On Pre-Training for Federated Learning

Hong-You Chen, Cheng-Hao Tu, Ziwei Li, Han-Wei Shen, Wei-Lun Chao
Department of Computer Science and Engineering, The Ohio State University
{chen.9301, tu.343, li.5326, shen.94, chao.209}@osu.edu

Abstract

In most of the literature on federated learning (FL), neural networks are initialized with random weights. In this paper, we present an empirical study on the effect of pre-training on FL. Specifically, we aim to investigate if pre-training can alleviate the drastic accuracy drop when clients’ decentralized data are non-IID. We focus on FedAvg, the fundamental and most widely used FL algorithm. We found that pre-training does largely close the gap between FedAvg and centralized learning under non-IID data, but this does not come from alleviating the well-known model drifting problem in FedAvg’s local training. Instead, how pre-training helps FedAvg is by making FedAvg’s global aggregation more stable. When pre-training using real data is not feasible for FL, we propose a novel approach to pre-train with synthetic data. On various image datasets (including one for segmentation), our approach with synthetic pre-training leads to a notable gain, essentially a critical step toward scaling up federated learning for real-world applications.

1 Introduction

The increasing attention to data privacy and protection has attracted a flurry of research interests in federated learning (FL) [51, 82, 39]. In FL, data are kept separate at individual clients. The goal of FL is thus to learn a “global” model in a decentralized way. Specifically, one would hope to obtain a model whose accuracy is as good as if it were trained using centralized data.

FedAvg [60] is arguably the most widely used FL algorithm, which assumes that every client is connected to a server. FedAvg trains the global model in an iterative manner, between parallel local model training at the clients and global model aggregation at the server. FedAvg is easy to implement and enjoys theoretical guarantees of convergence [88, 72, 29, 54, 86]. Its performance, however, can degrade drastically when clients’ data are not IID — clients’ data are often collected individually and doomed to be non-IID. That is, the accuracy of the global model can be much lower than its counterpart trained with centralized data. To alleviate this issue, existing literature has explored better approaches for local training [52, 41, 1] or global aggregation [78, 37, 8].

In this paper, we investigate a different and rarely studied dimension in FL — initialization. Specifically, we investigate if model pre-training can alleviate the non-IID issue and fundamentally improve FedAvg. In centralized learning, pre-training on large-scale datasets [20, 35] has been shown to improve accuracy, generalizability, robustness, etc. In most of the FL literature, however, neural

Preprint. Under review.
networks are initialized with random weights. This setup makes sense from the FL perspective: FL is meant to learn a model without centralized data; thus, there should be no other data available for pre-training. With that being said, from an application perspective, pre-training has led to several breakthroughs in areas like computer vision and natural language processing; many pre-trained models are publicly available. It is hard to imagine that one in these areas would simply ignore the pre-trained models, even if the task at hand needs to be done in an FL setting. To incorporate these two perspectives, we conduct our study in two aspects and focus on visual recognition.

First, assuming pre-trained models are available, we conduct a systematic study to investigate whether and how they benefit FL. We use models pre-trained on datasets like ImageNet [19] to initialize FEDAVG for downstream federated tasks. As one might expect, pre-training significantly improves FEDAVG’s accuracy under non-IID conditions; the gap between FEDAVG and centralized training is largely reduced. To better understand this, we analyze the training dynamics of FEDAVG between local training and global aggregation. To our surprise, pre-training does not alleviate local model drifting [52] [41], a well-known issue under non-IID data. Instead, how pre-training helps FEDAVG is by making global aggregation more stable. Concretely, FEDAVG averages the local models’ weights simply by coefficients that are proportional to local data sizes. Due to the model drifting in local training, these coefficients can be far from optimal [8]. Interestingly, with pre-training, FEDAVG is less sensitive to the coefficients, resulting in a stronger global model in terms of accuracy.

Second, assuming pre-trained models are not available and there are no real data for pre-training at the server, we explore the use of synthetic data to make pre-training applicable to FL. We choose to use fractal images [43] [2] for two reasons. First, fractals are artificial-looking and have less privacy concern. Second, fractals have been shown to capture geometric properties of elements found in nature [58]. Directly applying the existing pre-training methods with fractals [43] [2], however, can hardly improve FL, as will be shown in subsection 6.2. To resolve this issue, we propose a novel approach to pre-train with fractals, taking the inner working of their generating process by iterated function systems [4] into account. The resulting pre-trained model can better capture the geometric variations of objects and is more useful for FL tasks of natural images.

We validate our approach on a variety of image classification datasets, including CIFAR-10/100 [48], Tiny-ImageNet [49], and iNaturalist [76]. We also work on semantic segmentation using Cityscapes [16]. Our empirical results demonstrate significant and consistent improvements on FEDAVG via pre-training, even using synthetic data. More importantly, the more challenging the non-IID condition is, the larger the gain by pre-training is. Our analyses further show the compatibility of pre-training with advanced FL algorithms [1] [52], and reveal new insights for improving FL in practice.

Contributions and scopes. We conduct the very first systematic study on pre-training for FL, using five image datasets including iNaturalist and Cityscapes. Our novelty for this part is in the findings. We believe that such a study is timely and significant to the FL community. When pre-training using real data is not feasible (there is still a debate in the FL community), we present a novel approach to use synthetic data for pre-training, which is sufficient to attain a notable gain. Our analyses further reveal several new insights of FL, opening up future research directions.

We focus on visual recognition, which has been the focus of many FL works. We go beyond them by further studying semantic segmentation problems, not merely classification problems.

2 Related Work

Federated learning. FEDAVG [60] is the fundamental FL algorithm. Many works were proposed to improve it, especially to alleviate its accuracy drop under non-IID data. In global aggregation, [78] [55] matched local model weights before averaging. [56] [50] [89] [8] replaced weight average by model ensemble and distillation. [57] [69] applied server momentum and adaptive optimization. In local training, to reduce local model drifting — a problem commonly believed to cause the accuracy drop — [56] used mixed client and server data; FEDPROX [52], FEDDANE [53], and FEDDYN [1] employed regularization toward the global model; SCAFFOLD [40] and MIME [41] used control varieties or server statistics to correct local gradients; [79] [84] modified local update rules. We investigate a rarely studied aspect to improve FEDAVG, initialization. To our knowledge, very few works have studied this [55] [73]; none are as systematic as ours on the effect of pre-training. [67] [38] [15] used pre-trained models in their experiments but did not or only briefly analyze their effect.
Pre-training. Pre-training has been widely applied in computer vision, natural language processing, and many other application domains to speed up convergence and boost accuracy for downstream tasks [46, 27, 68, 74, 20, 83, 6]. Many works have attempted to analyze its impacts [36, 23, 32, 21, 32, 47]. For example, [35] and [10] found that pre-training improves robustness against adversarial examples; [64] studied the loss landscape when fine-tuning on target tasks; [61] empirically showed that pre-training reduces forgetting in continual learning. Despite the ample research on pre-training, its impacts on FL remain largely unexplored. We aim to fill in this missing piece.

Training with synthetic data. Since it is costly to collect and annotate data, many works explored the idea of synthesizing real-looking images for training, e.g., by using 3D graphic renders [65, 66, 63, 62, 26] or deep generative models [5, 22, 59]. A few recent studies explored images generated by simple procedures; e.g., random generative models [5] or fractals [43, 2]. Surprisingly, pre-training on these non-realistic images produces good initialization for downstream tasks, outperforming training from scratch. We choose to use fractals as they are artificial-looking and have less privacy concern.

3 Background: Federated Learning

We provide a brief background of federated learning (FL). We use classification as the running example. The goal is to learn a classifier \( h_\theta = g_\theta \circ f_\phi \), where \( f_\phi \) is the feature extractor parameterized by \( \phi \) and \( g_\theta \) is the classification head parameterized by \( \theta \). We use \( \theta \) to denote \( \{\phi, \theta\} \).

