1 and 2, respectively. The same pretrained source model $f_{\mathcal{S}}$ is distributed to each of the clients before the start of Reprogrammable-FL. At the start of each communication round, the server communicates the current global reprogramming parameters $\omega = \{\Theta, W\}$ to all clients; recall that Θ and W denote the trainable parameters of the input and output transformation layers, respectively. DP-SGD is used locally to obtain updated local reprogramming parameters that increase performance on the local training data. Finally, the clients return their reprogramming parameters to the server, which aggregates them to compute the latest global model.

Consider a setting of m clients where the private data on the i^{th} client with n_i samples are denoted by

$$\mathbf{x}^i, \mathbf{y}^i = \{x^i_{\mathcal{T},j}, y^i_{\mathcal{T},j}\}_{j=1}^{n_i}$$

Here, the subscript $\mathcal{T}$ indicates that samples from the target domain are used for local training and j indicates a specific training sample. Consider the training procedure of one client. At the beginning of each round, the client receives the latest global model, then trains for L local iterations. In each local iteration, the client samples a batch $\mathcal{B}$ uniformly at random from the local training dataset and updates the local parameters using DP-SGD. Once the local training ends, the clients send their local reprogramming parameters $\omega_L = (\Theta_L, W_L)$ back to the server. Algorithm 1 provides the details of one round of client training. For clarity, we abbreviate the loss function of the i^{th} client to

$$\ell(\omega; (\mathbf{x}^i_b, \mathbf{y}^i_b)) = \ell(\hat{y}(f_{\mathcal{S}}(\tilde{\mathbf{x}}^i_b)), \mathbf{y}^i_b),$$

where the subscript b denotes the b^{th} sample from the current batch of local data, and $\tilde{\mathbf{x}}^i_b$ and $\hat{y}$ are computed using the current local reprogramming parameters.

Algorithm 1 Federated Model Reprogramming (Reprogrammable-FL) – Client Side

```
Input: $\mathbf{x}^i, \mathbf{y}^i = \{x^i_{\mathcal{T},j}, y^i_{\mathcal{T},j}\}_{j=1}^{n_i}$
ClientUpdate$^i(\omega_t; C, \sigma, L, \mathcal{B}, f_{\mathcal{S}})$
$\omega^i_0 \leftarrow \omega$
for $t \in \{0, \ldots, L-1\}$ do
    $\mathcal{B} \leftarrow$ uniform sampling w/o replacement
    Update input transformation layer $\Theta^i_{t+1} \leftarrow \Theta^i_t - \eta \cdot \frac{1}{\mathcal{B}} \cdot [\sum_{b\in\mathcal{B}} \text{Clip}(\nabla_{\Theta^i}\ell(\omega^i_t; (\mathbf{x}^i_b, \mathbf{y}^i_b))) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I}))]$
    Update output transformation layer $W^i_{t+1} \leftarrow W^i_t - \eta \cdot \frac{1}{\mathcal{B}} \cdot [\sum_{b\in\mathcal{B}} \text{Clip}(\nabla_{W^i}\ell(\omega^i_t; (\mathbf{x}^i_b, \mathbf{y}^i_b))) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I}))]$
    $\omega^i_{t+1} \leftarrow (\Theta^i_{t+1}, W^i_{t+1})$
end for
return $\omega^i_L$
```

Consider the server-side computations of Reprogrammable-FL : the clients train their local models in a parallel manner and send the trained layer parameters $\{\omega^i_L\}_{i=1}^n$ back to the server after the local training procedures conclude. The server aggregates the local models to form the next global model, which is then sent back to the clients. Training continues in this manner for T rounds. Given a fixed pre-determined δ, and noise variance σ, the moments accountant approach of [5] is used at each round t to compute the expended privacy budget so far by all the clients over the entire dataset, ϵ. This computation uses knowledge of the total number of iterations over the full dataset, $t \times L$, and total effective batch size $m \times \mathcal{B}$. See Algorithm 2 for the server-side algorithm.

Algorithm 2 Federated Model Reprogramming (Reprogrammable-FL) – Server Side

```
Input: $\omega_0 = (\Theta_0, W_0)$ initialised randomly, $\delta$, $T$, $L$, $\mathcal{B}$, $C$, $\sigma$, $N$, $f_\mathcal{S}$
Output: $\omega_T = (\Theta_T, W_T)$
for $t \in \{0, \ldots, T-1\}$ do
    for all $i \in m$ in parallel do
        $\omega^i_{t+1} = \textbf{ClientUpdate}^i(\omega_t, C, \sigma, L, \mathcal{B}, f_\mathcal{S})$
    end for
    Update $\omega_{t+1} \leftarrow \sum_{i=1}^{m} \alpha_i \omega^i_{t+1}$
    Server calculates expended privacy budget $\varepsilon$ using moments accountant for fixed $\delta$
end for
```

IV. Experimental Evaluation

We simulate the performance of the Reprogrammable-FL and its comparison to baselines in a federated setting. These baselines are the current approaches that aim to improve the accuracy-privacy tradeoff. Our primary comparison with the baselines is done for the IID (independently and identically distributed) setting and we also show results in the non-IID setting.

The results shown are for a federated scenario with 100 rounds and three clients with each client doing 1 local iteration before aggregation of the parameters in all scenarios considered. Effects of more clients and more local epochs are then analyzed in the ablation study. All the baselines and Reprogrammable-FL are getting trained using DP-SGD with the server implementing Fed-Avg as the aggregation strategy. Ablation studies with more clients and different local iterations are also conducted. Minimal changes in the experimental structure between baselines and model reprogramming demonstrate the power of model reprogramming over baselines as the only change is the structure of the client models. The baselines used are explained below in more detail. See Figure 2 for the comparison of model reprogramming with the baselines.

