Co-Dream: Collaborative data synthesis with decentralized models

Anonymous Authors[1]

Abstract

We present a framework for distributed optimization that addresses the decentralized and siloed nature of data in real world. Existing works in Federated Learning address it by learning a centralized model from decentralized data. Our framework *Co-Dream* instead focuses on learning representation of data itself. By starting with random data and jointly synthesizing samples from distributed clients, we aim to create proxies that represent the global data distribution. Importantly, this collaborative synthesis is achieved using only local models, ensuring privacy comparable to sharing the model itself. The collaboration among clients is facilitated through federated optimization in the data space, leveraging shared input gradients based on local loss. This collaborative data synthesis offers various benefits over collaborative model learning, including lower dimensionality, parameter-independent communication, and adaptive optimization. We empirically validate the effectiveness of our framework and compare its performance with traditional federated learning approaches through benchmarking experiments.

1. Introduction

In the current era of big data, data is distributed among silos owned by different users or organizations, making it difficult to collaboratively train machine learning models on large datasets. Centralizing data is not always feasible due to regulatory and privacy concerns in domains such as healthcare, finance, and mobility. Federated Learning (FL) solves this problem by centrally aggregating clients' models instead of data. But if we could simply generate samples that represent characteristics of the data distribution while still maintaining privacy, then we would eliminate the need to aggregate the client models (and potentially eliminate the need for FL). Sharing samples offers much higher flexibility for training models and supports arbitrary model architectures(unlike FL) and tasks.

We design a framework for collaboratively synthesizing a proxy of the siloed data distributions, called *dreams*, without centralizing data or client models. Just like `FedAvg` (McMahan et al., 2017), `Co-Dream` also exhibits two-folds of privacy: (1) clients share *dreams'* updates instead of raw data, (2) clients can securely aggregate their *dreams* using existing cryptographic techniques without revealing their individual updates to the server.

Our proposed technique, `Co-Dream`, collaboratively optimizes *dreams* to aggregate knowledge from the client's local models. Importantly, our approach allows different model architectures to be used for each client. By sharing *dreams* in the data space rather than the model parameters, our method is model-agnostic and scalable to large models. The key idea is, to begin with randomly initialized samples and apply federated optimization on these samples for extracting knowledge from the client's local models trained on their original dataset. Our framework represents the first solution that combines both the privacy advantages of FL with the flexibility of model heterogeneity. Furthermore, communication is not dependent on the model parameter size, thereby alleviating scalability concerns.

By performing extensive experiments and analysis in Sec 3, we establish the feasibility of `Co-Dream` as a way for clients to collaboratively synthesize samples. Our results show that collaboratively optimized dreams give a higher performance (up to $\sim$ 20% accuracy improvement on CIFAR-10) and have lower sample complexity compared to independently optimized dreams. We believe that our proposed approach has the potential to rethink the way we approach data decentralization.

In summary, our contributions are summarized as 1) A framework for collaborative data synthesis by federated optimization in the data space. 2) Formulate the learning of a global model as a knowledge acquisition problem and design a personalized distillation procedure for *adaptively* extracting knowledge from the clients. 3) Empirical validation of our framework by benchmarking with existing algorithms and ablation studies across various design choices.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

2. Co-Dream

`Co-Dream` comprises of two key stages: **knowledge extraction** and **knowledge aggregation**.

In the knowledge extraction stage, we aim to obtain useful representations of data, called *dreams*, from each client that can be used for training a model that reaches similar performances as the client model. The clients begin with a few warmup rounds to pre-train their local model and then jointly optimize random noise images. To facilitate knowledge aggregation, we leverage the linearity of gradients to exploit the fact that our optimization process is gradient-based. This results in an optimization scheme similar to distributed SGD. However, unlike `FedAvg` and distributed-SGD, our aggregation step occurs in the data space. This makes our approach model-agnostic and compatible with FL setups that involve heterogeneous client architectures.

