Co-Dream: Collaborative Dream Synthesis over Decentralized Models

Abhishek Singh[1*], Gauri Gupta[1*], Yichuan Shi[1], Alex Dang[1], Ritvik Kapila[2], Sheshank Shankar[3], Mohammed Ehab[1], Ramesh Raskar[1]

[1]Massachusetts Institute of Technology
[2]Amazon
[3]Tesla AI
abhi24@mit.edu

Abstract

Federated Learning (FL) has pioneered the idea of "share wisdom not raw data" to enable collaborative learning over decentralized data. FL achieves this goal by averaging model parameters instead of centralizing data. However, representing "wisdom" in the form of model parameters has its own limitations including the requirement for uniform model architectures across clients and communication overhead proportional to model size.

In this work we introduce `Co-Dream` a framework for representing "wisdom" in data space instead of model parameters. Here, clients collaboratively optimize random inputs based on their locally trained models and aggregate gradients of their inputs. Our proposed approach overcomes the aforementioned limitations and comes with additional benefits such as adaptive optimization and interpretable representation of knowledge. We empirically demonstrate the effectiveness of `Co-Dream` and compare its performance with existing techniques.

Code — https://mitmedialab.github.io/codream.github.io/

Introduction

Machine learning (ML) model training is often hindered by the fragmented nature of data ownership. Federated Learning (FL) (McMahan et al. 2023) addresses this problem by aggregating clients' models centrally instead of their data. While FL does not give any privacy guarantee, it reduces privacy concerns by (1) sharing clients' models instead of their raw data and (2) using a linear operation (weighted average) to aggregate models that is amenable to secure aggregation techniques.

However, FL requires all clients to agree on the same model architecture. If the model has a large number of parameters, it may not be supported by all participating device hence reducing the number of participants. Some recent knowledge-distillation (KD) (Mora et al. 2022) techniques allow clients to share knowledge while allowing heterogeneous models. However, these KD algorithms depart from the model averaging paradigm, and hence incompatible with secure aggregation.

We propose a novel framework to solve this problem by aggregating collaboratively optimized representations of data (which we call *dreams*) instead of parameters. We show that dreams capture the knowledge embedded within local models and also facilitate the aggregation of local knowledge. Our key idea is to apply federated optimization on randomly initialized samples to extract knowledge from the client's local models trained on their original dataset. The goal of optimizing *dreams* is to enable KD, rather than generate realistic synthetic data.

The key benefits of our approach are: (1) **Flexibility**: Our proposed technique, `Co-Dream`, collaboratively optimizes *dreams* to aggregate knowledge from the client's local models. By sharing *dreams* in the data space rather than model parameters, our method is model-agnostic. (2) **Scalability**: Furthermore, communication does not depend on the model parameter size, alleviating scalability concerns. (3) **Privacy**: Just like `FedAvg` (McMahan et al. 2017), `Co-Dream` does not come with privacy guarantee but enhances privacy in two ways: Firstly, clients share *dreams'* updates, never raw data. Secondly, the linearity of the aggregation algorithm allows clients to securely aggregate their *dreams* without revealing their individual updates to the server.

Our framework comprises three stages: knowledge extraction , knowledge aggregation , and knowledge acquisition . We test `Co-Dream` by establishing the feasibility of `Co-Dream` as a way for clients to synthesize samples collaboratively and learn predictive models, validating `Co-Dream` as an alternative to FL. We empirically validate our framework by benchmarking with existing algorithms and conducting ablation studies across various design choices.

Preliminaries

Federated Learning (FL) minimizes the expected risk $\min_\theta \mathbb{E}_{\mathcal{D}\sim p(\mathcal{D})}\ell(\mathcal{D},\theta)$ where θ is the model parameters, $\mathcal{D}$ is a tuple of samples $(X \in \mathcal{X}, Y \in \mathcal{Y})$ of labeled data in supervised learning in the data space $\mathcal{X} \subset \mathbb{R}^d$ and $\mathcal{Y} \subset \mathbb{R}$, and ℓ is some risk function such as mean square error or cross-entropy (Konečnỳ et al. 2016; McMahan et al. 2023). Without directly accessing the true distribution, FL optimizes the empirical risk instead given by:

$$\min_\theta \sum_{k\in K} \frac{1}{|\mathcal{D}_k|}\ell(\mathcal{D}_k,\theta), \tag{1}$$

*These authors contributed equally.