A. BaseLine Full Finetune – (BL-FF)

This baseline has the clients using a pre-trained model where all parameters are tuned on the private dataset of the clients. The pre-trained model is modified such that the number of classes matches the performed task. For instance, to perform CIFAR-10 classification, a Resnet-50 model pre-trained on ImageNet is modified for 10 classes (by randomly initializing the last fully connected layer). Note that this is different from MR in that the model is modified to match the target tasks. Training in the federated setting means that the clients exchange all the model parameters after performing L local iterations in a DP-SGD manner and the server averages all the received parameters.

B. Baseline Train from Scratch – (BL-TS)

This baseline consists of using the same architecture of a source model and training it on some target task. The architecture is modified such that the output classes match the target task. This entire model is trained in a DP-SGD manner. In the federated setting, all the parameters are exchanged, similar to the full-finetune setting.

C. Baseline Partial Finetune – (BL-PF)

This baseline consists of using a pre-trained model on some source task to perform classification on some target task. In principle, this baseline assumes a similar setting to BL-FF with the difference being that instead of fully fine tuning all the model parameters, only the final modified classification layer is randomly initialized and finetuned on the target task with DP-SGD on the private data.

D. Parameters

These parameters are fixed for all the source models and target tasks. The motivation for making little change to the parameters was to demonstrate the strength of model reprogramming over the competitive baselines. The effect of using different batch sizes is studied in Section VI.

- Noise Variance σ^2 =1.1
- Clipping norm C = 1.0
- Training Batch Size $\mathcal{B}$ = 256
- Learning Rate η = 0.15
- δ = 10^{-5}

For model reprogramming, we treat the image size in the padded image to be a hyperparameter. Therefore, we upsample the image to size 200×200 where the size of the image is 224×224. For using baselines we upsample the image to size 224×224 for the source model that expects such input in all of our considered tasks.

E. Source Models

For our experiment, we employ five different source models. All these source models were trained on ImageNet as a source task. We employ the popular ResNet [23] architecture of various depths as our source models, including ResNet18, ResNet50, and ResNet152 of depths 18, 50 and 152 layers correspondingly. A more recent architecture ResNext [24] with different widths is also considered for evaluation, including Resnext50 and Resnext101. All these source models were trained on Imagenet as the source task.

F. Target Tasks and Datasets

We select three representative datasets for image classification to conduct our experiments. CIFAR-10 is a common dataset in FL [25] and model reprogramming [11]. Oxford-IIT Pet dataset is studied in various transfer learning applications [26]. These datasets are large and relatively more complex than MNIST datasets. Since privacy is an important requirement in healthcare-related machine learning applications, the third choice of the dataset (Blood MNIST) is used to demonstrate the efficacy of using model reprogramming in federated learning for health applications.

a) CIFAR-10: CIFAR-10 is a popular dataset that consists of 60000 images of size 32×32 with 10 classes (e.g.: dog, cat, airplane, etc). The data is split into 50000 training images and 10000 testing images. This is a well-balanced dataset such that each class has the same number, i.e., 6000 training samples. We evaluate this dataset in the IID setting such that the data is equally partitioned amongst the clients.

This dataset is also evaluated in the non-IID setting where the non-IID conditions are simulated in two ways [27]: one being to divide the training data unequally between the clients (similar to Quantity Skew in [27] except that the samples were not chosen through Dirichlet distribution) and the other being to divide the certain classes to certain clients. We simulate the former condition in this section and the class imbalance setting is evaluated in the ablation study. For three clients the data is divided in a 45:9:1 ratio.

b) Oxford-IIIT: Oxford-IIIT dataset has 37 classes with 200 images for each class. These contain 25 categories of dogs and 12 categories of cats. This is also a well-balanced dataset with a training size of 7400 samples and we demonstrate its performance in the IID condition.

c) Blood-MNIST: The Blood-MNIST dataset are peripheral images of blood cells that come from uninfected patients [28] of original shape $3 \times 360 \times 363$. This dataset has 17,092 images of 8 different blood cells. [29] preprocessed these images to $3 \times 28 \times 28$ which we upsample to size 224 before feeding them into the source models. We simulate this dataset on different source models for the IID case. The training set consists of 11,959 data samples and the test dataset consists of 3421 images.

G. Computational Resources

The CIFAR-10 dataset on all the source models with all of the baselines was evaluated using a core 2.5 GHz Intel Xeon Gold 6248 NVIDIA Tesla V100 GPU with a 32 GiB HBM and 768 GiB RAM per node. This platform was also used to simulate Blood-MNIST for ResNet-152 architecture.

The Oxford-IIIT and Blood MNIST were evaluated using Google Colab for all different source models. Google Colab uses Tesla T4 GPU on its platform. Our experiments were done using Pytorch 1.12 on both platforms.

V. Performance Evaluation and Discussion

For performance evaluation, we compare Reprogrammable-FL with baselines in both centralized and federated settings to study their accuracy-privacy tradeoffs.

Figs 3–11 show the performance comparison of the baselines with Reprogrammable-FL. The values on the y-axis are the accuracy and the values on the x-axis are the privacy budget consumed after every 10 rounds calculated from the moment accountant method. This ϵ value is a function of the total number of iterations of DP-FedAvg, the batch size, the size of the dataset, and the δ value). The batch size is fixed in the plots to illustrate how the choice of the number of iterations of DP-FedAvg and local epochs used in training the models impact the privacy budget. The choice of ϵ values are inspired from the work of [30], [22] and [31] but our central approach lies around fixing the number of global rounds and local epochs and calculating the result privacy budget and accuracy at each round. Fig 12 evaluates this performance for various batch sizes. Fig 13 shows a performance comparison of using different batch sizes, with other parameters constant. We see a slight advantage in using larger batch sizes in terms of the model utility achieved for a fixed number of iterations of FedAvg.

