Supplemental Material for Federated Learning Over Images: Vertical Decompositions and Pre-Trained Backbones Are Difficult to Beat

Erdong Hu[1], Yuxin Tang[1], Anastasios Kyrillidis, Chris Jermaine
Rice University
{eh51, yuxin.tang, anastasios, cmj4}@rice.edu

1. Data Processing Details

To reiterate from the main paper, the datasets we used in our experiments are CUB-200-2011 (Birds) [10], Stanford Cars (Cars) [4], (3) VGGFlowers (Flwrs) [7], (4) Aircraft (Aircrft) [6], (5) Describable Textures (Textre) [1], and (6) CIFAR 100 (CFAR) [5]. All datasets are partitioned along their original train-test split. For datasets with an additional validation set, in the case of Textre, Aircrft, and Flwrs, we merge this with the test dataset. The Aircrft dataset has different levels of aircraft type - Manufacturer, family, variant. Variant is the specific model of aircraft and we chose this as the label for highest granularity.

On the Birds, Flwrs, Aircrft, and Textre datasets we perform a randomized cropping followed by a 224×224 resizing on the train datasets, then a randomized horizontal flip, and a normalization of the RGB colors. On the test datasets we resize the images to 256×256 and then center crop them to 224×224, and again normalize the colors. For Cars and CFAR datasets we use the same processing we used for the test datasets for both the train and test datasets. These images are then converted to tensors to be used as inputs.

For inputs to all algorithms except FedFull, we input these image tensors into pretrained ResNet101 and a DenseNet121 convolutional neural networks. We use the existing versions downloaded from PyTorch, which are trained on the ImageNet dataset. We remove the final linear layer from these CNNs to retrieve the feature tensors as outputs which we concatenate and use as the inputs of most of the evaluated algorithms. ResNet101 produces 1024-dimensional vectors while DenseNet121 produces ones of dimension 2048, so the concatenated feature vector is 3072-dimensional.

To sample the data for N client devices and a dataset with k classes, we generate N vectors of length k, one for each client, drawn from a Dirichlet distribution. More precisely, given concentration parameter α and for a task of k classes, we sample client class distribution vectors $(x_1, \ldots, x_k) \in [0,1]^k$, such that $\sum_{i=1}^{k} x_i = 1$, according to the following probability density function:

$$p_\alpha(x_1, \ldots, x_k) = \frac{1}{B(\alpha)} \prod_{i=1}^{k} x_i^{\alpha-1} \quad (1)$$

where $B(\alpha)$ is a normalization. The concentration parameter α dictates the skewness of data with values closer to 0 being more skewed and larger values generating more i.i.d. samples. More generally the Dirichlet concentration parameter can be assigned per class by replacing α with α_i in Eq. (1), but we use the same value on all classes in the generated client distributions.

For each client we assign 500 images by sampling from each class, with replacement, the proportion given by these generated distribution vectors. The sampling is done with replacement because in many datasets some classes may not have sufficiently many examples, especially in the non-i.i.d. case.

2. Simulation Implementation Details

All of our experiments are done as simulations of synchronous, centralized federated learning. We assume a central server that broadcasts a global model that client devices download and upload for their local training. Our simulator is implemented with PyTorch [8]

Prior to training we initialize the client data distributions via Dirichlet sampling and these samples are fixed for an entire simulation. At the start of a global communication and training round, we randomly select the subset of clients that participate in the training. For IST and ISTProx this is also when the model partitioning is done.

In the centralized learning approximation experiments, we also train the central MLP models on the central server without partitioning to clients. In this case, for consistency of training data, we concatenated the sampled datasets of the participating clients and used this as the training set.

2.1. Models

Except FedFull, all algorithms use a single-hidden layer multilayer perceptron (MLP) as their base model. This con-

[1]Equal contribution.

sists of a linear layer, followed by a BatchNorm (Batch Normalization) layer, then a ReLU activation function, and then another linear layer followed by a softmax. As our pretrained extractor outputs are 3072-dimensional vectors, the input layer of the MLPs are 3072 wide, while the output layer has the number of classes of a particular dataset, e.g. on Birds we would have 200 width output layer.

For all algorithms except for IST, ISTProx, and FedFull, the MLP hidden layer has 1000 neurons. FedFull uses the ResNet18, initialized with the default weights pretrained on the ImageNet dataset and then fine-tuned on the various datasets.

For IST and ISTProx, we find scaling the hidden neurons based on the number of concurrent training sites made sense: 3000 neurons for ten sites, 6000 for 20, 9000 for 30, 18000 for 60, and so on. This ensures that the client models are not too small and in this setup the IST client models share the MLP architecture but with 300 hidden neurons in all cases.