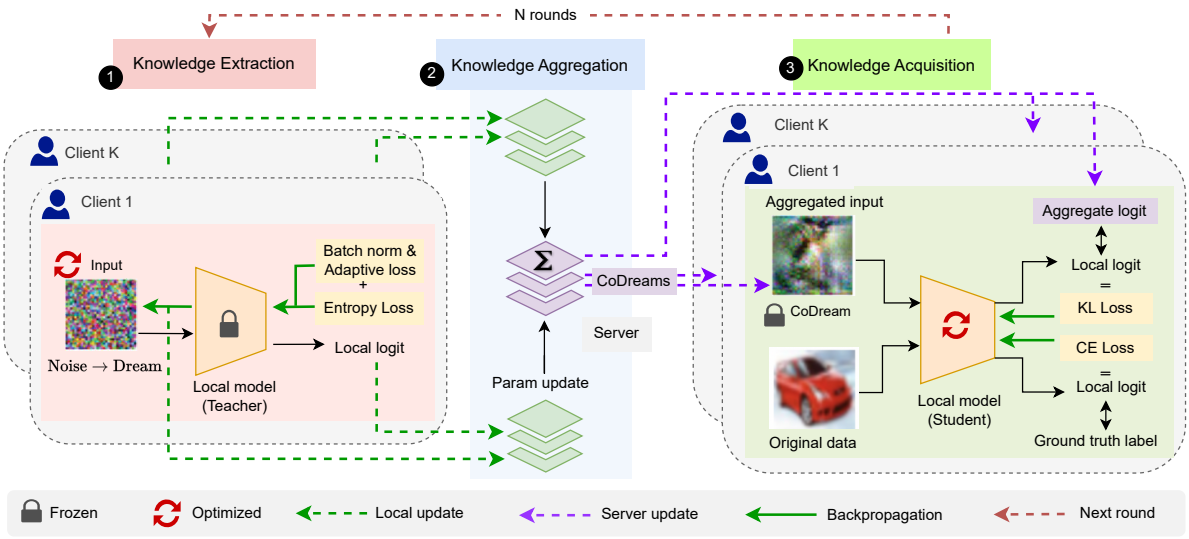

Figure 1: **Overview of the `Co-Dream` pipeline** comprising three stages: (1) Knowledge Extraction— each client generates *dreams*, representing the extracted knowledge from their local models (teacher). Starting with random noise images and frozen teacher models, clients optimize to reduce entropy on the output distribution while regularizing the batch norm and adaptive loss. The clients share their local updates of *dreams* and logits with the server. (2) Knowledge Aggregation—server aggregates *dreams* and soft labels from clients to construct a `Co-Dream` dataset. (3) Knowledge Acquisition—clients update their local models through two-stage training (i) on jointly optimized *co-dreams* with knowledge distillation (where clients act as students) and (ii) local dataset with cross-entropy loss.

The dataset $\mathcal{D}$ is distributed among K clients, with each client k holding a portion $\mathcal{D}_k$, such that $\mathcal{D} = \cup_{k \in K} \mathcal{D}_k$. The server broadcasts the global model θ^r to all clients, who then locally optimize it for M rounds to obtain $\theta_k^{r+1} = \arg\min_\theta \ell(\mathcal{D}_k, \theta^r)$. Each client sends its updated model θ_k^{r+1} or the difference $\theta_k^{r+1} - \theta_k^r$ (the pseudo-gradient) back to the server, which aggregates these updates and sends the new global model back to the clients.

Knowledge Distillation facilitates the transfer of knowledge from a teacher model ($f(\theta_T)$) to a student model ($f(\theta_S)$) by incorporating an additional regularization term into the student's training objective (Buciluǎ, Caruana, and Niculescu-Mizil 2006; Hinton et al. 2015). This regularization term (usually computed with Kullback-Leibler (KL) divergence $\mathsf{KL}(f(\theta_T, \mathcal{D}) || f(\theta_S, \mathcal{D}))$) encourages the student's output distribution to match the teacher's outputs.

DeepDream for Knowledge Extraction (Mordvintsev, Olah, and Tyka 2015) first showed that features learned in deep learning models could be *extracted* using gradient-based optimization in the feature space. Randomly initialized features are optimized to identify patterns that maximize a given activation layer. Regularization such as TV-norm and ℓ_1-norm has been shown to improve the quality of the resulting images. Starting with a randomly initialized input $\hat{x} \sim \mathcal{N}(0,\ I)$, label y, and pre-trained model f_θ, the optimization objective is

$$\min_{\hat{x}} \mathsf{CE}\left(f_\theta(\hat{x}),\ y\right) \ +\ \mathcal{R}(\hat{x}), \tag{2}$$

where CE is cross-entropy and $\mathcal{R}$ is some regularization. DeepInversion (Yin et al. 2020) showed that the knowledge distillation could be further improved by matching batch normalization statistics with the training data at every layer.

Related Work

Generative modeling techniques either pool locally generated data on the server (Song et al. 2022; Goetz and Tewari 2020) or use `FedAvg` with generative models (Rasouli, Sun, and Rajagopal 2020; Xin et al. 2020). Like FL, `FedAvg` over generative models is also not model agnostic. While we share the idea of generative data modeling, we do not expose individual clients' updates or models directly to the server.

Knowledge Distillation in FL is an alternative to `FedAvg` that aims to facilitate knowledge sharing among clients that cannot acquire this knowledge individually (Chang et al. 2019; Lin et al. 2020; Afonin and Karimireddy 2022; Chen and Chao 2021). However, applying KD in FL is challenging because the student and teacher models need to access the same data, which is difficult in FL settings.

Data-free Knowledge Distillation algorithms address this challenge by employing a generative model to generate synthetic samples as substitutes for the original data (Zhang et al. 2022a,b; Zhu, Hong, and Zhou 2021). These data-free KD approaches are not amenable to secure aggregation and must use the same architecture for the generative model.

Overall, these existing approaches lack active client collaboration in the knowledge synthesis process. Clients share their local models or locally generated data with the server without contributing to knowledge synthesis. We believe that collaborative synthesis is crucial for secure aggregation and bridging the gap between KD and FL. Our approach `Co-Dream` enables clients to synthesize dreams collaboratively while remaining compatible with secure aggregation techniques and being model agnostic.

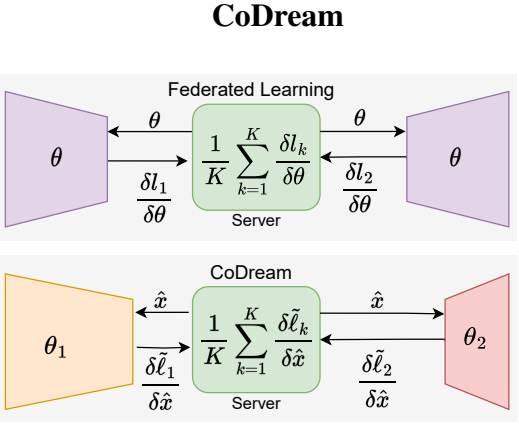

Figure 2: **Comparing aggregation framework in FL and `Co-Dream`.** In FL, the server aggregates the gradients of model parameters, whereas, in `Co-Dream`, aggregation happens in the gradients of the data space, called dreams ($\hat{x}$), allowing for different model architectures. Here K is the number of clients and $l, \tilde{l}$ are loss functions given in Eq 1 and Eq 3.

