

Table 3: Dataset Statistics

Dataset	Train Clients	Eval Clients	Test Clients	Samples/Client Mean	Std
CIFAR-10	100	10	10	500	63
Stack Overflow	10,815	378	1,115	5,821	34,229
FEMNIST	3,150	350	350	272	67
Reddit pushshift.io	9,403	4,352	4,352	34	63

A EXPERIMENT DETAILS

A.1 DATASETS AND MODELS

CIFAR-10. We evaluate a multi-class image classification problem on CIFAR-10 Krizhevsky et al. (2009) using a SqueezeNet Iandola et al. (2016). We normalize the images by the dataset mean and standard deviation. Following Hsu et al. (2019), we partition the dataset using a Dirichlet distribution with parameter 0.1. The statistics on the number of clients and examples in both the training and test splits of the datasets are in Table 3.

Stack Overflow. Stack Overflow consists of questions and answers from Stack Overflow. We experiment a next-word-prediction task using a DistilGPT-2 model with a casual LM head. We perform padding and truncation to ensure that each sentences have 25 words. We then use a GPT-2 tokenizer to encode the tokens.

FEMNIST. Federated EMNIST-62 (FEMNIST) consists of digits and English characters, totaling 62 classes. We evaluate a multi-class image classification problem on the federated version Caldas et al. (2018) which partitions the digits by the writer and filter out clients that less than one example. As for the model, we use a SqueezeNet 1.0 Iandola et al. (2016). Since FEMNIST contains grayscale images, we replicated the one channel value into three channels with the same values.

Reddit pushshift.io. Reddit pushshift.io contains preprocessed comments posted on the social network on December 2017. The dataset consists of 1,660,820 users totaling 56,587,343 tokens. Due to limited compute capabilities, we sub-sampled 9,403 users for training, 4352 for evaluation and 4352 for test. We evaluate a next-word-prediction task using a CharLM Kim et al. (2016) model.

A.2 IMPLEMENTATION DETAILS

We implemented all algorithms in Pytorch (Paszke et al., 2017) and evaluated them on a cluster of machines, each with eight NVidia V100 GPUs. We evaluate our experiments in FLSim³. For all experiments, we tune hyperparameters using Bayesian optimization (Snoek et al., 2012). We select the best hyperparameters based the final accuracy after a fixed number of rounds for each dataset.

A.3 HYPER-PARAMETERS

Below, we show the range for the client learning rate (η_ℓ), server learning rate (η_g) and proximal term (μ). We fixed server momentum for FEDAVG and FEDADAM β_1 at 0.9.

$$\begin{aligned}
 \beta &\in \{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.99\} \\
 \eta_\ell &\in [1 \cdot 10^{-6}, 10] \\
 \eta_g &\in [1 \cdot 10^{-6}, 10] \\
 \mu &\in [0.001, 0.01, 0.1, 0.5]
 \end{aligned}$$

³<https://github.com/facebookresearch/FLSim>

CIFAR-10		
Algorithms	Random	Pretrained
FedAdam Proximal	(0.001, 0.001, 0.5)	(0.001, 0.01, 0.01)
FedAdam SGD	(0.001, 0.001)	(0.0001, 0.01)
FedAdam MineLite	(0.0001, 0.00001)	(10, 0.000001)
FedAvg Proximal	(1, 0.01, 0.1)	(1, 0.01, 0.01)
FedAvg SGD	(1, 0.01)	(0.1, 0.001)
FedAvg MineLite	(1, 0.1)	(1, 0.01)
FedAvgM Proximal	(1, 0.01, 0.001)	(1, 0.01, 0.5)
FedAvgM SGD	(1, 0.01)	(1, 0.001)
FedAvgM MineLite	(0.001, 0.1)	(1, 0.0001)

Table 4: The (η_g, η_ℓ) or (η_g, η_ℓ, μ) for CIFAR-10 that achieve the metrics in Figure 1.