For MOON, we also extract the output after the ReLU activation function as the learned representation used in its contrastive term.

2.2. Training

We use batch size of 32 for all datasets and methods. A single local training round computes one batch. A learning rate of 0.01 is chosen on all algorithms except FedAdam which uses 0.03, with a 10x decay after 50% and 75% of total communication rounds have passed. In the centralized training case, we instead train an entire epoch by passing through all the concatenated data samples once.

Prior to training, we generate estimations of the FLOP usage of a forward or backward pass per batch via the Python *fvcore* [2] package's FLOPCountAnalysis as well as the model size via the *torchinfo* [12] package. After a communication round, we estimate the total bytes transferred by multiplying the model size by the number of participating clients and then double this to account for both downloading and uploading the model to the central server. To compute the FLOP usage after a communication round, the FLOPCountAnalysis gives an estimate of the forward propagation that we multiply by two to account for the backward pass and then by the number of local rounds and the number of participating clients. For MOON this FLOP estimate is also doubled to account for its extra two forward pass computations.

Except FedProx, ISTProx, and FedNova, all methods use a standard stochastic gradient descent optimizer for the local training with a momentum weight of 0.9. After a global communication round, we reinitialize these optimizers to avoid reusing stale momentum from the previous training.

FedProx and ISTProx use a proximal optimizer that also takes as input the initial state of the model before local training is done.

FedNova uses its own optimizer that is based on an implementation from [11].

2.3. Testing and Evaluation

As our goal is to evaluate the generalizability of the aggregated model, the testing dataset is taken as a whole rather than sampled by a Dirichlet distribution as in the case of the training set. Our final accuracy is computed by the proportion of the top-1 predicted classes that match the true label.

These methods may have inconsistent accuracy performance within a few communication rounds. So in computing the final accuracies, we average over a sliding window of 10 communication rounds to smooth out the accuracies.

We reiterate the process detailed in the main paper that we choose the highest accuracy among all algorithms and then take 90% of this accuracy as our threshold. We then identify the smallest GB transferred and GFLOP count needed to reach this 90% threshold.

3. Additional Tables

We include the following tables for additional results of the experiments in the main paper:

- Skewed data, 100 sites, 30% client participation Tab. 1
- Skewed data, 1000 sites, 2% client participation Tab. 2
- Skewed data, 1000 sites, 6% client participation Tab. 3

These tables generally follow the same trends that we observed in the skewed 10% participation experiments in the main paper. Again, we see that IST methods perform best across all three metrics on the Birds, Flwrs, Aircrft, and Textres dataset, but is inconsistent on the Cars and CFAR datasets. Averaging algorithms, in particular FedAvg, fare better on these two datasets. As we also addressed in the main paper, increasing client participation does not consistently nor significantly increase accuracies and generally increases the GFLOP and GB costs.

However, we note that moving from 100 clients to the 1000 client population cases, the final accuracies greatly increase across all algorithms and datasets, and are more closely aligned with the final accuracies of the i.i.d. cases. This effect merits further investigation, although we believe this phenomenon is due to the increased population mitigating the effects of non-i.i.d. data.

4. Hyperparameter Tuning

All hyperparameter tuning was done on the Birds dataset and in the interest of *off-shelf performance* we maintained the same sets of parameters for all other datasets. We have two separate sets of parameters for optimal communication and for FLOP usage due to potential trade-offs when

(a) Final accuracies

	Birds	Cars	Flwrs	Aircrft	Textre	CFAR
FedAvg	61.0	55.6	89.5	40.1	69.3	**65.4**
FedProx	58.0	48.1	89.1	37.7	66.2	60.5
MOON	59.2	50.9	86.4	37.3	66.1	61.4
FedAdam	51.4	31.8	80.1	29.5	54.4	45.5
FedNova	58.7	46.6	87.9	36.3	63.7	60.0
FedFull	1.5	0.9	3.0	1.6	4.1	1.2
IST	66.6	**58.1**	89.0	39.9	**69.6**	55.6
ISTProx	**67.8**	56.6	**90.4**	**43.6**	69.5	58.9

(b) Communication (GB) to threshold acc.

	Birds	Cars	Flwrs	Aircrft	Textre	CFAR
FedAvg	FAIL	**194**	144	508	**96**	**142**
FedProx	FAIL	FAIL	157	FAIL	377	399
MOON	FAIL	FAIL	249	FAIL	319	214
FedAdam	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
FedNova	FAIL	FAIL	469	FAIL	843	1031
FedFull	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
IST	183	279	**129**	644	115	FAIL
ISProx	**150**	FAIL	148	**318**	143	FAIL

(c) GFLOPs to threshold acc.