In centralized learning, we are given a training set \( \mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N} \), where \( x \) is the input (e.g., images) and \( y \in \{1, \cdots, C\} = \mathbb{C} \) is the true label. We can learn \( \theta \) by empirical risk minimization

\[
\min_{\theta} \mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \ell(y_i, h_\theta(x_i)).
\]

(1)

Here, \( \mathcal{L} \) is the empirical risk, and \( \ell \) is the loss function.

In FL, the training data are collected at \( M \) clients. Each client \( m \in [M] \) has a training set \( \mathcal{D}_m = \{(x_i, y_i)\}_{i=1}^{|D_m|} \). Except for the difference in training data configurations, the goal of learning remains the same — to train a “global” model \( h_\theta \). The optimization problem in Equation 1 can be rewritten as

\[
\min_{\theta} \mathcal{L}(\theta) = \sum_{m=1}^{M} \frac{|D_m|}{|\mathcal{D}|} \mathcal{L}_m(\theta).
\]

(2)

Here, \( \mathcal{L}_m \) is the empirical risk of client \( m \); \( \mathcal{D} = \bigcup_m \mathcal{D}_m \) denotes the aggregated data from all clients.

Federated averaging (FEDAVG). As clients’ data are separated, Equation 2 cannot be solved directly. A standard way to relax it is FEDAVG [60], which iterates between two steps — local training at the clients and global aggregation at the server — for multiple rounds. Within each round, the server broadcasts the “global” model to the clients, who then independently update the model locally using their data. The server then aggregates the “local” models back into the “global” model and proceeds to the next round. In general, the two steps can be formulated as

Local: \( \theta^{(t)}_m = \arg\min_{\theta} \mathcal{L}_m(\theta) \), initialized by \( \bar{\theta}^{(t-1)} \), \quad Global: \( \bar{\theta}^{(t)} \leftarrow \sum_{m=1}^{M} \frac{|D_m|}{|\mathcal{D}|} \theta^{(t)}_m \).

(3)

The superscript \( t \) indicates the models after round \( t \); \( \bar{\theta}^{(0)} \) denotes the initial model. That is, local training aims to minimize each client’s empirical risk, often by several epochs of stochastic gradient descent (SGD). Global aggregation takes an element-wise average over local model parameters.

Problem. When clients’ data are non-IID, \( \theta^{(t)}_m \) would drift away from each other and from \( \bar{\theta}^{(t-1)} \), making \( \bar{\theta}^{(t)} \) deviate from the solution of Equation 2 and resulting in a drastic accuracy drop.

4 Pre-training for Federated Learning

In most of the FL literature that learns neural networks, the model \( \bar{\theta}^{(0)} \), or more specifically the feature extractor \( \bar{\phi}^{(0)} \), is initialized with random weights. In this section, we attempt to provide a detailed empirical study and analysis on the effect of pre-training on FL. Specifically, we investigate whether and how pre-training helps FEDAVG alleviate the accuracy drop in non-IID conditions.
4.1 Preparation and notation

To begin with, let us identify factors that may affect FEDAVG’s accuracy along its training process. Let us denote by $D_{test} = \{(x_i, y_i)\}_{i=1}^N$ the global test data, by $L_{test}(\theta)$ the test loss, and by $A_{test}(\theta)$ the test accuracy. Following Equation \ref{eq:decomposition}, we can decompose $A_{test}(\theta^{(t)})$ after round $t$ by

\[
A_{test}(\theta^{(t-1)}) + \sum_{m=1}^M \frac{|D_m|}{|D|} A_{test}(\theta_m^{(t)}) - A_{test}(\theta^{(t-1)}) = \Delta L^G + \Delta L^C.
\]

The first term is the initial test accuracy of round $t$; the second term ($\Delta L^G$) is the average gain by local training; the third ($\Delta L^C$) is the gain by global aggregation. A negative $\Delta L^G$ indicates that local models after local training (at round $t$) has somewhat “forgotten” what the global model $\theta^{(t-1)}$ has learned \cite{3,25}. Namely, the local models drift away from the global model.

4.2 Experimental setup

**Data.** We conduct the study using CIFAR-10, CIFAR-100 \cite{48}, Tiny-ImageNet \cite{39}, iNaturalist \cite{76}, and Cityscapes \cite{17}. The first four are for image classification; the last is for semantic segmentation. \table{1} summarizes their statistics. (We also include NLP tasks such as sentiment analysis \cite{77} and language modeling \cite{60} in the Suppl. and show similar improvements by pre-training as shown in Figure \ref{fig:contrast}.)

| Non-IID splits. | To simulate non-IID conditions, we follow \cite{37} to partition the training set of CIFAR-10, CIFAR-100, and Tiny-ImageNet into $M$ clients. To split the data of class $c$, we draw an $M$-dimensional vector $q_c$ from Dirichlet($\alpha$), and assign data of class $c$ to client $m$ proportionally to $q_c[m]$. The resulting clients have different numbers of total images and class distributions. The small the $\alpha$ is, the severer the non-IID condition is. For iNaturalist, a dataset for species recognition, we use the data proposed by \cite{38}, which are split by the Geo locations. There are two versions, GEO-10K (39 clients) and GEO-3K (136 clients). For Cityscapes, a dataset of street scenes, we use the official training/valid sets which contain 183 cities, respectively. We split the training data by cities to simulate a realistic scenario. |
| Metrics. | We report the averaged accuracy (%) for classification and the mIoU (%) for segmentation. |
| Models. | We use ResNet20 \cite{33} for CIFAR-10/100, which is suitable for a $32 \times 32$ resolution. We use ResNet18 \cite{33} for Tiny-ImageNet and iNaturalist. For semantic segmentation on Cityscapes, we use DeepLabV3+ \cite{9} with a MobileNet-V2 \cite{71} backbone. |
| Pre-training. | We compare initializing these models with random or pre-trained weights. For all the datasets except Tiny-ImageNet, we use ImageNet \cite{19} for pre-training. For Tiny-ImageNet (a subset of ImageNet), we use Places365 \cite{87}. The pre-training is done by standard supervised training. Specifically, we obtain pre-trained checkpoints from PyTorch’s and Places365’s official sites \cite{4}.

**Training.** We perform FEDAVG for 100 iterative rounds. Each round of local training takes 5 epochs. We use the SGD optimizer with weight decay $1e^{-4}$ and a 0.9 momentum, except that on DeepLabV3+ we use an Adam \cite{44} optimizer. We apply the standard image pre-processing and data augmentation \cite{33}. We reserve 2% data of the training set as the validation set for hyperparameter tuning (e.g., for the learning rate). We follow the literature \cite{34} to decay the learning rate by 0.1 every 30 rounds. We leave more details and the selected hyperparameters in section \ref{sec:sensitivity} and the Suppl.

**Scopes of this section.** Due to the space limit, we mainly study the following settings and postpone CIFAR-100 to section \ref{sec:sensitivity}. For CIFAR-10 and Tiny-ImageNet, we use $\alpha = 0.3$ and set $M = 10$. For iNaturalist, we use GEO-3K ($M = 136$). All clients participate in every round, except that on GEO-3K we sample 20% per round to simulate partial participation. We include other settings, e.g., larger $M$ and different $\alpha$, client sampling, and learning rate scheduling in section \ref{sec:sensitivity} and the Suppl.