Our approach `Co-Dream` consists of three key stages.

In the **knowledge extraction** stage, each client extracts useful data representations, referred to as "dreams", from their locally trained models. Starting with random noise images $\hat{x}$, clients $k \in K$ optimize these images using objective $\tilde{\ell}$ to facilitate knowledge sharing from their local models with parameters θ_k (Section). Since this is a gradient-based optimization of the input *dreams*, we exploit the linearity of gradients to enable knowledge aggregation from all the clients:

$$\nabla_{\hat{x}}\left(\mathbb{E}_{k\sim K}[\tilde{\ell}(\hat{x},\ \theta_k)]\right) = \mathbb{E}_{k\sim K}\left[\nabla_{\hat{x}}(\tilde{\ell}(\hat{x},\ \theta_k))\right]$$

In the **knowledge aggregation** stage, the clients now jointly optimize these random noised images by aggregating the gradients from the local optimizations (Section). Unlike `FedAvg`, our aggregation occurs in the input data space over these *dreams*, making our approach compatible with heterogeneous client models.

Finally, in the **knowledge acquisition** stage, these collaboratively optimized images, or *dreams*, are then used for updating the server and clients without ever sharing the raw data or models. Specifically, clients act as students and train their models on the global *dreams* (Section). Figure 1 gives an overview of the `Co-Dream` pipeline for each round. We now discuss these stages in more detail.

Local dreaming for knowledge extraction

First, clients perform local *dreaming*, a model-inversion approach to extract information from their trained models. We use DeepDream (Mordvintsev, Olah, and Tyka 2015) and DeepInversion (Yin et al. 2020) approaches that enable data-free knowledge extraction from the pre-trained models. However, these are not directly applicable to a federated setting because the client models are continuously evolving, as they learn from their own data as well as other clients. A given client should synthesize only those *dreams* over which they are highly confident. As the client models evolve, their confidence in model predictions also changes over time. A direct consequence of this non-stationarity is that it is unclear how the label y should be chosen in Eq 2. In DeepInversion, the teacher uniformly samples y from its own label distribution because the teacher has the full dataset. However, in the federated setting, data is distributed across multiple clients with heterogeneous data distributions.

To keep track of a given client's confidence, we take a simple approach of treating the entropy of the output distribution as a proxy for the teachers' confidence. We adjust Eq 2 so that the teacher synthesizes *dreams* without any classification loss by instead minimizing the entropy (denoted by $\mathcal{H}$) on the output distribution. Each client (teacher) starts with a batch of representations sampled from a standard Gaussian ($\hat{x} = \mathcal{N}(0, 1)$), and optimizes *dreams* using Eq 3. Formally, we optimize the following objective for synthesizing *dreams*:

$$\min_{\hat{x}}\left\{\tilde{\ell}(\hat{x},\ \theta) \ =\ \mathcal{H}\left(f_\theta(\hat{x})\right) + \mathcal{R}_{bn}(\hat{x}) + \mathcal{R}_{adv}(\hat{x})\right\} \quad (3)$$

where $\mathcal{H}$ is the entropy for the output predictions, $\mathcal{R}_{bn}$ is the feature regularization loss and $\mathcal{R}_{adv}$ is a student-teacher adversarial loss. To improve the *dreams* image quality, we enforce feature similarities at all levels by minimizing the distance between the feature map statistics for *dreams* and training distribution, which is stored in the batch normalization layers. Hence, $\mathcal{R}_{bn}(\hat{x}) = \sum_l ||\mu^l_{feat} - \mu^l_{bn}|| + ||\sigma^l_{feat} - \sigma^l_{bn}||$. Further, to increase the diversity in generated *dreams*, we add an adversarial loss to encourage the synthesized images to cause student-teacher disagreement. $\mathcal{R}_{adv}$ penalizes similarities in image generation based on the Jensen-Shannon divergence between the teacher and student distribution, $\mathcal{R}_{adv}(\hat{x}) = -JSD(f_t(\hat{x})||f_s(\hat{x}))$, where the client model is the teacher and the server model is the student model. To do this adaptive teaching in a federated setting, the server shares the gradient $\nabla_{\hat{x}} f_s(\hat{x})$ with the clients for local adaptive extraction. The clients then locally calculate $\nabla_{\hat{x}}\tilde{\ell}(\hat{x},\ \theta_k)$ which is then aggregated at the server for knowledge aggregation in Eq 4. Thus, $\mathcal{R}_{adv}$ helps extract knowledge from the clients that the clients know and the server does not know.

Unlike generative models that generate data with objective to resemble the real data, the goal of optimizing *dreams* is to perform knowledge distillation. Therefore, as shown in Figure 6, *dreams* do not resemble real images.

Collaborative dreaming for knowledge aggregation

Since the data is siloed and lies across multiple clients, we want to extract the collective knowledge from the distributed system. While `FedAvg` aggregates gradients of the model updates from clients, it assumes the same model architecture across clients and thus is not model-agnostic.