Figure 3a shows the performance of MR over the baselines in the centralized case using CIFAR-10 and ResNet-50. The evaluation is measured as the test accuracy for various ϵ values calculated from the moment's accountant method [5]. We see clearly that for any ϵ value, MR gives better accuracy than any of the baselines.

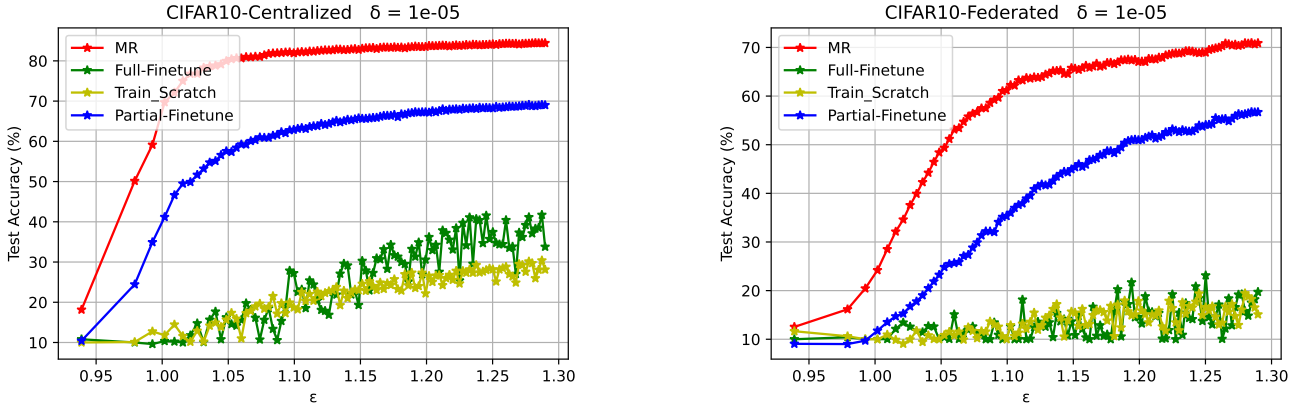
In the federated learning case, this ϵ value is evaluated using the moment's accountant method after every global round. Test accuracies are evaluated for these various ϵ values. Figure 3b shows the performance in the federated setting of three clients for different budget (ϵ) values in the IID case for CIFAR-10.

Figure 4 shows the Non-IID case for quantity imbalance for three clients in CIFAR-10. The test accuracy is the accuracy of the global model plotted against the budget consumed by the global model. It can be clearly observed that MR does much better than its baselines in both IID and non-IID settings.

Table I shows the accuracies for given ϵ values for various combinations of target task and source models in the IID case, computing using one run for each of the experiments. Table II reports averages and variances over three runs in the accuracies of partial-finetuning and reprogrammable-FL. Table III considers the non-IID scenario (Quantity Skew). In all cases, we see MR outperforms the baselines significantly. Additionally, we make a note that for any epsilon values below the reported ones, MR consistently provides a better accuracy tradeoff than any of the baselines. Notably, compared to the best baseline and given the same privacy budget in Table I, MR can attain a significant increase in the accuracy across different source models by up to 61.4%/62.7%/23.1% on CIFAR-10/Oxford-IIIT/Blood-MNIST, respectively. For the non-IID case, MR has improvements on a similar scale. On CIFAR-10 it provides a 63.1% better accuracy tradeoff than the baselines for similar budget values.

We make the following key observations regarding these empirical results:

- Partial finetuning is the most competitive baseline in most cases. One possible explanation for this is that it updates fewer parameters than Full-Fintetune or Train-from-Scratch (see Table IV), thus this form of fine-tuning has a lower sample complexity and thus does not need as large a privacy budget. On the other hand, given a tight privacy budget, both Train-from-Scratch and Full-Finetune fail to train the model to reach high accuracy due to an excessive number of training parameters and the added noise in DP-SGD.
- Although MR learns an order of magnitude more parameters than partial finetuning (see Table IV), its accuracy



(a) Centralized setting

(b) Federated setting

Fig. 3: Privacy-accuracy tradeoff for CIFAR-10; f_S = ResNet-50

TABLE I: Comparison of federated learning with DP-SGD in the IID setting. The reported number is test accuracy (%).

Models Data	Resnet18	Resnet50	Resnet152	ResNext50	ResNext101
CIFAR-10 (ϵ = 1.04)	MR = 74.68 BL-PF = 57.7 BL-TS = 16.01 BL-FF = 17.58	MR = 79.08 BL-PF = 58.31 BL-TS = 14.4 BL-FF = 16.9	MR = 83.45 BL-PF = 24.96 BL-TS = 10.45 BL-FF = 13.8	MR = 81.7 BL-PF = 46.15 BL-TS = 16.52 BL-FF = 20.25	MR = 87.55 BL-PF = 46.13 BL-TS = 10.0 BL-FF = 10.0
Oxford-IIIT (ϵ = 5.29)	MR = 72.55 BL-PF=67.15 BL-TS = 3.63 BL-FF = 5.10	MR = 73.3 BL-PF = 67.3 BL-TS = 2.83 BL-FF = 2.59	MR = 79.59 BL-PF = 34.88 BL-TS = 2.48 BL-FF = 2.72	MR = 79.64 BL-PF = 58.32 BL-TS = 2.9 BL-FF = 3.0	MR = 80.8 BL-PF = 18.07 BL-TS = 2.99 BL-FF = 2.73
Blood-MNIST (ϵ =1.96)	MR = 65.7 BL-PF = 63.19 BL-TS = 53.08 BL-FF = 52.32	MR = 62.7 BL-PF = 53.3 BL-TS = 42.99 BL-FF = 33.0	MR = 60.15 BL-PF = 37.01 BL-TS = 45.16 BL-FF = 19.47	MR = 67.46 BL-PF = 56.7 BL-TS = 33.7 BL-FF = 20.98	MR = 59.9 BL-PF = 37.7 BL-TS = 19.5 BL-FF = 29.5