	Birds	Cars	Flwrs	Aircrft	Textre	CFAR
FedAvg	FAIL	FAIL	7057	FAIL	6222	FAIL
FedProx	FAIL	FAIL	630	FAIL	1507	**1597**
MOON	FAIL	FAIL	1321	FAIL	1359	1815
FedAdam	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
FedNova	FAIL	FAIL	9392	FAIL	16877	20625
FedFull	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
IST	1400	FAIL	**367**	2610	754	FAIL
ISTProx	**602**	FAIL	594	**1274**	**572**	FAIL

Table 1: Skewed data results, 100 sites, 30% participation.

(a) Final accuracies

	Birds	Cars	Flwrs	Aircrft	Textre	CFAR
FedAvg	71.5	**56.4**	95.6	46.2	73.0	72.7
FedProx	70.2	50.1	95.4	45.7	71.9	71.7
MOON	70.3	48.0	95.1	45.7	71.7	71.5
FedAdam	47.2	28.2	82.5	22.7	48.2	42.6
FedNova	70.8	47.1	95.7	46.2	71.9	**74.1**
FedFull	1.0	0.7	1.7	1.3	3.3	1.3
IST	73.3	50.8	95.8	46.8	**75.0**	59.9
ISTProx	**73.7**	50.3	**96.1**	**48.1**	68.2	61.7

(b) Communication (GB) to threshold acc.

	Birds	Cars	Flwrs	Aircrft	Textre	CFAR
FedAvg	122	**74**	36	**162**	**65**	**92**
FedProx	141	FAIL	44	191	79	128
MOON	137	208	44	220	69	129
FedAdam	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
FedNova	377	478	113	465	146	234
FedFull	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
IST	92	FAIL	29	343	133	FAIL
ISTProx	**85**	FAIL	**26**	254	122	FAIL

(c) GFLOPs to threshold acc.

	Birds	Cars	Flwrs	Aircrft	Textre	CFAR
FedAvg	12243	**7410**	3586	16190	6523	9175
FedProx	563	FAIL	176	**766**	**317**	**513**
MOON	1239	FAIL	380	1275	546	917
FedAdam	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
FedNova	7534	9568	2263	9307	2924	4682
FedFull	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
IST	366	FAIL	115	1372	532	FAIL
ISTProx	**342**	FAIL	**104**	1018	487	FAIL

Table 2: Skewed data results, 1000 sites, 2% participation.

optimizing for one over the other, although in practice we find many cases where *one set of parameters work best for both*. We consider the number of local rounds of stochastic gradient descent as a hyperparameter with either 1, 5, or 25 iterations. Other hyperparameters are algorithm specific. Unless otherwise stated, we selected the parameters by table-scanning all combinations of possible hyperparameter configurations and optimizing on each of FLOP and GB transferred. We detail each method and the choices of hyperparameters below:

4.1. FedAvg

FedAvg does not have any other hyperparameter besides local training rounds. We find that FedAvg performs better on both communication and FLOP usage with **25** local iterations.

4.2. FedProx

FedProx has proximal hyperparameter $\mu \in [0.05, 0.1, 0.15, 0.2]$. For optimizing both FLOP and communication we find that $\mu = \mathbf{0.2}$ and **1** local round of training does best on both metrics.

4.3. FedFull

We reuse the choice of $\mu = \mathbf{0.2}$ from FedProx, but we find that 1 local iteration no longer works due to training the full model that a single batch is insufficient. We chose **5** local iterations for all cases as we found 25 too slow, but in either case the method generally times out due to a single communication round requiring too many FLOPs.

4.4. MOON

MOON has a weight hyperparameter for its contrastive term, similar to that of the proximal term used in FedProx,

(a) Final accuracies

	Birds	Cars	Flwrs	Aircrft	Textre	CFAR
FedAvg	71.6	55.6	95.6	46.2	71.9	74.0
FedProx	70.3	48.1	95.8	46.5	72.9	72.8
MOON	70.6	50.9	95.6	46.2	71.5	72.9
FedAdam	60.0	45.0	89.5	35.2	60.3	55.0
FedNova	71.4	46.6	95.2	46.2	72.5	**74.9**
FedFull	0.9	0.7	1.0	1.0	6.5	1.5
IST	**74.5**	**58.1**	96.3	49.2	75.7	64.8
ISTProx	74.1	56.6	**96.5**	**49.9**	**75.8**	65.8

(b) Communication (GB) to threshold acc.