We propose a novel mechanism for aggregating the knowledge by collaboratively optimizing *dreams* across different clients. Instead of each client independently synthesizing *dreams* using Eq 3, they now collaboratively optimize them by taking the expectation over each client's local loss w.r.t.

the same $\hat{x}$: $\min_{\hat{x}} \mathbb{E}_{k \in K} \left[\tilde{\ell}(\hat{x},\ \theta_k) \right]$. This empirical risk can be minimized by computing the local loss at each client. Therefore, the update rule for $\hat{x}$ can be written as:

$$\hat{x} \leftarrow \hat{x} - \nabla_{\hat{x}} \sum_{k \in K} \frac{1}{|\mathcal{D}_k|} \tilde{\ell}(\hat{x},\ \theta_k)$$

Using the linearity of gradients, we can write it as

$$\hat{x} \leftarrow \hat{x} - \sum_{k \in K} \frac{1}{|\mathcal{D}_k|} \nabla_{\hat{x}} \tilde{\ell}(\hat{x},\ \theta_k) \tag{4}$$

Clients compute gradients locally on a shared input and send them to the server, which aggregates the gradients and returns the updated input. This approach, like distributed-SGD, optimizes in the data space rather than the model parameter space. As a result, `Co-Dream` is model-agnostic (Fig 2) and compatible with existing cryptographic aggregation methods, since only the aggregated output is revealed, not individual gradients.

We experimentally demonstrate that collaborative optimization indeed embeds the knowledge from multiple client models in the same dream dataset.

Knowledge acquisition

Finally, the local clients and the server act as students and update their models using the collaboratively trained *dreams* obtained from Eq 4. The clients share soft logits for each *dream*, which are then aggregated by the server to perform knowledge distillation on the following objective:

$$\min_{\theta} \sum_{\hat{x} \in \hat{\mathcal{D}}} \mathsf{KL} \left(\sum_{k} \frac{1}{|\mathcal{D}_k|} f_{\theta_k}(\hat{x}) \,\middle\|\, f_{\theta}(\hat{x}) \right) \tag{5}$$

We provide the complete algorithm of `Co-Dream` in Algorithm 1. Note that the choice of parameters such as local updates M, global updates R, local learning rate η_l, global rate η_g, and the number of clients K typically guide the trade-off between communication efficiency and convergence.

Analysis of `Co-Dream`

The benefits of `Co-Dream` are inherited from using KD, along with additional advantages arising from our specific optimization technique. `Co-Dream` extracts the knowledge from clients in *dreams* and shares the updates of these dreams instead of model gradients (∇_θ) as done in FL.

Communication Analysis: We use the following notation: d is the dimension of the inputs or *dreams*, n is the batch size of *dreams* generated, and R is the number of aggregation rounds. In `FedAvg` and its variants, the communication is $|\theta| \times R$. Since `Co-Dream` communicates input gradients ($\nabla_{\hat{x}}$) instead of model gradients (∇_θ), the total communication is $d \times n \times R$. For heavily parameterized models, $d \times n \ll |\theta|$. In a single batch, the communication complexity of `Co-Dream` does not scale with larger models. Table 3 provides a comprehensive communication analysis for different model architectures in `FedAvg` vs `Co-Dream`.

Privacy Analysis: Various model inversion and reconstruction attacks (Haim et al. 2022; Hitaj, Ateniese, and Perez-Cruz 2017) have shown private sensitive information can be reconstructed from just the model weights. While several reconstruction attacks perform model inversion, `Co-Dream` is optimized for improving performance on knowledge distillation. However, in `Co-Dream`, the clients collaborate by sharing the gradients of *dreams'* without even sharing their model parameters. A simple application of data processing inequality shows that sharing dreams is at least as private as sharing model parameters. Similar to `FedAvg`, the synchronization step between the clients is a linear operation (weighted average) and hence offers an additional layer of privacy by using secure aggregation (Bonawitz et al. 2017). Finally, table 4 shows that our approach empirically outperforms benchmarks against the state-of-the-art LiRA membership inference attack (Carlini et al. 2022).

Flexibility of models: Since the knowledge aggregation in `Co-Dream` is done by sharing the updates of *dreams* in data space, `Co-Dream` is model agnostic and allows for collaboration among clients with different model architectures. We empirically observe no performance drop in collaborative learning with clients of different model architectures.

Customization in sharing knowledge: Additionally, sharing knowledge in the data space enables adaptive optimization, such as synthesizing adversarially robust samples or class-conditional samples for personalized learning.

Algorithm 1: `Co-Dream` Algorithm

Input: Number of client K, local models and data θ_k and $\mathcal{D}_k, k \in K$, local learning rate η_k, global learning rate η_g, local training rounds M, global training epochs R, total number of epochs N.

for $t = 1$ **to** N **do**
  Server initializes a batch of dreams $\hat{x} \sim \mathcal{N}(0, 1)$;
  for $r = 1$ **to** R **do**
    Server broadcasts $\hat{x}^r$ to all clients
    for *each client* $k \in K$ ***in parallel*** **do**
      $\hat{x}^r_{k,0} := \hat{x}^r$;
      for $m = 1$ **to** M **do**
        // Local knowledge extraction (Eq 3)
        $\hat{x}^r_{k,m} \leftarrow \hat{x}^r_{k,m-1} - \eta_k \cdot \nabla_x(\tilde{\ell}(\hat{x}^r_{k,m-1}, \theta_k))$
      end
      each client shares pseudo-gradient $\nabla \hat{x}^r_k = \hat{x}^r_{k,M} - \hat{x}^r$ with the server;
    end
    // Collaborative knowledge aggregation (Eq 4)
    $\hat{x}^{r+1}_S \leftarrow \hat{x}^r + \eta_g \sum_{k \in K} \frac{1}{|\mathcal{D}_k|} \nabla \hat{x}^r_k$;
    // Server aggregates model predictions
    $\hat{\mathcal{D}} := \{\hat{x}^{r+1}, \hat{y}^{r+1}_S := \sum_k \frac{1}{|\mathcal{D}_k|} f_{\theta_k}(\{\hat{x}^{r+1}\})\}$;
    // Local knowledge acquisition (Eq 5)
    for *each client* $k \in K$ ***in parallel*** **do**
      LocalUpdate($\hat{\mathcal{D}}, \theta_k$); LocalUpdate($\mathcal{D}_k, \theta_k$);
    end
    LocalUpdate($\hat{\mathcal{D}}, \theta_s$);
  end
end