TABLE II: Performance evaluation in the IID setting across 3 runs (mean ± standard deviation)

Models Data	Resnet18	Resnet50	Resnet152	ResNext50	ResNext101
CIFAR-10	MR = 75.58 ± 0.15 BL-PF = 58.69 ± 0.15	MR = 78.86 ± 0.22 BL-PF = 55.92 ± 2.49	MR = 83.90 ± 0.44 BL-PF = 26.93 ± 1.54	81.27± 0.33 45.69 ± 1.139	88.12± 0.51 47.37± 1.82

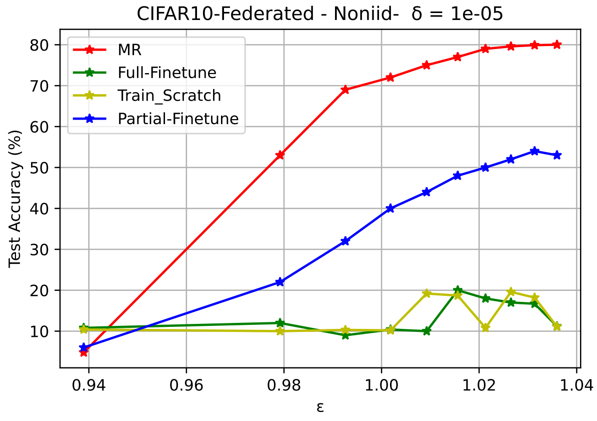


Fig. 4: Non-IID setting (Quantity Skew); f_S = ResNet-50

TABLE III: Performance evaluation in the Non-IID setting (Quantity Skew). The reported numbers are the test accuracies.

Models / Data	Resnet18	Resnet50	ResNext50

CIFAR-10
(ϵ = 1.04)
	MR = 79.68
	BL-PF = 56.72
	BL-TS =15.07
	BL-FF =19.55
		MR = 80.3
		BL-PF = 56.5
		BL-TS = 16.8
		BL-FF = 12.18
			MR = 81.2
			BL-PF = 18.16
			BL-TS = 10.45
			BL-FF = 49.8

outperforms partial finetuning.The observed superior performance of MR over these baselines in the federated setting is consistent with the findings in the centralized non-private setting as reported in previous works [9], [13], [14]. As shown in [13], the accuracy of MR in the noise-free setting can be explained by its ability to align the representations of the target domain to that of the source domain used to train the source model. We discuss an intuition as to why MR additionally displays a superior privacy-utility tradeoff to the baselines in Section VII.

- It was also observed (see Table I) that deeper source models in model reprogramming do better than shallow source models. For instance, on the Oxford-IIIT

TABLE IV: Comparison of number of trainable parameters

Models Data	Resnet18	Resnet50	Resnet152	ResNext50	ResNext101
CIFAR-10	MR = 160538 BL-PF = 20490 BL-TS = 11181642 BL-FF = 11181642	MR = 160538 BL-PF = 20490 BL-TS = 23528522 BL-FF = 23528522	MR = 160538 BL-PF = 20490 BL-TS = 58164298 BL-FF = 58164298	MR = 160538 BL-PF = 20490 BL-TS = 23000394 BL-FF = 23000394	MR = 160538 BL-PF = 20490 BL-TS = 81426762 BL-FF = 81426762
Oxford-IIIT	MR = 187565 BL-PF = 75813 BL-TS = 1195493 BL-FF = 1195493	MR = 187565 BL-PF = 75813 BL-TS = 23583845 BL-FF = 23583845	MR = 187565 BL-PF = 75813 BL-TS = 58219621 BL-FF = 58219621	MR = 187565 BL-PF = 75813 BL-TS = 23055717 BL-FF = 23055717	MR = 187565 BL-PF = 75813 BL-TS = 81482085 BL-FF = 81482085
Blood-MNIST	MR = 158536 BL-PF = 16392 BL-TS = 1180616 BL-FF = 1180616	MR = 158536 BL-PF = 16392 BL-TS = 23524424 BL-FF = 23524424	MR = 158536 BL-PF = 16392 BL-TS = 58160200 BL-FF = 58160200	MR = 158536 BL-PF = 16392 BL-TS = 22996296 BL-FF = 22996296	MR = 158536 BL-PF = 16392 BL-TS = 81422624 BL-FF = 81422624

dataset, ResNext101 shows a performance improvement of 62.73% and ResNet152 has a 44.71% improvement on baselines compared to shallower models like ResNet18 and ResNet50 that have improvements under 10%. We investigate this further in Section VI.

- Similarly on CIFAR-10 (see Table I), ResNet152 shows an improvement of 58.49% and ResNext50 has a 61.45% improvement compared to shallower models like ResNet50 and ResNet18 that show relatively smaller improvements over the best baseline results for a given budget. The results suggest that MR is a more efficient approach for exploiting pre-trained large neural networks.

VI. Ablation Study

A. Multiple Clients

The multiple client scenarios are simulated by increasing the number of clients and keeping the client source models and local epochs constant. Particularly, the rounds of communication are kept at 20 and each client does 3 local iterations on CIFAR-10 dataset and ResNet50 as the source model. We simulate this for the IID setting with each client having the same number of samples.

Figure 5 shows 5 clients scenario, Figure 6 shows the 10 clients scenario, and Figure 7 shows the 20 clients scenario. From the figures, it can be clearly observed that Reprogrammable-FL outperforms baselines when clients increase in the IID setting.

B. Multiple Local Iterations

We simulate the effect of multiple local iterations with a fixed number of clients on the CIFAR-10 dataset and ResNet50 as the source model. Data is distributed among 5 clients and 20 rounds of communication are performed.