\footnote{For ResNet20, we pre-trained with the same procedure but on down-sampled $32 \times 32$ ImageNet images. \footnote{We found this this number quite stable for all experiments and freeze it throughout the paper.} \footnote{We use the same learning rate within each dataset, which is selected by FEDAVG without pre-training.}}

\begin{table}[h]
\centering
\caption{Summary of datasets and setups.}
\begin{tabular}{|c|c|c|c|c|c|c|}
\hline
Dataset & \#Class & \#Training & \#Test & \#Clients & $M$ & Resolution \\
\hline
CIFAR-10/100 & 10/100 & 50K & 10K & 10 & 32$^2$ & \\
Tiny-ImageNet & 200 & 100K & 10K & 100/100/1000 & 64$^2$ & \\
iNaturalist-GEO & 1203 & 120K & 36K & 39/136 & 224$^2$ & \\
Cityscapes & 19 & 3K & 0.5K & 18 & 768$^2$ & \\
\hline
\end{tabular}
\end{table}
According to [41, 8], this simple weight average may gradually deviate from the solution of Equation 1 we found notable local model drifting (i.e., negative $\Delta$).

To verify our hypothesis, we conduct a further analysis on global aggregation. In Figure 2, we compare the test accuracy/mIoU using the global model $\theta^{(t)}$ ($\times$), and the averaged test accuracy/mIoU using each local model $\theta^{(t)}_m$ ($\bullet$). The red and green arrows indicate the gain by local training ($\Delta^{(t)}_L$)/global aggregation ($\Delta^{(t)}_G$). For brevity, we only draw the first 50 rounds but the final accuracy/mIoU are after 100 rounds.

### 4.3 Results, observations, and discussions

Figure 2 summarizes the results. For each combination (dataset + pre-training or not), we show the test accuracy/mIoU using global model, i.e., $A_{test}(\theta^{(t)})$ ($\times$). We also show the averaged test accuracy/mIoU using each local model, i.e., $\sum_{m=1}^{M} \frac{1}{|D_m|} A_{test}(\theta^{(t)}_m)$ ($\bullet$). The red and green arrows indicate the gain by local training ($\Delta^{(t)}_L$)/global aggregation ($\Delta^{(t)}_G$). For brevity, we only draw the first 50 rounds but the final accuracy/mIoU are after 100 rounds. For each combination, we also depict the accuracy of centralized learning. We have the following observations and one hypotheses.

**Pre-training significantly improves FEDAVG.** On CIFAR-10, FEDAVG with pre-training has a 8.1% gain over FEDAVG without pre-training (90.8% vs. 82.7%). On Tiny-ImageNet, the gain is 7.9% (50.3% vs. 42.4%). Similar findings can be seen on iNaturalist and Cityscapes. Moreover, we found that pre-training brings more benefits to federated learning than to centralized learning. For the latter, the gains on CIFAR-10 and Tiny-ImageNet are 1.6% and 5.4%. In other words, while it is expected that pre-training would improve neural network training as it encodes the knowledge of external data, pre-training seems to offer extra benefits to FEDAVG.

**Pre-training does not alleviate local model drifting.** Local model drifting is commonly believed as the main cause of accuracy drop in FL [8, 52, 41]. For instance, on FEDAVG without pre-training, we found notable local model drifting (i.e., negative $\Delta^{(t)}_L$). This can be seen from the slanting line segments: segments with negative slopes (i.e., $\bullet$ of round $t$ is lower than $\times$ of round $t-1$) indicate drifting. Interestingly, while pre-training improves FEDAVG’s test accuracy, we see no clear decrease but sometimes increase in local model drifting. We thus postulate that reducing local model drifting may not be necessary for achieving a high test accuracy in FL.

**Global aggregation compensates for local model drifting.** The main reason that reducing local model drifting may not be necessary is the gain $\Delta^{(t)}_G$ produced by global aggregation. As in Figure 2, global aggregation (vertical segments from $\bullet$ to $\times$) almost always leads to improvements. Interestingly, the larger the drift is (i.e., negative $\Delta^{(t)}_L$), the larger the aggregation gain is (i.e., positive $\Delta^{(t)}_G$). With that being said, we see some exceptions in the first few rounds of FEDAVG: global aggregation results in a negative $\Delta^{(t)}_G$, especially for FEDAVG without pre-training. While these cases are scarce, they seem to suggest a hypothesis: pre-training leads to more robust global aggregation.

### 4.4 Analysis on global aggregation

To verify our hypothesis, we conduct a further analysis on global aggregation. In FEDAVG, global aggregation is by a simple weight average, using local data sizes as the coefficients. According to [41, 8], this simple weight average may gradually deviate from the solution of Equation 1 under non-IID conditions, and ultimately lead to a drastic accuracy drop. Here, we want to analyze if pre-training can alleviate this issue. Of course, for over-parameterized models like neural networks, it is unlikely to find a unique minimum of Equation 1 to calculate the deviation. We thus propose an alternative way to quantify the robustness of aggregation. We focus on CIFAR-10, Tiny-ImageNet.

![Figure 2](image-url)
When pre-trained models are not available, or there are no centralized real data for pre-training, we resort to the use of synthetic data. Specifically, we build upon the recent works by [43, 2].

In this work, we focus on the effect of pre-training on robust aggregation. We verify that pre-training leads to more robust global aggregation for FEDAVG.

Fractal generation. A fractal image can be rendered via an affine Iterative Function System (IFS) [43, 4]. The IFS rendering process can be thought of as “drawing” pixels iteratively on a canvas. The pixel transition is governed by a set of $2 \times 2$ affine transformations, which we call an IFS code $S$:

$$S = \left\{ \left( A_k \in \mathbb{R}^{2 \times 2}, b_k \in \mathbb{R}^2, p_k \in \mathbb{R} \right) \right\}_{k=1}^K.$$  

5.1 Background: supervised fractal pre-training

Our idea is inspired by [8], which showed that the coefficients used by FEDAVG may not optimally combine the local models. This motivates us to search for the optimal convex aggregation and calculate its accuracy gap against the simple average. The smaller the gap is, the more robust the global aggregation is. We define the optimal convex aggregation as follows:

$$\bar{\theta}^*(t) = \sum_{m=1}^M \lambda_m^* \hat{\theta}^{(t)}_m,$$

where $\{\lambda_m^*\} = \arg\max_{\lambda_m \geq 0, \sum_m \lambda_m = 1} A_{\text{test}}(\sum_{m=1}^M \lambda_m \bar{\theta}^{(t)}_m)$  

That is, we search for the vector $\lambda = [\lambda_1, \cdots, \lambda_M]^\top$ in the $(M-1)$-simplex that maximizes $A_{\text{test}}$. Here, we apply SGD to find $\{\lambda_m^*\}$. (See the Suppl. for details.)

Figure 3 shows the curves of $A_{\text{test}}(\theta^{(t)})$ and $A_{\text{test}}(\bar{\theta}^{(t)})$. For $A_{\text{test}}(\theta^{(t)})$, we replace the simple weight average by the optimal convex aggregation throughout the entire FEDAVG. At the beginning of each round, we send the optimal convex aggregation back to clients for their local training. It can be seen that $A_{\text{test}}(\bar{\theta}^{(t)})$ outperforms $A_{\text{test}}(\theta^{(t)})$. The gap is larger for FEDAVG without pre-training than with pre-training. In other words, for FEDAVG with pre-training, the simple weight average has a much closer accuracy to the optimal convex aggregation. This verifies that pre-training leads to more robust global aggregation for FEDAVG.

Robust aggregation seems to come from a lower-variance loss surface. Why does pre-training lead to robust aggregation? To answer it, we investigate the variation of $L_{\text{test}}(\sum_{m=1}^M \lambda_m \hat{\theta}^{(t)}_m)$ for different $\lambda$ on the simplex. Concretely, we uniformly sample $\lambda$ for 300 times, construct the global models and calculate their test losses, and compute the 95% confidence interval. As shown in Figure 4, FEDAVG with pre-training has a smaller interval; i.e., a lower-variance loss surface in aggregation. This helps explain why it has a smaller gap between $A_{\text{test}}(\bar{\theta}^{(t)})$ and $A_{\text{test}}(\theta^{(t)})$.