Model	Heterogeneous Clients (Independent clients 1-4)				Method			
	WRN-16-1	VGG-11	WRN-40-1	ResNet-34	Independent	Centralized	AvgKD	Co-Dream (ours)
iid($\alpha = \inf$)	52.2	55.1	43.5	54.2	$51.6_{(4.5)}$	68.8	$52.9_{(1.4)}$	**69.6**$_{(1.0)}$
$\alpha = 10$	19.1	23.6	27.6	16.2	$19.7_{(1.7)}$	58.5	$50.4_{(1.3)}$	**62.2**$_{(2.6)}$
$\alpha = 1$	41.3	38.2	37.1	50.1	$41.7_{(5.1)}$	**64.8**	$42.4_{(2.9)}$	$60.0_{(1.7)}$
$\alpha = 0.1$	29.1	22.3	33.1	21.5	$27.2_{(4.9)}$	**43.0**	$30.2_{(3.3)}$	$40.6_{(0.9)}$

Table 1: **Performance comparison with heterogeneous client models**: on CIFAR10 dataset. Left: Accuracy for independent heterogeneous clients with different models; Right: Average client model performance comparison of Co-Dream with other baselines

Method	MNIST			SVHN			CIFAR10		
	iid($\alpha = \inf$)	$\alpha = 1$	$\alpha = 0.1$	iid($\alpha = \inf$)	$\alpha = 1$	$\alpha = 0.1$	iid($\alpha = \inf$)	$\alpha = 1$	$\alpha = 0.1$
Centralized	$85.0_{(0.9)}$	$61.4_{(7.1)}$	$36.9_{(7.6)}$	$80.8_{(1.3)}$	$75.6_{(1.4)}$	$54.6_{(13.6)}$	$65.7_{(2.9)}$	$65.3_{(0.4)}$	$45.5_{(6.8)}$
Independent	$52.4_{(7.0)}$	$36.3_{(6.2)}$	$22.0_{(4.2)}$	$51.3_{(9.2)}$	$42.3_{(6.4)}$	$19.6_{(9.2)}$	$46.4_{(2.0)}$	$39.7_{(3.4)}$	$23.5_{(5.2)}$
FedAvg	$84.7_{(1.6)}$	$60.3_{(3.4)}$	$40.0_{(6.9)}$	$82.9_{(0.4)}$	$79.1_{(0.9)}$	$47.1_{(23.7)}$	$67.2_{(0.4)}$	$62.3_{(0.9)}$	$34.8_{(8.3)}$
FedProx	$78.6_{(3.5)}$	$62.6_{(3.6)}$	$38.1_{(11.0)}$	$86.9_{(0.1)}$	$84.3_{(0.6)}$	$48.7_{(26.7)}$	$70.8_{(1.8)}$	$62.3_{(2.9)}$	$27.1_{(9.8)}$
Moon	$85.1_{(2.6)}$	$66.2_{(4.4)}$	$42.3_{(11.8)}$	$80.1_{(0.1)}$	$76.5_{(1.2)}$	$41.7_{(21.8)}$	$66.6_{(1.4)}$	$64.8_{(0.8)}$	$35.5_{(10.8)}$
AvgKD	$61.3_{(2.3)}$	$44.3_{(4.8)}$	$21.4_{(4.3)}$	$75.4_{(0.7)}$	$61.2_{(4.6)}$	$20.7_{(10.9)}$	$54.2_{(0.9)}$	$46.4_{(3.3)}$	$25.9_{(6.2)}$
SCAFFOLD	$87.5_{(0.6)}$	$70.2_{(3.6)}$	$38.8_{(13.7)}$	$86.0_{(0.1)}$	$84.5_{(0.7)}$	$13.5_{(4.4)}$	$73.9_{(1.5)}$	$67.5_{(4.6)}$	$22.8_{(7.8)}$
FedGen	$64.5_{(1.9)}$	$51.0_{(4.3)}$	$31.4_{(7.4)}$	$49.7_{(1.6)}$	$44.2_{(4.1)}$	$34.9_{(19.7)}$	$66.2_{(0.4)}$	$62.8_{(1.8)}$	$40.2_{(9.0)}$
Co-Dream (ours)	$80.6_{(0.5)}$	$57.7_{(3.6)}$	$35.7_{(9.2)}$	$81.4_{(0.1)}$	$80.1_{(0.8)}$	$44.5_{(17.7)}$	$69.5_{(0.3)}$	$64.8_{(0.3)}$	$36.6_{(8.4)}$

Table 2: Performance overview of different techniques with different data settings. A smaller α indicates higher heterogeneity.

Experiments

We systematically experiment and evaluate multiple aspects of Co-Dream. Unless stated otherwise, we used ResNet-18 (He et al. 2015) for training the client and server models and set the total number of clients $K = 4$. We conduct our experiments on 3 real-world datasets, including MNIST (LeCun et al. 1998), SVHN (Netzer et al. 2011), and CIFAR10 (Krizhevsky, Hinton et al. 2009). To validate the effect of collaboration, we train clients with 50 samples per client for MNIST and 1000 samples per client for CIFAR10 and SVHN datasets. For reference, we include two unrealistic baselines — Independent and Centralized. In the Centralized baseline, all the client data are aggregated in a single place. In the Independent baseline, clients only learn from their local data.