	Birds	Cars	Flwrs	Aircrft	Textre	CFAR
FedAvg	369	**178**	91	693	146	190
FedProx	389	FAIL	103	569	158	**271**
MOON	381	183	111	641	157	272
FedAdam	FAIL	FAIL	213	FAIL	FAIL	FAIL
FedNova	982	1821	275	1735	405	729
FedFull	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
IST	130	411	31	670	137	FAIL
ISTProx	**110**	FAIL	**30**	**558**	**58**	FAIL

(c) GFLOPs to threshold acc.

	Birds	Cars	Flwrs	Aircrft	Textre	CFAR
FedAvg	11958	FAIL	4141	FAIL	4921	8811
FedProx	1557	FAIL	412	2278	633	**1083**
MOON	3152	FAIL	751	5752	1422	2336
FedAdam	FAIL	FAIL	5875	FAIL	FAIL	FAIL
FedNova	19646	**36432**	5509	34723	8112	14594
FedFull	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
IST	569	FAIL	136	2495	**231**	FAIL
ISTProx	**441**	FAIL	**120**	**2232**	233	FAIL

Table 3: Skewed data results, 1000 sites, 6% participation.

ISTProx, and FedNova. However this hyperparmater has range $\mu \in [1, 5, 10]$. The contrastive term also has a temperature parameter $\tau \in [0.1, 0.5, 1]$.

For FLOP optimized run, we choose with **1** local round, $\mu = \mathbf{1}, \tau = \mathbf{0.5}$

For communication optimized run, we choose with **5** local rounds, $\mu = \mathbf{10}, \tau = \mathbf{0.1}$

4.5. FedNova

FedNova also uses proximal hyperparameter $\mu \in [0.05, 0.1, 0.15, 0.2]$ and we reuse the choice $\mu = \mathbf{0.2}$ along with a choice of **5** local training rounds. Additionally, FedNova has a parameter τ_{eff}, the effective number of local training rounds. We use the recommended choice for τ_{eff} in [11], in the case of a nonzero proximal term, of dividing our local rounds by the number of participating clients.

4.6. FedAdam

In the Adam algorithm of FedAdam, we use the standard recommended settings [3, 9] of $\beta_1 = \mathbf{0.9}, \beta_2 = \mathbf{0.99}$ for the momentum and RMSProp weight decays. We chose the adaptability parameter $\tau = \mathbf{0.01}$ from a range of $\tau \in [10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}]$. We chose a local learning rate of **0.03** and global learning rate of **0.01** both from a range $[0.1, 0.03, 0.01, 0.003]$. For FLOP optimized FedAdam, we chose local rounds of **5**. For communication optimized, we chose local rounds of **25**

4.7. IST

IST has the local model size as a parameter. Here we tuned the central model size in the case of 10 client devices from [1000, 1500, 2000, 3000] and found that 3000 had the best final accuracy. Therefore we chose **300** as the local model size. Unlike other hyperparameters, we did not tune this to optimize for FLOPs or bytes. We found that **1** local round has best performance on both FLOPs and bytes transferred.

4.8. ISTProx

We reuse the **300** local model size in IST. Like FedProx, ISTProx has a proximal term $\mu \in [0.05, 0.1, 0.15, 0.2]$. We did not see much of an impact with the proximal term on IST as it does with FedProx, but $\mu = \mathbf{0.2}$ is still consistently the best choice on both metrics in our tuning along with training on **1** local round.

References

[1] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014. 1

[2] FAIR. facebookresearch/fvcore: Collection of common code that's shared among different research projects in fair computer vision team. https://github.com/facebookresearch/fvcore. Accessed: 2022-10-11. 2

[3] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 4

[4] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013. 1

[5] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009. 1

[6] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew B. Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *CoRR*, abs/1306.5151, 2013. 1

[7] Maria-Elena Nilsback and Andrew Zisserman. A visual vocabulary for flower classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1447–1454, 2006. 1

[8] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. Curran Associates Inc., Red Hook, NY, USA, 2019. 1

[9] Sashank J. Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and Hugh Brendan McMahan. Adaptive federated optimization. In *International Conference on Learning Representations*, 2021. 4

[10] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. Caltech-ucsd birds-200-2011 (cub-200-2011). Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 1

[11] Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. *Advances in neural information processing systems*, 33:7611–7623, 2020. 2, 4

[12] Tyler Yep. Tyleryep/torchinfo: View model summaries in pytorch! `https://github.com/TylerYep/torchinfo`. Accessed: 2022-10-11. 2