2.1. Local dreaming to extract knowledge from models

While DeepDream (Mordvintsev et al., 2015) and DeepInversion (Yin et al., 2020) (see Appendix B) both enable data-free knowledge extraction, they are not directly applicable to FL because the teacher models are continuously evolving and the student learns from multiple teachers as well as its own data. A direct consequence of this non-stationarity is that it is unclear how the label y should be chosen in Eq 5. In DeepInversion, the teacher uniformly samples y from its own label distribution because the teacher has the full dataset. However, we cannot assume this in FL because data is distributed across multiple clients with heterogeneous data distributions. Additionally, any given client should synthesize only those *dreams* over which they are highly confident.

The main issue with directly applying Eq 5 is how 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 5 so that the teacher synthesizes *dreams* without any classification loss by instead minimizing the entropy (denoted by $\mathcal{H}$) on the output distribution. Formally, we optimize the following objective for synthesizing *dreams*:

$$\min_{\hat{x}} \left\{ \tilde{\ell}(\hat{x},\, \theta) \;:=\; \mathcal{H}\left(f_\theta(\hat{x})\right) \;+\; \mathcal{R}(\hat{x}) \right\}. \tag{1}$$

The teacher starts with a batch of representations sampled from a standard Gaussian ($\hat{x} = \mathcal{N}(0,1)$), and these *dreams* are optimized using Eq 1. In contrast to Eq 5, we do not restrict $\hat{x}$ to the data space but allow it to be the representation at any layer. In Sec 3, we show that for certain experiments, sharing representations from the penultimate layer performs equally as well as sharing in the data space. Note that, unlike generative models, the only goal of optimizing *dreams* is to enable KD rather than maximize the likelihood of the data. Therefore, *dreams* do not need to appear like real images. We show several visual results of *dreams* in Supplementary.

2.2. Collaborative dreaming for knowledge aggregation

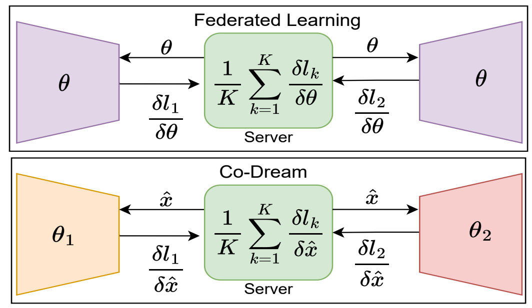

Figure 1. Comparing aggregation framework in FL & `Co-Dream`.

If we had assumed that server could be trusted, then the best way to aggregate knowledge would be to pool data from all users at the server and train a teacher model on this aggregated data by applying the knowledge extraction objective described in Eq 1. This can be written as

$$\min_{\hat{x}} \tilde{\ell}\left(\hat{x}, \theta^*\right) \quad \text{s.t.} \;\; \theta^* = \arg\min_{\theta} \underset{\mathcal{D}_k \sim p(\mathcal{D})}{\mathbb{E}} \left[\ell\left(\mathcal{D}_k,\; \theta\right)\right]. \tag{2}$$

However, we cannot obtain θ^* without centralized data. Furthermore, estimating θ^* using `FedAvg` will not generally be model-agnostic. Therefore, we collaboratively optimize *dreams* by taking the expectation over every client's own loss with respect to the same $\hat{x}$:

$$\min_{\hat{x}} \underset{\mathcal{D}_k \sim p(\mathcal{D})}{\mathbb{E}} \left[\phi(\hat{x},\; \mathcal{D}_k) := \tilde{\ell}\left(\hat{x},\; \arg\min_{\theta} \ell\left(\theta,\; \mathcal{D}_k\right)\right)\right]. \tag{3}$$