To simulate real-world conditions, we perform experiments on both IID and non-IID data. We use Dirichlet distribution $Dir(\alpha)$ to generate non-IID data partition among labels for a fixed number of total samples at each client. The parameter α guides the degree of imbalance in the training data distribution. A small α generates more skewed data.

Fast dreaming for knowledge extraction

Despite the impressive results of the original DreamInversion (Yin et al. 2020), we find 2000 local iterations to be too slow for a single batch of image generation when performed collaboratively in Co-Dream. Therefore, we use the Fast-datafree (Fang et al. 2022) approach that learns a prior for initializing dreams rather than initializing with random noise every time, to speed up image generation by a factor of 10 to 100 while preserving the performance. However, in each aggregation round, the client now shares both the local generator model and the dreams for aggregation by the server. Instead of 2000 global aggregation rounds (R) in Co-Dream, CoDream-fast performs only a single global aggregation round with 5 local rounds. We perform several subsequent experiments using CoDream-fast. More details on the implementation can be found in the Supplement material.

Model-agnostic collaborative learning

Since Co-Dream shares updates in the data space instead of the model space, our approach is model agnostic. We evaluate our approach across heterogeneous client models including ResNet-34 (He et al. 2016), VGG-11 (Simonyan and Zisserman 2014), and Wide-ResNets (Zagoruyko and Komodakis 2016) (WRN-16-1 and WRN-40-1). Table 1 shows the performance of Co-Dream against Centralized, Independent, and model agnostic FL baselines such as Avg-KD (Afonin and Karimireddy 2022). Note that FedGen is not completely model agnostic as it requires the client models to have a shared feature extractor and thus cannot be applied to our setting. We exclude FedAvg as it doesn't support heterogeneous models. Performing FL under both heterogeneous models and non-IID data distribution is a challenging task, yet Co-Dream outperforms the baselines.

Communication efficiency

We compare the client communication cost of Co-Dream and FedAvg per round for different model architectures in Table 3. In FedAvg, the clients share the model with the server, whereas, in Co-Dream, they share the *dreams*(size of data). However, in Co-Dream, each batch of *dreams* is refined for 400 rounds, whereas in CoDream-fast there is only a single round of aggregation along with the sharing of a lightweight generator model (as explained in Section . The communication of both Co-Dream and CoDream-fast is model agnostic and does not scale with large models.

Model	FedAvg	Co-Dream	CoDream-fast
Resnet34	166.6 MB	600 MB	23.5MB
Resnet18	89.4 MB	600 MB	23.5MB
VGG-11	1013.6 MB	600 MB	23.5MB
WRN-16-1	1.4 MB	600 MB	23.5MB
WRN-40-1	4.5 MB	600 MB	23.5MB

Table 3: Communication analysis of FedAvg vs Co-Dream and CoDream-fast per round

Varying number of clients

We test whether Co-Dream actually encapsulates knowledge from multiple clients. We test this hypothesis by aggregating knowledge by varying the number of clients $K = [2, 4, 8, 12, 24]$, while keeping the total dataset size constant. Thus, as K increases, each client has fewer local samples. As expected, performance declines with more clients, since each client's knowledge is less representative of the overall distribution. However, this drop is sublinear (Figure 3), making Co-Dream viable for cross-device federated learning. The gap between Co-Dream and FedAvg remains similar across different K.

In summary, Co-Dream sees a graceful decline in accuracy as data gets more decentralized. The framework effectively distills collective knowledge, even when local datasets are small. We conclude that averaging gradients in data space can combine the knowledge in the similar way as averaging gradients in the space of model parameters.

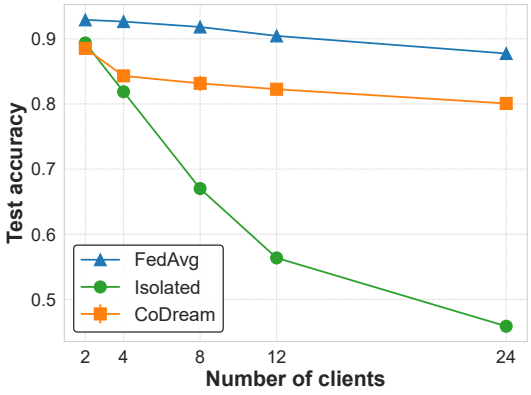

Figure 3: **Comparison by varying the number of clients.** The performance gap widens between Co-Dream and independent optimization as we increase the number of clients.

Non-IID datasets benchmarking

We evaluate the feasibility of Co-Dream for both IID and non-IID settings by varying $\alpha = 0.1, 0.5$ and report the performances of different methods in Table 2. We include popular non-IID specific algorithms such as FedProx (Li et al. 2020), Moon(Li, He, and Song 2021), and Scaffold (Karimireddy et al. 2020). We also include other model-agnostic federated baselines such as FedGen(Zhu, Hong, and Zhou 2021) and AvgKD (Afonin and Karimireddy 2022). The results show that Co-Dream is competitive with other non-iid techniques across all datasets and data partitions. Even as α becomes smaller (i.e., data become more imbalanced), Co-Dream still performs well. Note that Co-Dream does not beat other state-of-the-art non-iid techniques since it is not designed for the non-iid data challenges. It is analogous to FedAvg in the data space, and thus, existing non-iid tricks can also be applied to Co-Dream to improve its accuracy further.

Analysis of sample complexity of dreams

We plot the accuracy of the server model trained from scratch against the number of batches of *dreams* it is trained on as shown in Fig 4. Note that the quality of generated dreams for training increases as training progresses in each round. We observe that 5 batches per round is a good enough size, after which the marginal gain is very small.