5 Pre-training with Synthetic Data

When pre-trained models are not available, or there are no centralized real data for pre-training, we resort to the use of synthetic data. Specifically, we build upon the recent works by [43, 2] and propose a novel and more effective way to leverage fractal images for pre-training. We choose fractals since they are artificial-looking and have less privacy concern, making them suitable for federated applications. In the following, we give a concise background, followed by our approach.

5.1 Background: supervised fractal pre-training

Fractal generation. A fractal image can be rendered via an affine Iterative Function System (IFS) [43, 3]. The IFS rendering process can be thought of as “drawing” pixels iteratively on a canvas. The pixel transition is governed by a set of $2 \times 2$ affine transformations, which we call an IFS code $S$:

$$S = \left\{ \left( A_k \in \mathbb{R}^{2 \times 2}, b_k \in \mathbb{R}^2, p_k \in \mathbb{R} \right) \right\}_{k=1}^K.$$  

Figure 4: Pre-training leads to a lower-variance loss surface. We show the test loss by the FEDAVG’s global model $\theta^{(t)}$ (curve) and the 95% confidence interval of the test losses by the sampled convex combinations (shaded area) on CIFAR-10 and Tiny-ImageNet. We also show the histograms of losses at round 10 and 50.

Optimal convex aggregation. Our idea is inspired by [8], which showed that the coefficients used by FEDAVG may not optimally combine the local models. This motivates us to search for the optimal convex aggregation and calculate its accuracy gap against the simple average. The smaller the gap is, the more robust the global aggregation is. We define the optimal convex aggregation as follows:

That is, we search for the vector $\lambda = [\lambda_1, \cdots, \lambda_M]^\top$ in the $(M-1)$-simplex that maximizes $A_{\text{test}}$. Here, we apply SGD to find $\{\lambda_m^*\}$. (See the Suppl. for details.)

Figure 3: Pre-training leads to robust aggregation. We show for each combination (dataset-pre-training or not) the test accuracy using the global model $\bar{\theta}^{(t)}$ (solid). We also show $\theta^{(t)}$ (dashed), which applies the optimal convex aggregation throughout the entire FEDAVG process. FEDAVG with pre-training has a smaller gap.
Here, an \((A_k, b_k)\) pair specifies an affine transformation and \(p_k\) is the corresponding probability. Concretely, given a set \(S\), an IFS generates an image as follows. Starting from an initial pixel coordinate \(v_0 \in \mathbb{R}^2\), it repeatedly samples one transformation \(k\) with replacement according to the probability \(p_k\), and performs \(v_{n+1} = A_k v_n + b_k\) to arrive at the next pixel coordinate. This process continues until a sufficient number of iterations is reached. The traveled coordinates \(\{v_0, \cdots, v_n, \cdots\}\) are then used to synthesize a fractal: by rendering each pixel as a binary or continuous value on a black canvas. Due to the randomness in the IFS, one code can create different but geometrically-similar fractals.

**Supervised pre-training.** Since the IFS code controls the generation process and hence the geometry of the fractal, it can essentially be seen as the fractal’s ID or class label. In \([43]\), the authors proposed to sample \(J\) different codes \(\{S_j\}_{j=1}^J\) and create for each code a number of images to construct a \(J\)-class classification dataset. This synthetic labeled dataset is then used to pre-train a neural network via a multi-class loss (e.g., cross entropy). The following-up work by \([2]\) proposed to create more complex images by painting multiple (denoted by 1) fractals on one canvas. The resulting image thus has \(I\) out of \(J\) labels, on which a multi-label loss is more suitable for supervised pre-training.

### 5.2 Our approach: Fractal Pair Similarity (FPS)

We propose a novel way for fractal pre-training, inspired by one critical insight — *we can in theory sample infinitely many IFS codes and create fractals with highly diverse geometric properties.* That is, instead of creating a \(J\)-class dataset that limits the diversity of codes (e.g., \(J = 1K\)) but focuses more on intra-class variation, we propose to trade the latter for the former.

We propose to sample a new set of codes and create an image that contains multiple fractals on the fly. Namely, for each image, we sample a small set of \(I\) codes \(\{S_i\}_{i=1}^I\), generate \(I\) corresponding fractals, and composite them into one image. The resulting dataset will have all its images from different classes (i.e., different sets of \(\{S_i\}_{i=1}^I\)) in theory.

**Analogy to self-supervised learning.** How can we pre-train a neural network using a dataset whose images are all of different class labels? Here, we propose to view the dataset as essentially “unlabeled”, and draw an analogy to self-supervised learning \([57]\), especially those based on contrastive learning \([31, 31, 13, 12, 28]\) or similarity learning \([28, 14]\). The core idea of these approaches is to treat every image as from a different class and learn to either repulse different images away (i.e., negative pairs) or draw an image and an augmented version of it closer (i.e., positive pairs). This line of approaches is so effective that they can even outperform supervised pre-training in many tasks \([24]\).

We note that while \([27]\) has applied self-supervised learning to fractals, the motivation is very different from ours: the authors directly used the supervised dataset created by \([43]\) but ignored the labels.

**Fractal Pair Similarity (FPS).** Conventionally, to employ contrastive learning or similarity learning, one must perform data augmentation such that a single image becomes a positive pair. Common methods are image-level scale/color jittering, flips, crops, RandAugment \([13]\), SelfAugment \([70]\), etc. Here, we propose to exploit one unique way for fractals (or broadly, synthetic data), which is to use the IFS to create a pair of images based on the same set of codes \(\{S_i\}_{i=1}^I\). We argue that this method can create stronger and more physically-meaningful argumentation: not only do different fractals from the same codes capture intra-class variation, but we also can create a pair of images with different placements of the same \(I\) fractals (more like object-level augmentation). Moreover, this method can easily be compatible with commonly used augmentation.

**Implementation.** We apply a similarity learning approach SimSiam \([14]\) for pre-training, due to its efficiency and effectiveness. See Figure 5 for an illustration and see the Suppl. for details. Other self-supervised learning algorithms are easily applicable. We use the IFS code sampling scheme proposed in \([2]\) (with \(K \in \{2, 3, 4\}\)) and similarly apply scale jittering on each IFS code before we use it to render fractals. We also follow \([2]\) to color, resize, and flip the fractals. We pre-sample a total of 100K IFS codes in advance and uniformly sample \(I\) codes from them to generate a pair of images.
6 Experiments

We conduct extensive experiments to validate FPS and justify the importance of pre-training for FL.

6.1 Setup

This section follows the datasets and setup described in subsection 4.2, unless stated otherwise.

Fractal pre-training. For ResNet20, we render $32 \times 32$ images with $I = 2$ and run the IFS for 1K iterations. For ResNet18 and DeepLabV3+, we render $224 \times 224$ images by sampling $I$ uniformly from $\{2, 3, 4, 5\}$ per image to increase the diversity, and run the IFS for 100K iterations. For multi-label pre-training, we choose $J = 1K$ IFS codes as it gives the best validation result. All methods are pre-trained for 100 epochs. Each epoch has 1M image pairs or 2M images for fair comparisons.

Notation. Given a model $\theta_{FL}$ learned in FL, we denote the corresponding centrally learned model on the same dataset by $\theta_{CL}$. We use $\Delta_{CL-FL} = A_{test}(\theta_{CL}) - A_{test}(\theta_{FL})$ to denote their accuracy gap. We highlight the gain by pre-training, i.e., pre-training vs. random initialization, by magenta digits.