FEMNIST		
Algorithms	Random	Pretrained
FedAdam Proximal	(1e-06, 0.000862, 0.5)	(0.008043, 0.000109, 0.001)
FedAdam SGD	(1e-06, 0.001219)	(0.000877, 0.000125)
FedAdam MineLite	(1e-06, 10.0)	(10.0, 1e-06)
FedAvg Proximal	(0.002593, 1.88399, 0.001)	(0.012892, 0.58932, 0.1)
FedAvg SGD	(0.001177, 1.327235)	(0.000326, 3.810216)
FedAvg MineLite	(1.0, 0.01)	(0.1, 0.01)
FedAvgM Proximal	(2e-06, 100.0, 0.01)	(0.000451, 3.455496, 0.01)
FedAvgM SGD	(0.00133, 2.087185)	(0.001842, 2.677946)
FedAvgM MineLite	(10.0, 1e-05)	(0.1, 0.001)

Table 5: The (η_g, η_ℓ) or (η_g, η_ℓ, μ) for FEMNIST that achieve the metrics in Figure 1.

Stack Overflow		
Algorithms	Random	Pretrained
FedAdam Proximal	(0.02592, 0.000323, 0.01)	(0.001852, 0.000204, 0.5)
FedAdam SGD	(0.008602, 9.2e-05)	(0.006503, 5.3e-05)
FedAdam MineLite	(0.1, 10)	(0.1, 0.001)
FedAvg Proximal	(0.064397, 0.602131, 0.001)	(0.041602, 1.0, 0.01)
FedAvg SGD	(0.003366, 1.0)	(0.028053, 1.0)
FedAvg MineLite	(0.01, 10)	(1e-06, 100)
FedAvgM Proximal	(0.024937, 0.206152, 0.5)	(0.035737, 0.084879, 0.5)
FedAvgM SGD	(0.133601, 0.338568)	(0.026972, 0.101941)
FedAvgM MineLite	(100.0, 100)	(0.1, 0.01)

Table 6: The (η_g, η_ℓ) or (η_g, η_ℓ, μ) for Stack Overflow that achieve the metrics in Figure 1.

Algorithms	Reddit	
	Random	Pretrained
FedAdam Proximal	(0.01, 1.0, 0.01)	(1e-06, 0.1, 0.01)
FedAdam SGD	(100.0, 100.0)	(0.0001, 0.1)
FedAdam MineLite	(100.0, 1e-06)	(0.0001, 1e-06)
FedAvg Proximal	(1.0, 1.0, 0.01)	(1.0, 0.1, 0.01)
FedAvg SGD	(0.001, 10.0)	(100.0, 0.001)
FedAvg MineLite	(100.0, 100.0)	(100.0, 0.0001)
FedAvgM Proximal	(0.01, 1.0, 0.01)	(100.0, 1e-05, 0.01)
FedAvgM SGD	(100.0, 100.0)	(0.0001, 0.1)
FedAvgM MineLite	(100.0, 10.0)	(100.0, 1e-06)

Table 7: The (η_g, η_ℓ) or (η_g, η_ℓ, μ) for Reddit that achieve the metrics in Figure 1.Table 8: Comparison of characteristics considered in previous work and the methods analyzed in this paper. Notation: *NA* = Normalized Averaging, *LS* = Non-identical Local Steps, *GM* = Global Momentum, *AS* = Adaptive Server Learning Rate, *GS* = Apply Server Optimizer State Locally.

	NA	LS	GM	AS	GS
FEDAVG NOVA	✓	✓	✗	✗	✗
FEDAVG PROXIMAL	✗	✓	✗	✗	✗
FEDAVG SGD	✗	✓	✗	✗	✗
FEDAVG GD	✗	✗	✗	✗	✗
FEDAVG MIMELITE	✗	✓	✗	✗	✓
FEDAVGM NOVA	✓	✓	✓	✗	✗
FEDAVGM PROXIMAL	✗	✓	✓	✗	✗
FEDAVGM SGD	✗	✓	✓	✗	✗
FEDAVGM GD	✗	✗	✓	✗	✗
FEDAVGM MIMELITE	✗	✓	✓	✗	✓
FEDADAM NOVA	✓	✓	✓	✓	✗
FEDADAM PROXIMAL	✗	✓	✓	✓	✗
FEDADAM SGD	✗	✓	✓	✓	✗
FEDADAM GD	✗	✗	✓	✓	✗
FEDADAM MIMELITE	✗	✓	✓	✓	✓

A.4 IMPACT OF PRE-TRAINING ON HYPER-PARAMETERS

In this section, we show the combinations of server learning rate (η_g) and client learning rate (η_ℓ) on CIFAR-10 with SGD at the client and various server optimizers. Overall, Figure 7 shows that pre-training requires to smaller client learning rates, roughly one order of magnitude smaller compared to random initialization. Moreover, using FEDADAM is more robust of the changes in client side learning rates.