Figure 8 shows that in the IID setting, for a fixed budget and increasing the number of local iterations we can see an improvement in the privacy accuracy tradeoff. For instance, see that for a budget of $\epsilon = 1$ Reprogrammable-FL provides an almost 50 % improvement in test accuracy when local iteration increases from 1 to 5.

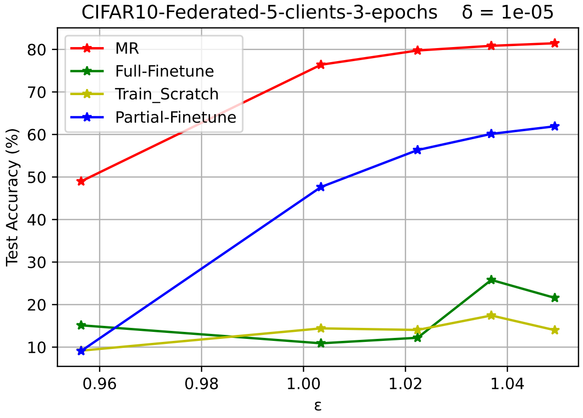


Fig. 5: 5 Clients in IID setting; f_S = ResNet-50

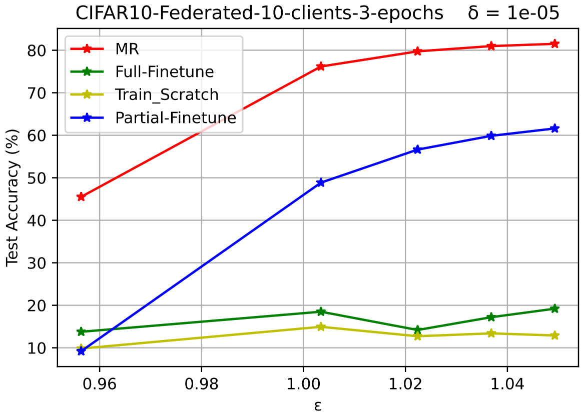


Fig. 6: 10 Clients in IID setting; f_S = ResNet-50

We also report that for 10 local epochs and 5 clients after 1 round of communication, the test accuracies are 80.09/14.59/13.22/12.25 (%) for MR/BL-PF/BL-TS/BL-FF for $\epsilon = 0.97$. In addition, for 20 epochs and 5 clients after 1 round of communication, the test accuracies are 83.49/12.4/16.13/18.49(%) for MR/BL-PF/BL-TS/BL-FF for $\epsilon = 0.99$.

Thus, for a given budget with each client performing higher local iterations, Reprogrammable-FL still outperforms baselines.

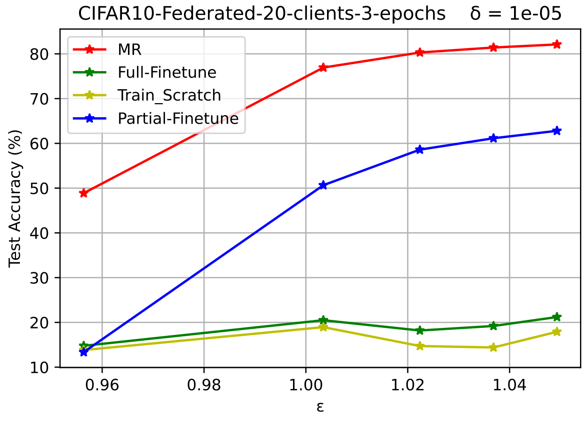


Fig. 7: 20 Clients in IID setting; f_S = ResNet-50

C. Deep vs Shallow Models

We also see in Figure 9 that the performance gap between Reprogrammable-FL and the best baseline is larger for deeper models than shallow models. While partial finetune suffers in performance when deeper models are used, Reprogrammable-FL maintains its high utility when being trained privately.

D. Non-IID – Class Imbalance Scenario

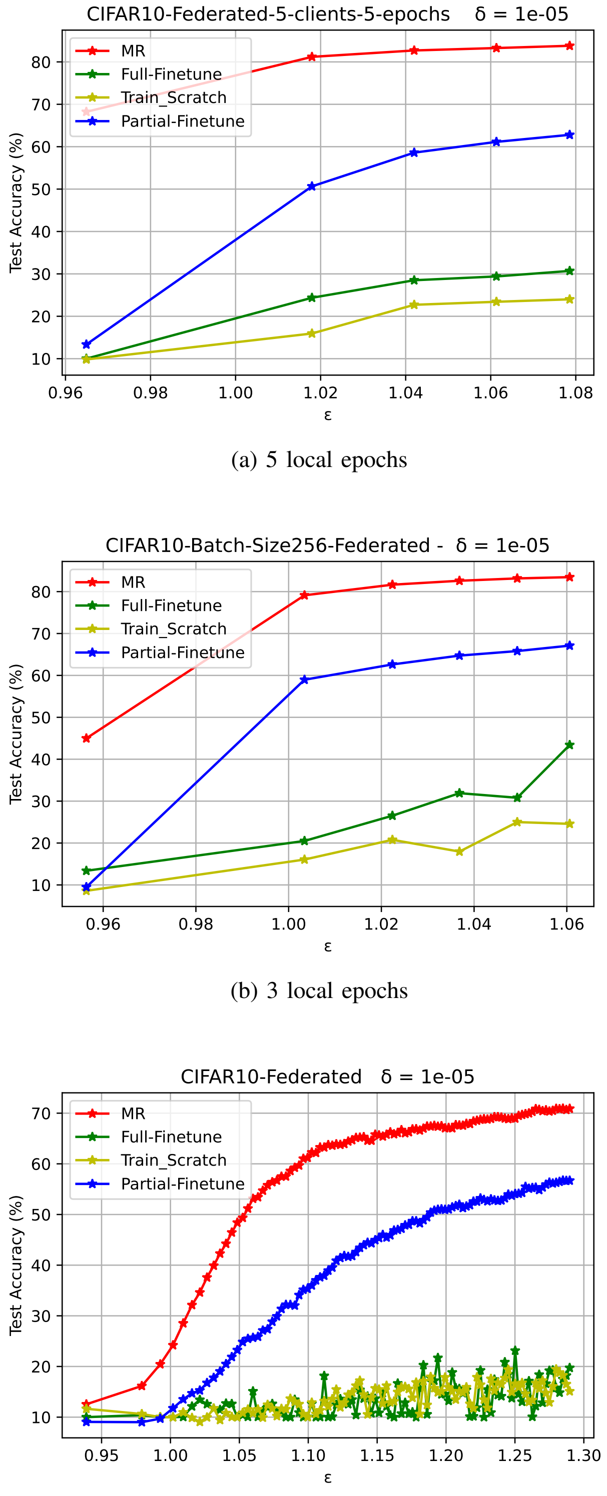
In addition to the quantity skew non-IID setting demonstrated in the paper, Figure 10 demonstrates a class label imbalance scenario using ResNet-50 as the source model for the clients and CIFAR-10 as the dataset. In this setting, each of the clients holds 3,3,4 nonoverlapping CIFAR-10 classes. This is done for 50 rounds and 3 local epochs with 3 clients. We can clearly see that in the class imbalance scenario, Reprogrammable-FL also outperforms baselines.