6.2 Comparisons on synthetic pre-training

Table 2: Comparison on synthetic pre-training methods in section 5. Means of 3 runs are reported.

| Init. / Dataset | C10  | C100 | Tiny-ImgNet |
|-----------------|------|------|-------------|
| Random          | 74.4 | 51.4 | 42.4        |
| Fractal + Multi-label | 73.0 | 51.0 | 40.9        |
| StyleGAN + SimSiam | 79.2 | 53.0 | 44.6        |
| Fractal + SimSiam | 77.4 | 51.7 | 44.2        |
| FPS (ours) + SimSiam | 80.5 | 54.7 | 45.7        |
| FPS (ours) + MoCo-V2 | 79.9 | 53.9 | 46.1        |
| ImageNet / Places365 | 87.9 | 62.2 | 50.3        |

We first compare synthetic pre-training methods. We also consider the images generated by (the best version of) random StyleGAN [3, 42]. We use these methods to initialize FEDAVG for non-IID CIFAR-10/100 of Dirichlet(0.1) and Tiny-ImageNet of Dirichlet(0.3). We split the training data into $M = 10$ clients with full participation every round. Results are in Table 2. Our approach FPS outperforms all the baselines except the real pre-training. By taking a deeper look, we found that multi-label supervised training with fractals cannot learn strong enough features for downstream tasks; the accuracy is worse than random initialization, essentially a case of negative transfer [80]. We attribute this to both the limited diversity of fractals and the learning mechanism.

Self-supervised learning, on the contrary, can better learn from diverse fractals (i.e., each fractal a class), although the gain is not pronounced.

By taking the inner working of fractals into account to create geometrically-meaningful positive pairs, our FPS unleashes the power of synthetic pre-training.

We apply another self-supervised learning algorithm MoCo-V2 [13] to show the generalizability of FPS. The accuracies are similar, but SimSiam is faster to train. Later, we focus on FPS (+SimSiam).

6.3 iNaturalist and Cityscapes

Table 3: Top-1 test accuracy on iNaturalist-GEO [38]

| Init. / Setting | GEO-10K | GEO-3K | Centralized | Avg. $\Delta_{CL-FL}$ |
|-----------------|---------|--------|-------------|----------------------|
| Random          | 20.9    | 12.2   | 48.5        | 32.0                 |
| FPS             | 27.8 (+6.9) | 17.7 (+5.5) | 50.2 (+1.7) | 27.5                 |
| ImageNet        | 46.6 (+25.7) | 45.6 (+33.4) | 58.4 (+9.9) | 12.3                 |

Table 4: mIoU (%) of segmentation on Cityscapes.

| Init. / Setting | Cities | Centralized | $\Delta_{CL-FL}$ |
|-----------------|--------|-------------|------------------|
| Random          | 41.0   | 57.2        | 16.2             |
| FPS             | 64.9 (+23.9) | 67.2 (+10.0) | 2.3              |
| ImageNet        | 67.6 (+26.6) | 71.1 (+13.9) | 3.5              |

Table 3 summarizes the results on iNaturalist 2017 [77], using the GEO-10K (39 clients and 50% clients per round) and GEO-3K (136 clients and 20% clients per round) splits according to Geo locations proposed by [38].

Table 4 summarizes the semantic segmentation results (mIoU) on Cityscapes [16]. The training data are split into 18 clients by cities, and we consider full client participation every round.

From both tables, we see clear gaps between centralized and federated learning (i.e., $\Delta_{CL-FL}$) on realistic non-IID splits, and pre-training with either synthetic (i.e., FPS) or real data notably reduces the gaps. Specifically, compared to random initialization, FPS brings an encouraging gains in FL (> 5% on iNaturalist; > 20% on Cityscapes). The gain is larger than that on centralized learning.

4We tried to increase $J$ to have more diverse fractals, but cannot improve due to poor convergence.
6.4 Different federated settings

We study different federated settings, including different numbers of clients $M$, non-IID degrees $\alpha$, and client participation. We also consider different numbers of local epochs per round but keep the total local epochs as 500. Figure 6 summarizes the results, in which we use Tiny-ImageNet with a base setting ($M = 10$ clients, Dir($\alpha = 0.3$), 100% participation, and local epoch = 5) and change one variable at a time. We see consistent gains by either real and synthetic pre-training across all the settings. Importantly, when the settings become more challenging (e.g., smaller $\alpha$ or more clients), the gain by pre-training gets larger, justifying its practical applicability and value.

6.5 Compatibility to other FL methods

We study if pre-training could improve more advanced FL methods like FedProx [52], FedDyn [1], and MOON [50]. Figure 7 shows the results. FPS is compatible with these methods to boost their accuracy. Interestingly, when pre-training using real data is considered, all these methods perform quite similarly. This suggests that FedAvg is still a quite strong FL approach if pre-trained models are available.

6.6 Further Analysis

Pre-training makes network sizes scale better. On CIFAR-10 Dir(0.3), we study different network depths and widths based on ResNet20. As in Figure 8 with random initialization, deeper models have quite limited gains; wider models improve more. With FPS, both going deeper or wider has notable gains, suggesting that pre-training makes training larger models easier in FL.

Federated pre-training on ImageNet. So far we discuss pre-training in a single site and then apply the pre-trained model to FL. We perform a preliminary trial to show that federated pre-training is also promising. We split ImageNet into 100 clients with Dir(0.3) and train a ResNet18 by FedAvg. It reaches 53.5% top-1 validation accuracy at 100 rounds (10% clients every round). We use this checkpoint to initialize the iNaturalist experiment and achieve 40.5% and 37.4% on GEO-10K and GEO-3K (cf. Table 3), respectively. The results are encouraging: they are close to using centralized ImageNet pre-training and much higher than training from scratch.

7 Conclusion

We investigate a rarely studied dimension in federated learning (FL): initialization. We conduct the very first systematic study on pre-training for FL, and show that it largely bridges the gap between FL and centralized learning. When model pre-training using real data is not feasible, we present a novel approach to use synthetic data for pre-training, which is sufficient to attain a notable gain. We validate our approach on five image datasets, including iNaturalist and Cityscapes. Our analyses further reveal several new insights of FL, opening up future research directions.
Acknowledgments

This research is supported in part by grants from the National Science Foundation (IIS-2107077, OAC-2118240, and OAC-2112606), the OSU GI Development funds, and Cisco Systems, Inc. We are thankful for the generous support of the computational resources by the Ohio Supercomputer Center and AWS Cloud Credits for Research.

References

[1] Durmus Alp Emre Acar, Yue Zhao, Ramon Matas, Matthew Mattina, Paul Whatmough, and Venkatesh Saligrama. Federated learning based on dynamic regularization. In ICLR, 2021.

[2] Connor Anderson and Ryan Farrell. Improving fractal pre-training. In WACV, pages 1300–1309, 2022.

[3] Manel Baradad, Jonas Wulf, Tongzhou Wang, Phillip Isola, and Antonio Torralba. Learning to see by looking at noise. In NeurIPS, 2021.

[4] Michael F Barnsley. Fractals everywhere. Academic press, 2014.

[5] Christopher Bowles, Liang Chen, Ricardo Guerrero, Paul Bentley, Roger Gunn, Alexander Hammers, David Alexander Dickie, Maria Valdés Hernández, Joanna Wardlaw, and Daniel Rueckert. Gan augmentation: Augmenting training data using generative adversarial networks. arXiv preprint arXiv:1810.10863, 2018.

[6] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In NeurIPS, volume 33, pages 1877–1901, 2020.

[7] Sebastian Caldas, Peter Wu, Tian Li, Jakub Konečný, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings. arXiv preprint arXiv:1812.01097, 2018.

[8] Hong-You Chen and Wei-Lun Chao. Fedbe: Making bayesian model ensemble applicable to federated learning. In ICLR, 2021.

[9] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In Proceedings of the European conference on computer vision (ECCV), pages 801–818, 2018.

[10] Tianlong Chen, Sijia Liu, Shiyu Chang, Yu Cheng, Lisa Amini, and Zhangyang Wang. Adversarial robustness: From self-supervised pre-training to fine-tuning. In CVPR, pages 699–708, 2020.

[11] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In ICML, pages 1597–1607, PMLR, 2020.

[12] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big self-supervised models are strong semi-supervised learners. In NeurIPS, 2020.