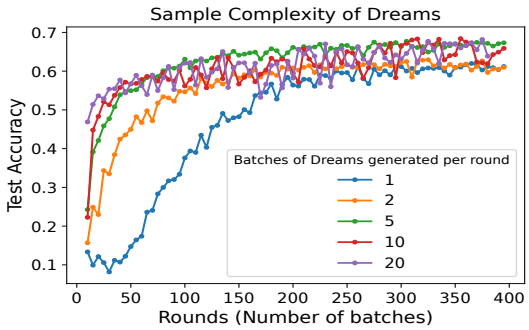

Figure 4: **Sample complexity of dreams** in reaching target accuracy. The performance improvement saturates as we add more batches.

Validating knowledge-extraction based on Eq 3

We evaluate whether the knowledge-extraction approach (Sec) allows for the effective transfer of knowledge from teacher to student. We first train a teacher model from scratch on different datasets, synthesize samples with our knowledge-extraction approach, and then train a student on the extracted *dreams*. To validate its compatibility within an FL setting where clients have a small local dataset, we reduce the size of the training set of the teacher to reduce its local accuracy and measure corresponding student performance. Results in Fig 5 show that the teacher-student performance gap does not degrade consistently even when the teacher's accuracy is low. This result is interesting because the extracted features get worse in quality as we decrease the teacher accuracy, but the performance gap is unaffected.

Collaborative vs Independent optimization

We evaluate the effectiveness of collaborative optimization of *dreams* over multiple clients in aggregating the knowledge by comparing the performance of collaboratively optimized *dreams* in Co-Dream (using Eq 3) with independently optimized *dreams*. We show the aggregation step in Eq 3 not only helps in secure averaging, leading to more privacy but also improves the performance (Table 5, last row).

Metric	CIFAR10			CIFAR100		
	Single Model	FedAvg	CoDream-fast (ours)	Single Model	FedAVG	CoDream-fast (ours)
Balanced Accuracy	57.39%	54.12%	**50.77%**	79.86%	51.84%	**50.42%**
AUROC	65.58%	63.17%	**59.72%**	77.01%	58.24%	**57.53%**
TPR @ 0.1% FPR	72.39%	53.71%	**39.36%**	39.63%	34.37%	**31.40%**
TPR @ 0.001% FPR	11.34%	5.00%	**2.66%**	30.41%	1.98%	**0.66%**

Table 4: Evaluation of privacy leakage by performing Membership Inference Attack

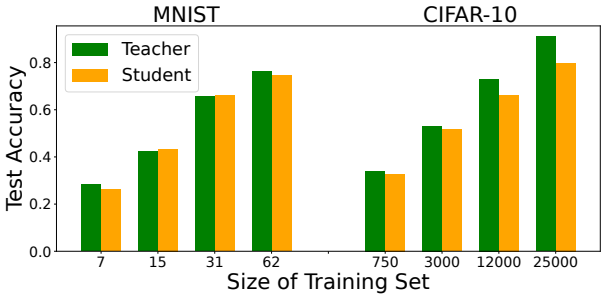

Figure 5: **Validating the effectiveness of knowledge transfer from teacher to student**: We vary the size of the training dataset (on the x-axis) for the teacher and compare its accuracy with the student trained on dreams generated using Eq 3

Data partition	iid	$\alpha = 1$	$\alpha = 0.1$
Co-Dream	$69.2_{(0.1)}$	$61.6_{(0.5)}$	$45.6_{(1.5)}$
w/o $\mathcal{R}_{adv}$	$65.7_{(0.2)}$	$58.4_{(1.3)}$	$42.0_{(1.4)}$
w/o $\mathcal{R}_{bn}$	$51.2_{(6.1)}$	$33.1_{(7.1)}$	$24.1_{(5.2)}$
w/o collab	$64.4_{(0.5)}$	$58.4_{(1.4)}$	$30.8_{(3.2)}$

Table 5: Ablation of components in Co-Dream on CIFAR10

Contribution of loss components $\mathcal{R}_{bn}$ and $\mathcal{R}_{adv}$ in knowledge extraction

We further explore the impacts of various components of loss function in data generation in Eq. 3. Through leave-one-out testing, we present results by excluding $\mathcal{R}_{bn}$ (w/o $\mathcal{R}_{bn}$) and excluding $\mathcal{R}_{adv}$ (w/o $\mathcal{R}_{adv}$). Table 5 shows removing either component influences the accuracy of the overall model, illustrating the impact of each part of the loss function plays an important role in generating good quality *dreams*.

Membership Inference Attack Evaluation

We evaluate whether models trained over dreams leak less or more information than their federated or centralized counterpart. We evaluate information leakage in the trained models using the LiRA membership inference attack (Carlini et al. 2022). We train all models to achieve a similar classification accuracy to ensure a fair comparison of their privacy-preserving capabilities. To simulate the LiRA attack, we perform a single query per datapoint to calculate the ϕ scores, utilizing one target model and ten shadow models. Since the attack objective is binary classification, we measure attack success with balanced accuracy, AUROC, and true-positive rates at the 0.1% and 0.001% false positive rates, with lower scores indicating lesser privacy leakage. We plot the results in Table 4. CoDream-fast achieves a better performance on all the metrics for the LiRA attack. While not a worst-case theoretical improvement, the results hint that CoDream-fast can be a more viable alternate for sharing knowledge.

Visual representation of dreams

Figure 6 visualizes the *dreams* generated by CoDream-fast on CIFAR10. While not visually similar to the original training data, these *dreams* effectively encapsulate collaborative knowledge. Unlike traditional model inversion algorithms, the objective for these dreams is to enable decentralized knowledge transfer and not reconstruct the raw data. Thus, models trained on *dreams* perform well despite their visual differences from the underlying distribution. We visualize more images of Co-Dream across different data distributions in the Supplementary.