B ALGORITHMS SUMMARY

The FEDOPT framework is shown in Algorithm 1.

In this section, we summarize the differences between the SERVEROPT and CLIENTOPT combinations used in our experiments.

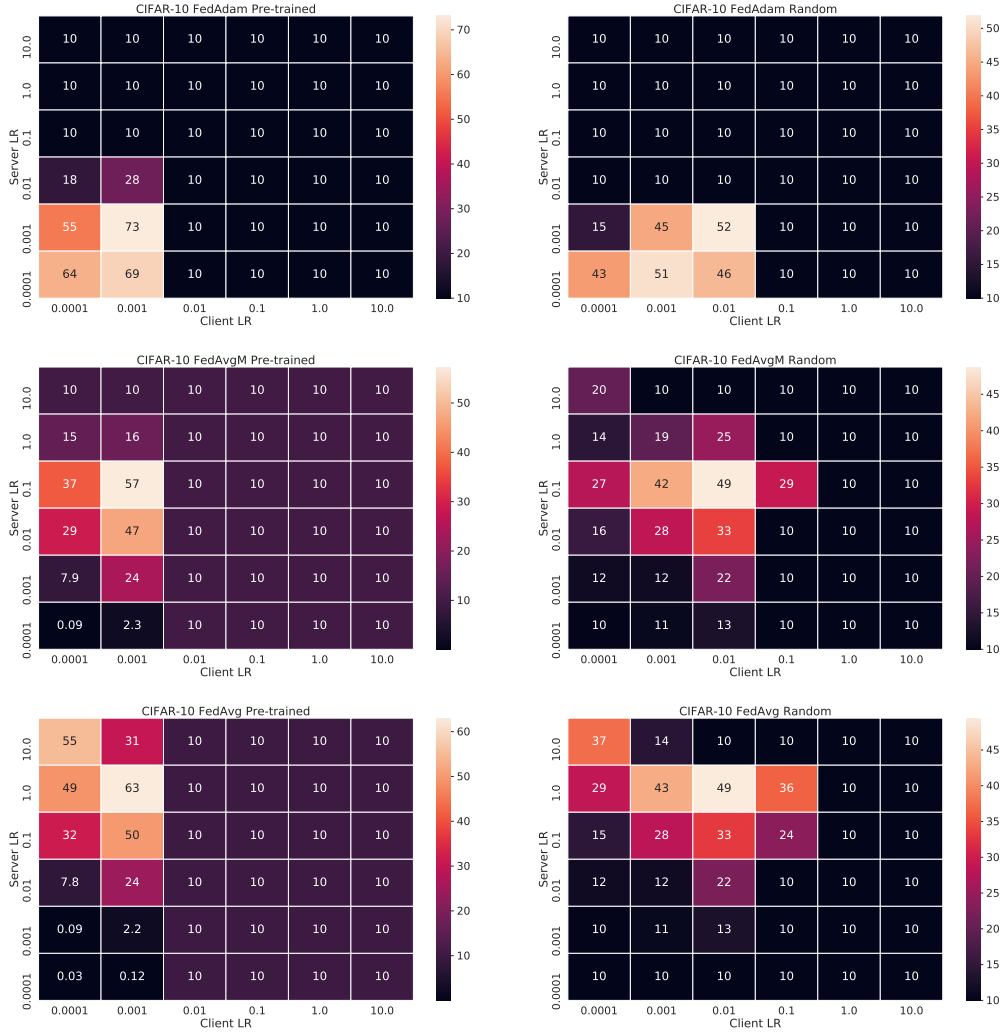


Figure 7: The test accuracy of FEDADAM, FEDAVGM, and FEDAVG for various client and server learning rates combination on CIFAR-10. We fixed momentum for FEDAVGM and β_1 for FEDADAM to be 0.9.