[13] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. arXiv preprint arXiv:2003.04297, 2020.

[14] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In CVPR, pages 15750–15758, 2021.

[15] Gary Cheng, Karan Chadha, and John Duchi. Fine-tuning is fine in federated learning. arXiv preprint arXiv:2108.07313, 2021.
[16] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In CVPR, pages 3213–3223, 2016.

[17] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Scharwächter, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset. In CVPR Workshop on The Future of Datasets in Vision, 2015.

[18] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In CVPR, pages 702–703, 2020.

[19] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In CVPR, 2009.

[20] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.

[21] Josip Djolonga, Jessica Yung, Michael Tschannen, Rob Romijnders, Lucas Beyer, Alexander Kolesnikov, Joan Puigcerver, Matthias Minderer, Alexander D’Amour, Dan Moldovan, et al. On robustness and transferability of convolutional neural networks. In CVPR, pages 16458–16468, 2021.

[22] Gabriel Eilertsen, Apostolia Tsirikoglou, Claes Lundström, and Jonas Unger. Ensembles of gans for synthetic training data generation. arXiv preprint arXiv:2104.11797, 2021.

[23] Dumitru Erhan, Aaron Courville, Yoshua Bengio, and Pascal Vincent. Why does unsupervised pre-training help deep learning? In Proceedings of the thirteenth international conference on artificial intelligence and statistics, pages 201–208. JMLR Workshop and Conference Proceedings, 2010.

[24] Linus Ericsson, Henry Gouk, and Timothy M Hospedales. How well do self-supervised models transfer? In CVPR, pages 5414–5423, 2021.

[25] Robert M French. Catastrophic forgetting in connectionist networks. Trends in cognitive sciences, 3(4):128–135, 1999.

[26] Chuang Gan, Jeremy Schwartz, Seth Alter, Martin Schrimpf, James Traer, Julian De Freitas, Jonas Kubilius, Abhishek Bhandwaldar, Nick Haber, Megumi Sano, et al. Threedworld: A platform for interactive multi-modal physical simulation. arXiv preprint arXiv:2007.04954, 2020.

[27] Priya Goyal, Mathilde Caron, Benjamin Lefauveaux, Min Xu, Pengchao Wang, Vivek Pai, Mannat Singh, Vitaliy Liptchinsky, Ishan Misra, Armand Joulin, et al. Self-supervised pretraining of visual features in the wild. arXiv preprint arXiv:2103.01988, 2021.

[28] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. In NeurIPS, 2020.

[29] Farzin Haddadpour and Mehrdad Mahdavi. On the convergence of local descent methods in federated learning. arXiv preprint arXiv:1910.14425, 2019.

[30] Chaoyang He, Murali Annavaram, and Salman Avestimehr. Group knowledge transfer: Federated learning of large cnns at the edge. In NeurIPS, 2020.

[31] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In CVPR, pages 9729–9738, 2020.

[32] Kaiming He, Ross Girshick, and Piotr Dollár. Rethinking imagenet pre-training. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 4918–4927, 2019.

[33] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In CVPR, 2016.
[34] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

[35] Dan Hendrycks, Kimin Lee, and Mantas Mazeika. Using pre-training can improve model robustness and uncertainty. In *ICML*, pages 2712–2721. PMLR, 2019.

[36] Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. Pretrained transformers improve out-of-distribution robustness. *arXiv preprint arXiv:2004.06100*, 2020.

[37] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

[38] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Federated visual classification with real-world data distribution. In *ECCV*, 2020.

[39] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

[40] Sai Praneeth Karimireddy, Martín Jaggi, Satyen Kale, Mehryar Mohri, Sashank J Reddi, Sebastian U Stich, and Ananda Theertha Suresh. Mime: Mimicking centralized stochastic algorithms in federated learning. *arXiv preprint arXiv:2008.03606*, 2020.

[41] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *ICML*, 2020.

[42] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[43] Hirokatsu Kataoka, Kazushige Okayasu, Asato Matsumoto, Ei suke Yamagata, Ryosuke Yamada, Nakamasa Inoue, Akio Nakamura, and Yutaka Satoh. Pre-training without natural images. In *ACCV*, 2020.

[44] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[45] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[46] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *ECCV*, pages 491–507. Springer, 2020.

[47] Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? In *CVPR*, pages 2661–2671, 2019.

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

[49] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 2015.

[50] Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10713–10722, 2021.

[51] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.
[52] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In MLSys, 2020.

[53] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smithy. Feddane: A federated newton-type method. In 2019 53rd Asilomar Conference on Signals, Systems, and Computers, 2019.

[54] Xiang Li, Kaixuan Huang, Wenhai Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. In ICLR, 2020.

[55] Bill Yuchen Lin, Chaoyang He, Zihang Zeng, Hulin Wang, Yufen Huang, Mahdi Soltanolkotabi, Xiang Ren, and Salman Avetisianr. Fednlp: Benchmarking federated learning methods for natural language processing tasks. In NAACL, 2022.

[56] Tao Lin, Lingjing Kong, Sebastian U Stich, and Martin Jaggi. Ensemble distillation for robust model fusion in federated learning. In NeurIPS, 2020.

[57] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Li Mian, Zhaoyu Wang, Jing Zhang, and Jie Tang. Self-supervised learning: Generative or contrastive. IEEE Transactions on Knowledge and Data Engineering, 2021.

[58] Benoit B Mandelbrot and Benoit B Mandelbrot. The fractal geometry of nature, volume 1. WH freeman New York, 1982.

[59] Nikolaus Mayer, Eddy Ilg, Philipp Fischer, Caner Hazirbas, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. What makes good synthetic training data for learning disparity and optical flow estimation? IJCV, 126(9):942–960, 2018.

[60] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. Communication-efficient learning of deep networks from decentralized data. In AISTATS, 2017.

[61] Sanket Vaibhav Mehta, Darshan Patil, Sarath Chandar, and Emma Strubell. An empirical investigation of the role of pre-training in lifelong learning. arXiv preprint arXiv:2112.09153, 2021.

[62] Hiroaki Mikami, Kenji Fukumizu, Shogo Murai, Shuji Suzuki, Yuta Kikuchi, Taiji Suzuki, Shin-ichi Maeda, and Kohei Hayashi. A scaling law for synthetic-to-real transfer: How much is your pre-training effective? arXiv preprint arXiv:2108.11018, 2021.

[63] Samarth Mishra, Rameswar Panda, Cheng Perng Phoo, Chun-Fu Chen, Leonid Karlinsky, Kate Saenko, Venkatesh Saligrama, and Rogerio S Feris. Task2sim: Towards effective pre-training and transfer from synthetic data. arXiv preprint arXiv:2112.00054, 2021.

[64] Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. What is being transferred in transfer learning? In NeurIPS, 2020.

[65] Xingchao Peng, Baochen Sun, Karim Ali, and Kate Saenko. Learning deep object detectors from 3d models. In Proceedings of the IEEE international conference on computer vision, pages 1278–1286, 2015.

[66] Aayush Prakash, Shaad Boochoon, Mark Brophy, David Acuna, Eric Cameracci, Gabriël State, Omer Shapira, and Stan Birchfield. Structured domain randomization: Bridging the reality gap by context-aware synthetic data. In ICRA, pages 7249–7255. IEEE, 2019.

[67] Liangqiong Qu, Yuyin Zhou, Paul Pu Liang, Yingda Xia, Feifei Wang, Li Fei-Fei, Ehsan Adeli, and Daniel Rubin. Rethinking architecture design for tackling data heterogeneity in federated learning. arXiv preprint arXiv:2106.06047, 2021.

[68] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In ICML, 2021.

[69] Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and H Brendan McMahan. Adaptive federated optimization. In ICLR, 2021.
[70] Colorado J Reed, Sean Metzger, Aravind Srinivas, Trevor Darrell, and Kurt Keutzer. Self-augment: Automatic augmentation policies for self-supervised learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 2674–2683, 2021.