E. Impact of Batch Size Selection

Figure 11 shows the performance of using different batch sizes in Reprogrammable-FL and the competing baselines. Reprogrammable-FL outperforms the baselines on all smaller batch sizes evaluated. This experiment was done on 3 local clients, with each client doing 3 local epochs. The number of global rounds is 50. Figure 12 shows that larger batch sizes achieve better utility per global round in Reprogrammable-FL over smaller batch sizes.

F. Area under ROC Curve

In addition to measuring the accuracy of the classifier, an important characteristic of a good classifier is its ability to distinguish between various classes. Ideally, for a classifier, this would mean that a model has no false positives and negatives. A comparatively better classifier would thus be such that it is able to distinguish between classes with fewer false positive/negatives. A metric that measures this is the area under the receiver operating characteristic (ROC) [32]. Values closer to 1 mean that the classifier is better at distinguishing between classes. A value of 0.5 means that the model is merely guessing between the true and false class thus in effect being a futile model.

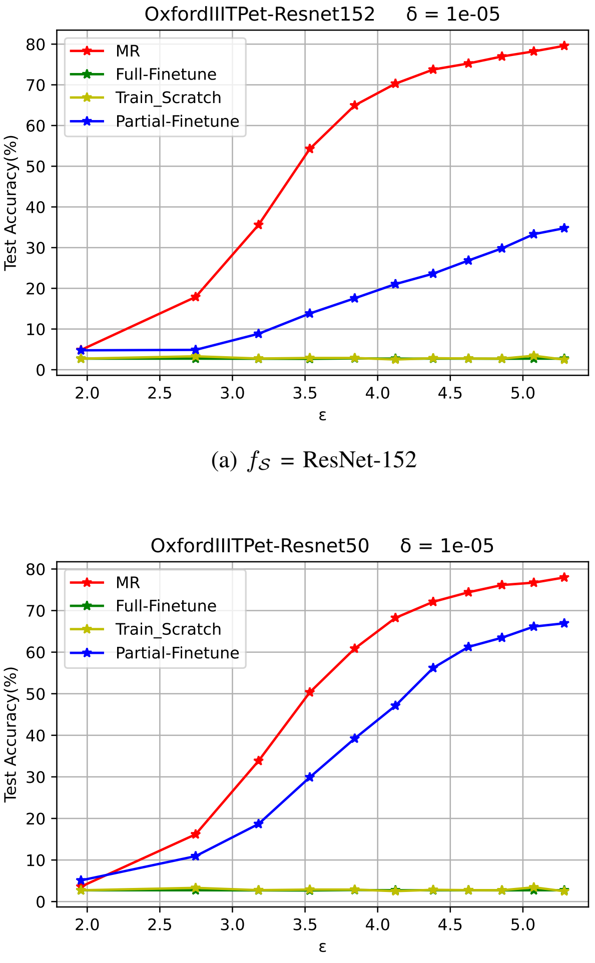


(a) 5 local epochs

(b) 3 local epochs

(c) 1 local epoch

Fig. 8: Comparison of privacy-accuracy tradeoff for CIFAR-10 with 3 clients and varying local epochs; f_S = ResNet-50



(a) $f_{\mathcal{S}}$ = ResNet-152

(b) $f_{\mathcal{S}}$ = ResNet-50

Fig. 9: Performance comparison of privacy-accuracy tradeoff for Oxford-IIIT when using Deeper/Shallow source models

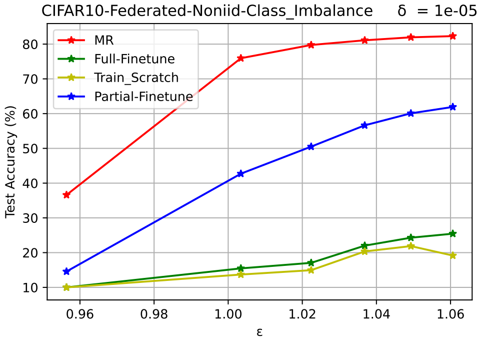
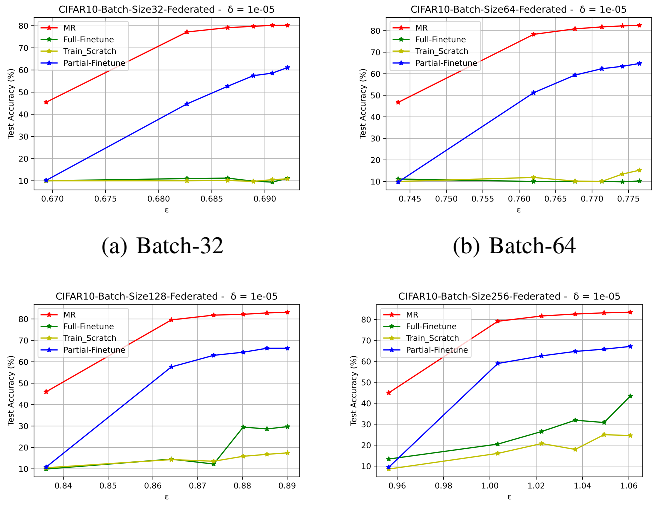


Fig. 10: Non-IID (Class Imbalance) setting; $f_{\mathcal{S}}$ = ResNet-50

For a multi-class classification problem, this metric is computed by doing a one vs rest comparison with all the other classes. The score reported is the average score for all classes.