Eq 3 can be optimized for distributed data even if not exactly equivalent to optimizing Eq 2. The empirical risk (Eq 3) can be minimized by computing the local loss at each client. Therefore, the update rule for $\hat{x}$ can be written as

$$\hat{x} \leftarrow \hat{x} - \nabla_{\hat{x}} \sum_{\mathcal{D}_k \in \mathcal{D}} \frac{1}{|\mathcal{D}_k|} \phi\left(\hat{x},\; \mathcal{D}_k\right).$$

While no single party can compute this gradient because models are decentralized due to the linearity of gradients, we can write the above equation as

$$\hat{x} \leftarrow \hat{x} - \sum_{\mathcal{D}_k \in \mathcal{D}} \frac{1}{|\mathcal{D}_k|} \nabla_{\hat{x}} \phi\left(\hat{x},\; \mathcal{D}_k\right) \tag{4}$$

The clients compute gradients locally with respect to the input and share them with the server, which aggregates the gradients and returns the updated input to the clients. This formulation is the same as the distributed-SGD formulation, but the optimization is performed in the data space instead of the model parameter space. Our framework is compatible with existing cryptographic aggregation techniques, as the aggregation step is linear and only reveals the final aggregated output without exposing individual client gradients. Collaboratively optimizing representations, known as *dreams* in our approach, is a novel concept that has not been explored before. Our experiments (Sec 3) demonstrate that dreams obtained through this approach capture knowledge from all clients and outperform dreams synthesized by independent clients regarding the server's performance.

Similar to `FedAvg`, we perform multiple local steps before synchronization to enhance communication efficiency as follows: At the start of every round r, each user k starts with the same parameter $\hat{x}^r_{k,0} := \hat{x}^{r-1}$ and update its local parameters for M steps, i.e., $\hat{x}^r_{k,m} = \hat{x}^r_{k,m-1} - \eta_l \cdot g_k(\hat{x}^r_{k,m-1})$. Here, $g_k(x) := \nabla_x(\tilde{\ell}(x, \mathcal{D}_k))$ is gradient function for the client k and η_l is the local learning rate for the clients. Upon the completion of the local optimization, each client sends its local updates $\hat{x}^r_{k,M} - \hat{x}^r$ to the server. The local updates are commonly referred to as pseudo-gradients and are aggregated by the server as follows: $\hat{x}^{r+1} = \hat{x}^r + \eta_g \sum_{k \leq K} \frac{1}{|\mathcal{D}_k|} \left(\hat{x}^r_{k,M} - \hat{x}^r\right)$. Note that the choice of parameters such as local updates M, 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 of the optimization problem.

2.3. Analysis of `Co-Dream`

Communication Comparison. To understand the communication efficiency of this procedure, recall the notation: d is the dimension of the inputs, n is the number of samples generated, and R is the number of aggregation rounds. Since our approach is model agnostic, the total communication is $d \times n\ R$. In FedAvg and its recent variants, the communication is usually of the order $|\theta| \times R$. For heavily parameterized models, the communication is $|\theta| \ll d \times n$. We comprehensively evaluate the role of n in the performance of our system.

Benefits of `Co-Dream` are inherited from the usage of KD, along with additional advantages arising from our specific optimization technique.

(1) Lower dimensionality. `Co-Dream` communicates input gradients ($\nabla_{\hat{x}}$) instead of model gradients (∇_θ), which can be advantageous for robust averaging and privacy mechanisms due to their dependence on the dimensionality of samples. Moreover, the data dimensionality remains constant even if the model increases in depth and width, which makes this approach suitable for large FL models.

(2) Optimization in the data space. First, being model agnostic, `Co-Dream` allows for collaboration among clients with different model architectures. Second, the shared inputs are semi-interpretable, enabling better analysis of the learned knowledge. Third, clients can collaborate without revealing their proprietary ML models, enhancing privacy. Fourth, sharing knowledge in the data space enables adaptive optimization, such as synthesizing adversarially robust samples or class-conditional samples. Finally, the linearity of the aggregation algorithm makes our approach compatible with secure averaging (Bonawitz et al., 2017).