[71] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4510–4520, 2018.

[72] Sebastian U Stich. Local sgd converges fast and communicates little. In ICLR, 2019.

[73] Joel Stremmel and Arjun Singh. Pretraining federated text models for next word prediction. In Future of Information and Communication Conference, pages 477–488, 2021.

[74] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In ICCV, pages 843–852, 2017.

[75] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning? In NeurIPS, 2020.

[76] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The naturalist species classification and detection dataset. In CVPR, 2018.

[77] Grant Van Horn and Pietro Perona. The devil is in the tails: Fine-grained classification in the wild. arXiv preprint arXiv:1709.01450, 2017.

[78] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. In ICLR, 2020.

[79] Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. In NeurIPS, 2020.

[80] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In CVPR, pages 3733–3742, 2018.

[81] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. ACM Transactions on Intelligent Systems and Technology (TIST), 10(2):1–19, 2019.

[82] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. NeurIPS, 32, 2019.

[83] Xin Yao, Tianchi Huang, Rui-Xiao Zhang, Ruiyu Li, and Lifeng Sun. Federated learning with unbiased gradient aggregation and controllable meta updating. arXiv preprint arXiv:1910.08234, 2019.

[84] Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Trong Nghia Hoang, and Yasaman Khazaeni. Bayesian nonparametric federated learning of neural networks. In ICML, 2019.

[85] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. arXiv preprint arXiv:1806.00582, 2018.

[86] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017.

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

We provide details omitted in the main paper.

• Appendix A: additional details of fractals pre-training (cf. section 5 of the main paper).
• Appendix B: details of experimental setups (cf. section 4 and section 6 of the main paper).
• Appendix C: additional experimental results and analysis (cf. section 4 of the main paper).
• Appendix D: additional discussions.

To keep the same reference numbers as in the main paper, we use plain text for those newly added references in the supplementary material.

A Additional Details of Fractal Pre-training

Figure 9: Examples of image pairs of FPS. We generate three image pairs with FPS using \( I = 2, 3, \) and 5 IFS codes. The resulting pairs are shown in (a), (b), and (c), which contain 2, 3, and 5 fractals in each image, respectively. In each pair, the two fractals rendered from the same IFS code reflect intra-class variations and different placements.

A.1 More details on fractal pair generation

We provide more details of our FPS algorithm described in section 5. To generate a fractal image pair \((x_1, x_2)\) for similarity learning, we first randomly sample \( I \) distinct IFS codes. Then, each IFS code is used to produce two fractal shapes applied with random coloring, resizing, and flipping. Finally, we obtain two sets of fractals; each set contains \( I \) distinct fractal shapes that can be pasted on a black canvas to generate one fractal image. The resulting two fractal images are further applied with image augmentations, such as random resized cropping, color jittering, flipping, etc, to get \( x_1 \) and \( x_2 \). We provide more examples of image pairs generated by our FPS in Figure 9. In each pair, the two fractals generated from the same IFS code show intra-class variations, in terms of shapes and colors, and are placed at random locations.

A.2 Self-supervised learning approaches

After generating fractal image pairs with FPS, we pre-train models by applying two self-supervised learning approaches, SimSiam [14] (similarity-based) and MoCoV2 [13] (contrastive-based). We briefly describe them in the following.

SimSiam. Given a positive image pair \((x_1, x_2)\), we process them by an encoder network \( f \) to extract their features. A prediction MLP head \( h \) is then applied to transform the features of one image to match the features of the other. Let \( p_1 = h(f(x_1)) \) and \( z_2 = f(x_2) \). The objective (to learn \( f \) and \( h \)) is to minimize the negative cosine similarity between them:

\[
D(p_1, z_2) = -\frac{p_1}{\|p_1\|_2} \cdot \frac{z_2}{\|z_2\|_2},
\]

where \( \| \cdot \|_2 \) is the \( l_2 \)-norm. Since the relation between \( x_1 \) and \( x_2 \) is symmetric, the final loss can be written as follows:

\[
\mathcal{L} = \frac{1}{2} D(p_1, z_2) + \frac{1}{2} D(p_2, z_1).
\]
MoCoV2. Similar to SimSiam \[14\], MoCoV2 \[13\] also aims to maximize the similarity between features extracted from a positive image pair \((x_1, x_2)\). The main difference is that MoCoV2 adopts the contrastive loss, which also takes negative pairs into account. Following the naming in MoCoV2 \[13\], let \(x^q = x_1\) be the query and \(x^k_i = x_2\) be the positive key. We also have negative images/keys \(\{x^k_1, x^k_2, \ldots, x^k_N\}\) in the mini-batch. We define \(q = f_q(x^q)\), \(k_+ = f_k(x^k_+\rangle\), and \(k_i = f_k(x^k_i)\), where \(f_q\) is the encoder for query images and \(f_k\) is the encoder for keys. The objective function for MoCoV2 (to learn \(f_q\) and \(f_k\)) is written as follows:

\[
\mathcal{L} = -\log \frac{\exp(q \cdot k_+ / \tau)}{\sum_{i=0}^{N} \exp(q \cdot k_i / \tau)},
\]

where \(\tau\) is a temperature hyper-parameter. Besides the contrastive loss, MoCoV2 maintains a dictionary to store and reuse features of keys from previous mini-batches, thereby making the negative keys not limited to the current mini-batch. Finally, to enforce stability during training, a momentum update is applied on the key encoder \(f_k\).

In subsection 6.2 and Table 2 of the main paper, we pre-train ResNet-18 and ResNet-20 for 100 epochs using SimSiam \[14\] and MoCoV2 \[13\]. Specifically, these neural network architectures are used for \(f\) in SimSiam and \(f_q\) and \(f_k\) in MoCoV2. Each epoch consists of 1M image pairs, which are generated on-the-fly, for FPS. For a comparison to StyleGAN in Table 2, we use the pre-generated StyleGAN dataset provided in \[3\], which has 1.3M images. After pre-training, we keep \(f_q\) from MoCoV2, following \[13\].

For SimSiam, we use the SGD optimizer with learning rate 0.05, momentum 0.9, weight decay 1e-4, and batch size 256. For MoCoV2, we use the SGD optimizer with learning rate 0.03, momentum 0.9, weight decay 1e-4, and batch size 256. The dictionary size is set to 65,536. For both SimSiam and MoCoV2, the learning rate decay follows the cosine schedule (Loshchilov et al., 2017).

For other experiments besides Table 2, we mainly use SimSiam for FPS. We use the same training setup as mentioned above (e.g., 100 epochs).

### B FL Experiment Details

#### B.1 Datasets, FL settings, and hyperparameters

We train FEDAVG for **100 rounds**, with 5 local epochs and weight decay 1e-4. Learning rates are decayed by 0.1 every 30 rounds. Besides that, we summarize the training hyperparameters for each of the federated experiments included in the main paper in Table 6. We always reserve 2% data of the training set as the validation set for hyperparameter searching for finalizing the setups.
Table 7: Sent140 accuracy (100 rounds; local epoch = 5, 10% clients per round).

| Init. / Setting | FL | CL | ∆CL-FL |
|-----------------|----|----|--------|
| Random          | 75.4 | 80.5 | 5.1 |
| Pre-trained     | 83.4 (+8.0) | 85.0 (+4.5) | 1.6 |

Table 8: Shakespeare next-character prediction accuracy (30 rounds; local epoch = 10, 10% clients per round, 606 clients in total).

| Init. / Setting | FL | CL | ∆CL-FL |
|-----------------|----|----|--------|
| Random          | 46.4 | 59.1 | 12.7 |
| Pre-trained     | 52.6 (+6.2) | 59.6 (+0.6) | 7.0 |

For pre-processing, we generally follow the standard practice which normalizes the images and applies some augmentations. CIFAR-10/100 images are padded 2 pixels on each side, randomly flipped horizontally, and then randomly cropped back to 32 × 32. For the other datasets with larger resolutions, we simply randomly cropped to the desired sizes and flipped horizontally following the official PyTorch ImageNet training script.