Table V shows the AUROC performance for different source models on the CIFAR-10 dataset. It can be observed that Reprogrammable-FL has better distinguishability for various classes when trained privately.



(a) Batch-32

(b) Batch-64

(c) Batch-128

(d) Batch-256

Fig. 11: Comparison of privacy-accuracy tradeoff for CIFAR-10 with with different batch sizes; $f_{\mathcal{S}}$ = ResNet-50

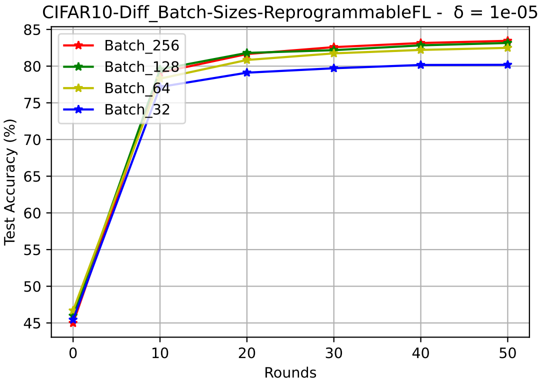


Fig. 12: Different batch sizes on Reprogrammable-FL; $f_{\mathcal{S}}$ = ResNet-50

TABLE V: Performance evaluation in the IID setting using AUROC

Models / Data	Resnet18	Resnet50	ResNext50

CIFAR-10
(ϵ = 1.04)
	MR = 0.971
	BL-FF = 0.745
	BL-TS = 0.736
	BL-PF = 0.910
		MR = 0.978
		BL-FF = 0.77
		BL-TS = 0.656
		BL-PF = 0.909
			MR = 0.982
			BL-FF = 0.77
			BL-TS = 0.70
			BL-PF = 0.88

G. Area under the curve

The improvement in the privacy-accuracy trade-off provided by Reprogrammable-FL can also be illustrated by the area under the privacy-utility curves. Concretely, we evaluate the area under the curve for Figure 8b. Table VI shows this com-

TABLE VI: Area under the curve (Accuracy % × the unit of privacy budget) performance evaluation in the IID setting

Models / Data	ResNet50

CIFAR-10	MR = 382.0
	BL-FF = 70.2
	BL-TS = 60.7
	BL-PF = 275.5

parison. We can observe that Reprogrammable-FL provides a better tradeoff compared to baselines.

VII. Theoretical Considerations

We observed significant improvement, with respect to privacy-utility tradeoff, of MR over transfer learning in the DP-FL setting. Theoretical justifications for the superior performance of MR over transfer learning are to our knowledge, non-existent in the literature even in the centralized setting. The clearest theoretical characterization of the performance of MR is given in [13], which establishes that when the source and target domain representations are well-aligned in the sense of Wasserstein distance, the generalization error of MR on the target domain is bounded. Note that this result does not argue or imply that MR should outperform transfer learning.

Intuitively, if the sample complexity of transfer learning is smaller than that of MR, we expect that MR should have a superior privacy-utility tradeoff, as a lower sample complexity suggests MR needs to "touch" less data to achieve a given generalization gap. The work [33] suggests that MR does have a smaller sample complexity than transfer learning. That work considers the sample complexity required to adapt a shared representation $h(x)$ from a class $\mathcal{H}$, learned on several prior tasks, to a new task by learning a function $f(x)$ in a class $\mathcal{F}$ and taking $f \circ h$ as the hypothesis for the new task. Their main result [33, Theorem 3] implies that, if the complexity of $\mathcal{F}$ is smaller than that of $\mathcal{H}$, a generalization gap of ς can be achieved on the target task by using

$$n = \mathcal{O}\left(\frac{\mathcal{C}(\mathcal{F})}{\varsigma^2}\right)$$

samples from the target domain to learn f, where $\mathcal{C}(\mathcal{F})$ is the Gaussian complexity of $\mathcal{F}$.

Partial fine-tuning, the most competitive baseline, exactly fits into this theoretical framework, where f is chosen from $\mathcal{F}_{\text{BL-PF}}$ corresponding to the layers that are fine-tuned. Meanwhile, MR almost fits into this theoretical framework, where f is chosen from $\mathcal{F}_{\text{MR}}$ corresponding to the output transformation. Because the multi-layer neural networks in $\mathcal{F}_{\text{BL-PF}}$ are more complex than the class of logistic regressions in $\mathcal{F}_{\text{MR}}$, we expect that MR will have smaller sample complexity, and thus a superior privacy-utility tradeoff.

The gap in making this argument rigorous lies in the fact that MR also uses an input transformation, so the hypothesis learned on the target task takes the form $f \circ h \circ g$. However, g is a linear transformation, so we expect that a rigorous exploration of the sample complexity of MR vs. that of partial fine-tuning will exhibit the behavior predicted using the results of [33]. We leave a rigorous confirmation of this intuition for future work.