B.1 PRE-TRAINING CHARLM

To pre-train CharLM, we train an the CharLM model Kim et al. (2016) using a vocab size of 5000. We train the model on Wikitext-103 for 100 epochs using AdamW as the optimizer, learning rate = 0.001, weight-decay = 0.00001, and eps = 1e-8. We will release the code and the pre-train models in the camera-ready version.

C ADDITIONAL RESULTS

C.1 PRE-TRAINING’S IMPACT ON VARYING LEVELS OF HETEROGENEITY

We show the result for varying levels of non-IIDness on CIFAR-10 below in Figure 9. We follow Hsu et al. (2019) generate the different splits and partition the dataset using a Dirichlet distribution with $\alpha \in [0.1, 1.0, 10]$. As $\alpha \rightarrow \infty$, the data becomes more IID. We train a Squeezenet model for 1000 rounds using FEDADAM at the server and SGD at the client.

α	Pre-trained	Random
0.1	75.9	50.6
1.0	78.1	66.8
10	80.1	72.8

Table 9: Average accuracy for varying levels of heterogeneity on CIFAR-10. We train a Squeezenet model for 1000 rounds using FEDADAM at the server and SGD at the client.

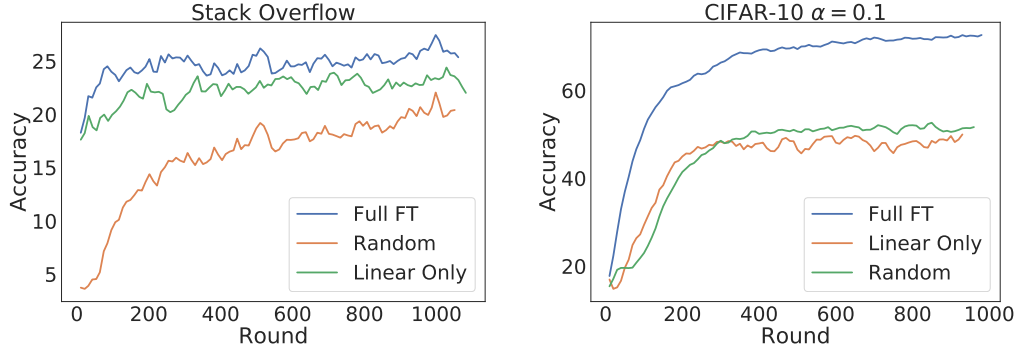


Figure 8: Average accuracy for full fine-tuning, random, and last layer only on Stack Overflow and CIFAR-10.

C.2 FINE-TUNING ONLY THE LAST LAYER

In this section, we present the results for fine-tuning only the last linear layer rather in the model as commonly done in practice. Figure 8 shows that fine-tuning only the last layer might not yield optimal model quality and should be consider carefully. While fine-tuning only the last layer can achieve close to full fine-tuning on Stack Overflow, the performance is much worse on CIFAR-10.

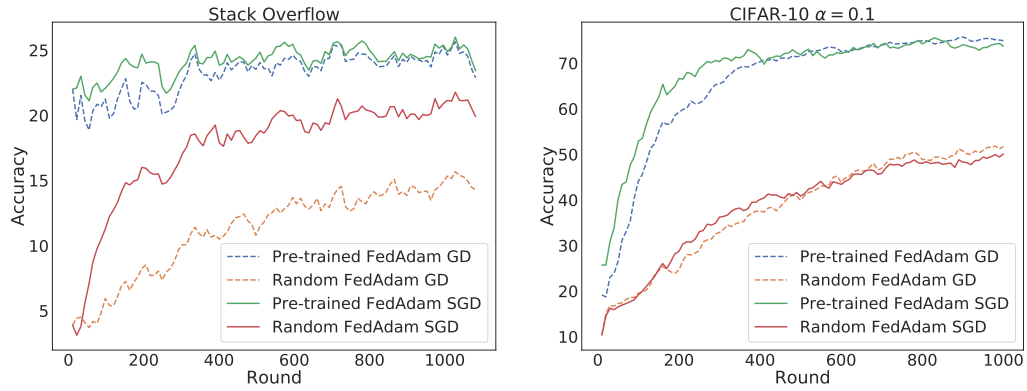


Figure 9: The average accuracy for Stack Overflow and CIFAR-10 comparing FEDADAM with SGD and FEDADAM with full-batch gradient descent (GD).