For the Cityscapes dataset, we use output stride 16. In training, the images are randomly cropped to 768 × 768 and resized to 2048 × 1024 in testing.

To further understand the effects of hyperparameters, we provide more analysis on Tiny-ImageNet in subsection C.4.

B.2 Optimal convex aggregation

We provide more details for learning the optimal convex experiment in section 4. To find the optimal combinations for averaging clients’ weights, we optimize λ using the SGD optimizer, with a learning rate 1e−5 for 20 epochs (batch size 32) on the global test set. The vector λ is ℓ1 normalized and each entry is constrained to be non-negative (which can be done with a softmax function in PyTorch) to ensure the combinations are convex. Since the optimization problem is not convex, we initialize λ with several different vectors, including uniform initialization, and return the best (in terms of the test accuracy) as \{λ⋆_m\} for Equation 5.

C Additional Experiments and Analyses

C.1 Scope and more experiments in NLP

In this paper, we focus on computer vision (CV). We respectfully think this should not degrade our contributions. First, CV is a big area. Second, while many federated learning (FL) works focus on CV, most of them merely studied classification using simple datasets like CIFAR. We go beyond them by experimenting with iNaturalist and Cityscapes (for segmentation), which are realistic and remain challenging even in centralized learning (CL). We believe our focused study in CV and encouraging results on these two datasets using either real or synthetic pre-training are valuable to the FL community.

That being said, here we provide two experiments on natural language processing (NLP) to verify that our observations are consistent with that on CV tasks. First, we conduct a sentiment analysis experiment on a large-scale federated Sent140 dataset (Caldas et al., 2019), which has 660K clients and 1.6M samples. We use a pre-trained DistilBERT (Sanh et al., 2019). Second, we experiment with another popular FL NLP dataset Shakespeare next-character prediction task proposed in (60). We use a version provided in (Caldas et al., 2019) which contains 606 clients, and use the Penn Treebank dataset (Marcinkiewicz et al., 1994) for pre-training a LSTM of two layers of 256 units each.

Table 7 and Table 8 show the results for the Sent140 and Shakespeare datasets, respectively. We also see a similar trend in Figure 10—pre-training helps more in FL than in CL and largely closes their gap, even if the local model drifts are not alleviated.

https://github.com/pytorch/examples/tree/master/395imagenet
Figure 10: Dynamics of FL on Sent140 and Shakespeare. We show the test accuracy of the global model $\bar{\theta}^{(t)}$ (×), and the averaged test accuracy using each local model $\tilde{\theta}_m^{(t)}$ (●).

Figure 11: Pre-training leads to a lower-variance loss surface (cf. Figure 4). We show the test loss by the FedAVG’s global model $\bar{\theta}^{(t)}$ (curve) and the 95% confidence interval of the test losses by the sampled convex combinations (shaded area) on iNaturalist-GEO-3K and Cityscapes.

C.2 More analysis on global aggregation

In section 4 we discuss how pre-training leads to more robust aggregation where the loss variance is smaller when we sample the convex combinations of local models. In Figure 4 we provide CIFAR-10 and Tiny-ImageNet due to the space limit. In Figure 11 we also include the iNaturalist-GEO-3K and Cityscapes. We have a consistent finding — using pre-training does lead to a lower-variance loss surface.

C.3 More comparisons on synthetic pre-training

In subsection 6.2 we compare synthetic pre-training methods. We also consider the images generated by (the best version of) a randomly-initialized StyleGAN [3, 42]. Due to the space limit, we only list some of the comparisons. Here we provide the complete results in Table 9. Across 3 datasets and 2 self-supervised algorithms, our FPS consistently outperforms the baselines. Since overall our FPS + SimSiam provides the strongest performance, we focus on it for other experiments.
Table 9: Comparison on synthetic pre-training methods in section 5. Means of 3 runs are reported.

| Init. / Dataset | C10       | C100      | Tiny-ImgNet |
|-----------------|-----------|-----------|-------------|
| Random          | 74.4±0.42 | 51.4±0.31 | 42.4±0.20   |
| Fractal + Multi-label | 73.0±0.43 | 51.0±0.44 | 40.9±0.22   |
| StyleGAN + SimSiam | 79.2±0.26 | 53.0±0.15 | 44.6±0.16   |
| Fractal + SimSiam | 77.4±0.30 | 51.7±0.28 | 44.2±0.20   |
| FPS (ours) + SimSiam | 80.5±0.25 | 54.7±0.11 | 45.7±0.14   |
| StyleGAN + MoCoV2 | 75.3±0.11 | 53.5±0.12 | 45.8±0.24   |
| Fractal + MoCoV2 | 73.5±0.29 | 53.3±0.24 | 44.5±0.17   |
| FPS (ours) + MoCoV2 | 79.9±0.15 | 53.9±0.16 | 46.1±0.18   |
| ImageNet / Places365 | 87.9±0.11 | 62.2±0.09 | 50.3±0.08   |

Figure 12: Tiny-ImageNet (10 clients, Dir(0.3), 100% participation, local epoch= 5) with different learning rate decay strategies: we show Global model (×) accuracy and the averaged accuracy using each local model (●).

C.4 More settings on Tiny-ImageNet

To understand how the FL settings affect the observations of pre-training, we further conduct studies on Tiny-ImageNet. We focus on the setting in the main text (i.e., 10 clients, Dir(0.3), 100% participation, local epoch= 5) but change one variable of settings such as learning rate scheduling in Figure 12 and the number of clients, non-IID degree, participation rate, and the number of local epochs in Figure 13. In Figure 6 of the main text, we report the mean for brevity, here we present the standard deviations as well over 3 different runs of experiments, which are generally small.

For different learning rate scheduling, we observe similar trends: pre-training does not alleviate client shift much but the test accuracy of the global model after aggregating the local models is higher. In the main paper, we focus on the schedule that decays the learning rate by 0.1 every 30 rounds given its better performance compared to the two (decay by 0.95 every round and no decays) here.

For client numbers, more clients will make the performance drop because each client has fewer data and the resulting local training is prone to over-fitting. However, using pre-training on both real data and our FPS still improves significantly.

For the non-IID degree, we manipulate it with the Dirichlet distributions by the α parameter. We observe that more non-IID (smaller α) settings degrade the performance of using random initialization sharply, while using pre-training is more robust (smaller accuracy drop).

For participation rates and the number of local epochs, we observe that they are not very sensitive variables, as long as they are large enough (e.g., participation rate > 30% and #local epochs > 5). Interestingly, using either fewer or more local epochs does not close the gap between using pre-training and random initialization.
Figure 13: Tiny-ImageNet (10 clients, Dir(0.3), 100% participation, local epoch = 5) with one variable changed. 

Mean ± std overall 3 runs are reported.

D Additional Discussions

D.1 Potential negative societal impacts

Our work discusses how using pre-training on a large dataset can improve FL. While not specific to our discussions, collecting the real dataset could potentially inject data bias or undermine data privacy if the collection process is not carefully-designed. However, we believe the concerns are mainly from data collections but not about the learning algorithms.

To remedy this, in [section 5] we propose an alternative of using synthetic data that does not require any real data and it can still improve FL significantly.

D.2 Computation resources

We implement all the codes in PyTorch and train the models with GPUs. For experiments with $32 \times 32$ images, we trained on a 2080 Ti GPU. Pre-training takes about 1 day and FL takes about 8 hours. For experiments with $224 \times 224$ images, we trained on an A6000 GPU. Pre-training takes about 2 − 3 days. and FL takes about 1 day. For the Cityscape dataset, we trained with an A6000 GPU for about 2 days.