VIII. Conclusion

In this paper, we proposed a new training method for federated learning with differential privacy, which we call Reprogrammable-FL. In contrast to finetuning a pre-trained model, Reprogrammable-FL attaches input and output transformation laters and only trains their associated parameters while keeping the pre-trained model weights intact. Our empirical evaluation shows that this non-intrusive approach provides stronger utility-privacy improvements than existing baselines such as full/partial finetuning in transfer learning and training from scratch. Given the same privacy budget, consistent accuracy improvements are observed across a variety of datasets and pre-trained models. We also find that Repogrammable-FL achieves even larger performance improvements over baselines when reprogramming a model with deeper architecture. Given that this paper is the first study to demonstrate the high accuracy gain of model reprogramming in differentially private FL, we believe our results and findings will set a new benchmark for FL and shed new light on future research in private FL.

References

[1] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," *Advances in neural information processing systems*, vol. 32, 2019.

[2] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in *2019 IEEE symposium on security and privacy (SP)*. IEEE, 2019, pp. 739–753.

[3] X. Jin, P.-Y. Chen, C.-Y. Hsu, C.-M. Yu, and T. Chen, "CAFE: Catastrophic data leakage in vertical federated learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 994–1006, 2021.

[4] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[5] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 308–318.

[6] K. Ligett, S. Neel, A. Roth, B. Waggoner, and S. Z. Wu, "Accuracy first: Selecting a differential privacy level for accuracy constrained erm," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[7] A. El Ouadrhiri and A. Abdelhadi, "Differential privacy for deep and federated learning: A survey," *IEEE Access*, vol. 10, pp. 22 359–22 380, 2022.

[8] D. Yu, S. Naik, A. Backurs, S. Gopi, H. A. Inan, G. Kamath, J. Kulkarni, Y. T. Lee, A. Manoel, L. Wutschitz *et al.*, "Differentially private fine-tuning of language models," *arXiv preprint arXiv:2110.06500*, 2021.

[9] P.-Y. Chen, "Model reprogramming: Resource-efficient cross-domain machine learning," *arXiv preprint arXiv:2202.10629*, 2022.

[10] L. Torrey and J. Shavlik, "Transfer learning," in *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*. IGI global, 2010, pp. 242–264.

[11] G. F. Elsayed, I. Goodfellow, and J. Sohl-Dickstein, "Adversarial reprogramming of neural networks," in *International Conference on Learning Representations*, 2019.

[12] R. Vinod, P.-Y. Chen, and P. Das, "Reprogramming language models for molecular representation learning," in *Annual Conference on Neural Information Processing Systems*, 2020.

[13] C.-H. H. Yang, Y.-Y. Tsai, and P.-Y. Chen, "Voice2series: Reprogramming acoustic models for time series classification," in *International Conference on Machine Learning*. PMLR, 2021, pp. 11 808–11 819.

[14] Y.-Y. Tsai, P.-Y. Chen, and T.-Y. Ho, "Transfer learning without knowing: Reprogramming black-box machine learning models with scarce data and limited resources," in *International Conference on Machine Learning*. PMLR, 2020, pp. 9614–9624.

[15] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, A. Singh and J. Zhu, Eds., vol. 54. PMLR, 2017, pp. 1273–1282.

[16] M. Mohri, G. Sivek, and A. T. Suresh, "Agnostic federated learning," in *Proceedings of the 36th International Conference on Machine Learning*, vol. 97. PMLR, 09–15 Jun 2019, pp. 4615–4625.

[17] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[18] S. Caldas, J. Konečny, H. B. McMahan, and A. Talwalkar, "Expanding the reach of federated learning by reducing client resource requirements," *arXiv preprint arXiv:1812.07210*, 2018.

[19] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Robust and communication-efficient federated learning from non-iid data," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 9, pp. 3400–3413, 2019.

[20] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1175–1191.

[21] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[22] H. B. McMahan, G. Andrew, U. Erlingsson, S. Chien, I. Mironov, N. Papernot, and P. Kairouz, "A general approach to adding differential privacy to iterative training procedures," *arXiv preprint arXiv:1812.06210*, 2018.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[24] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1492–1500.

[25] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *arXiv preprint arXiv:1806.00582*, 2018.

[26] D. Tran, J. Liu, M. W. Dusenberry, D. Phan, M. Collier, J. Ren, K. Han, Z. Wang, Z. Mariet, H. Hu *et al.*, "Plex: Towards reliability using pretrained large model extensions," *arXiv preprint arXiv:2207.07411*, 2022.

[27] Q. Li, Y. Diao, Q. Chen, and B. He, "Federated learning on non-iid data silos: An experimental study," in *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, 2022, pp. 965–978.

[28] A. Acevedo, A. Merino, S. Alférez, Á. Molina, L. Boldú, and J. Rodellar, "A dataset of microscopic peripheral blood cell images for development of automatic recognition systems," *Data in brief*, vol. 30, 2020.

[29] J. Yang, R. Shi, D. Wei, Z. Liu, L. Zhao, B. Ke, H. Pfister, and B. Ni, "Medmnist v2: A large-scale lightweight benchmark for 2d and 3d biomedical image classification," *arXiv preprint arXiv:2110.14795*, 2021.

[30] Z. Luo, D. J. Wu, E. Adeli, and L. Fei-Fei, "Scalable differential privacy with sparse network finetuning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 5059–5068.

[31] A. Kurakin, S. Chien, S. Song, R. Geambasu, A. Terzis, and A. Thakurta, "Toward training at imagenet scale with differential privacy," *arXiv preprint arXiv:2201.12328*, 2022.

[32] D. Zhu, X. Wu, and T. Yang, "Benchmarking deep auroc optimization: Loss functions and algorithmic choices," *arXiv preprint arXiv:2203.14177*, 2022.

[33] N. Tripuraneni, M. Jordan, and C. Jin, "On the theory of transfer learning: The importance of task diversity," *Advances in Neural Information Processing Systems*, vol. 33, pp. 7852–7862, 2020.