Figure 6: Visualization of *dreams* generated on CIFAR10

Conclusion

We introduce Co-Dream, a model-agnostic learning framework that leverages a knowledge extraction algorithm by performing gradient descent in the input space. We view this approach as a complementary technique to FedAvg, which performs gradient descent over model parameters. Through comprehensive evaluations and ablation studies, we validate the effectiveness of our proposed method.

While we restrict the scope of our work on collaborative learning applications, we believe Co-Dream can serve as a building block for several other interesting problems such as identifying similarity between models without relying on proxy datasets. We believe further research is warranted in client dropouts, stragglers, formal privacy guarantees, bias to explore the effectiveness of Co-Dream under those constraints.

References

Afonin, A.; and Karimireddy, S. P. 2022. Towards Model Agnostic Federated Learning Using Knowledge Distillation. In *International Conference on Learning Representations*.

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

Buciluǎ, C.; Caruana, R.; and Niculescu-Mizil, A. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, 535–541.

Carlini, N.; Chien, S.; Nasr, M.; Song, S.; Terzis, A.; and Tramer, F. 2022. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, 1897–1914. IEEE.

Chang, H.; Shejwalkar, V.; Shokri, R.; and Houmansadr, A. 2019. Cronus: Robust and heterogeneous collaborative learning with black-box knowledge transfer. *arXiv preprint arXiv:1912.11279*.

Chen, H.-Y.; and Chao, W.-L. 2021. Fed{BE}: Making Bayesian Model Ensemble Applicable to Federated Learning. In *International Conference on Learning Representations*.

Fang, G.; Mo, K.; Wang, X.; Song, J.; Bei, S.; Zhang, H.; and Song, M. 2022. Up to 100x faster data-free knowledge distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 6597–6604.

Goetz, J.; and Tewari, A. 2020. Federated Learning via Synthetic Data. arXiv:2008.04489.

Haim, N.; Vardi, G.; Yehudai, G.; Shamir, O.; and Irani, M. 2022. Reconstructing training data from trained neural networks. *Advances in Neural Information Processing Systems*, 35: 22911–22924.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Deep residual learning for image recognition. arXiv 2015. *arXiv preprint arXiv:1512.03385*, 14.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hinton, G.; Vinyals, O.; Dean, J.; et al. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7).

Hitaj, B.; Ateniese, G.; and Perez-Cruz, F. 2017. Deep models under the GAN: information leakage from collaborative deep learning. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, 603–618.

Karimireddy, S. P.; Kale, S.; Mohri, M.; Reddi, S.; Stich, S.; and Suresh, A. T. 2020. Scaffold: Stochastic controlled averaging for federated learning. In *International conference on machine learning*, 5132–5143. PMLR.

Konečnỳ, J.; McMahan, H. B.; Ramage, D.; and Richtárik, P. 2016. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324.

Li, Q.; He, B.; and Song, D. 2021. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 10713–10722.

Li, T.; Sahu, A. K.; Zaheer, M.; Sanjabi, M.; Talwalkar, A.; and Smith, V. 2020. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2: 429–450.

Lin, T.; Kong, L.; Stich, S. U.; and Jaggi, M. 2020. Ensemble distillation for robust model fusion in federated learning. *Advances in Neural Information Processing Systems*, 33: 2351–2363.

McMahan, H. B.; Moore, E.; Ramage, D.; Hampson, S.; and y Arcas, B. A. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*.

McMahan, H. B.; Moore, E.; Ramage, D.; Hampson, S.; and y Arcas, B. A. 2023. Communication-Efficient Learning of Deep Networks from Decentralized Data. arXiv:1602.05629.

Mora, A.; Tenison, I.; Bellavista, P.; and Rish, I. 2022. Knowledge Distillation for Federated Learning: a Practical Guide. *arXiv preprint arXiv:2211.04742*.

Mordvintsev, A.; Olah, C.; and Tyka, M. 2015. Inceptionism: Going deeper into neural networks.

Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; and Ng, A. Y. 2011. Reading digits in natural images with unsupervised feature learning.

Rasouli, M.; Sun, T.; and Rajagopal, R. 2020. FedGAN: Federated Generative Adversarial Networks for Distributed Data. arXiv:2006.07228.

Simonyan, K.; and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Song, R.; Liu, D.; Chen, D. Z.; Festag, A.; Trinitis, C.; Schulz, M.; and Knoll, A. 2022. Federated learning via decentralized dataset distillation in resource-constrained edge environments. *arXiv preprint arXiv:2208.11311*.

Xin, B.; Yang, W.; Geng, Y.; Chen, S.; Wang, S.; and Huang, L. 2020. Private FL-GAN: Differential Privacy Synthetic Data Generation Based on Federated Learning. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2927–2931.

Yin, H.; Molchanov, P.; Alvarez, J. M.; Li, Z.; Mallya, A.; Hoiem, D.; Jha, N. K.; and Kautz, J. 2020. Dreaming to distill: Data-free knowledge transfer via deepinversion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8715–8724.

Zagoruyko, S.; and Komodakis, N. 2016. Wide residual networks. *arXiv preprint arXiv:1605.07146*.

Zhang, J.; Chen, C.; Li, B.; Lyu, L.; Wu, S.; Ding, S.; Shen, C.; and Wu, C. 2022a. Dense: Data-free one-shot federated learning. *Advances in Neural Information Processing Systems*, 35: 21414–21428.

Zhang, L.; Shen, L.; Ding, L.; Tao, D.; and Duan, L.-Y. 2022b. Fine-tuning global model via data-free knowledge distillation for non-iid federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10174–10183.

Zhu, Z.; Hong, J.; and Zhou, J. 2021. Data-free knowledge distillation for heterogeneous federated learning. In *International Conference on Machine Learning*, 12878–12889. PMLR.