Limitations of `Co-Dream` are mainly due to the additional layer of optimization for synthesizing *dreams*, i.e. the clients now need to optimize ML models locally and optimize representations collaboratively. Therefore, the following limitations arise:

(1) Additional computation on the client device: While the number of parameters on the client device remains unchanged, as gradients are applied in the data space, the client device has an additional computation burden. This additional computation can be offloaded to the server if secure aggregation is not required.

(2) Sample inefficient - Experimentally, we find that many samples are required to effectively transfer knowledge among clients due to the redundancy of features in independently generated $\hat{x}$. We believe the problem can be circumvented by not using the same initialization, and we show promising results in Sec 3.

3. Experiments

Setup: We evaluate the effectiveness of `Co-Dream` at each of the two stages: knowledge extraction 2.1, and knowledge aggregation 2.2, on three image classification datasets (MNIST (LeCun et al., 1998), CIFAR10 (Krizhevsky et al., 2009), and PathMNIST (Yang et al., 2023)). We also analyze several aspects of `Co-Dream` with ablation experiments. For quantitative evaluation, we train a student model from scratch on only dreams and treat the model's accuracy as a proxy for the quality of the synthesized dreams.

Validating knowledge-extraction in low data settings. We evaluate whether the knowledge-extraction approach (Sec 2.1) allows for the effective transfer of knowledge from teacher to student. We first train a teacher model on different datasets, synthesize samples with our knowledge-extraction approach, and then train a student on the extracted knowledge. 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 and evaluate how this affects student performance. Results in Fig 2 show that the teacher-student performance gap does not consistently degrade 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.

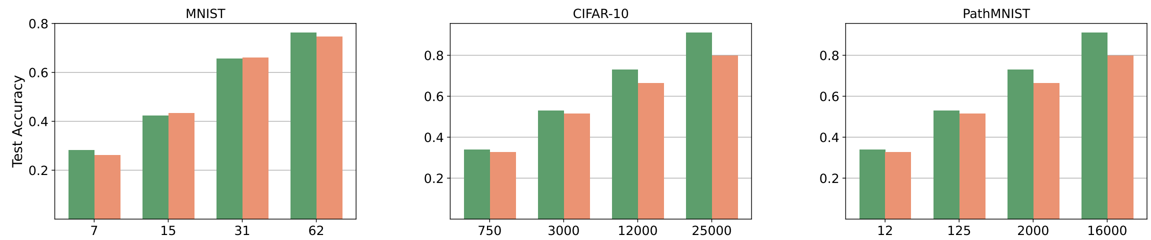

Figure 2. **Effectiveness of knowledge transfer from teacher to student.** We vary the size of the training dataset (on the x-axis) of the teacher (in green) and compare its accuracy with the student (in orange). We find that the student can perform similarly to the teacher even though the perceptual quality of the teacher samples is poor.

Validating collaborative optimization. We evaluate how distributing data across multiple clients affects the quality of the extracted knowledge. We keep a fixed dataset size (128 samples for MNIST, 24k samples for CIFAR10, and 24k samples for PathMNIST) and distribute these samples evenly among the clients. We evaluate the effectiveness of collaborative dreaming by varying the number of clients $K = \{1, 2, 4, 6, 8, 12, 24\}$ and training a student model from scratch on the extracted knowledge. While a performance drop is expected as the number of clients increases, we observe in Fig 3 that the performance drop is sublinear and quite compatible with cross-device FL settings.

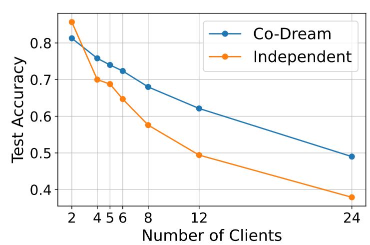

Figure 3. Comparison by varying the number of clients. The performance gap widens between `Co-Dream` and independent optimization as we increase the number of clients.

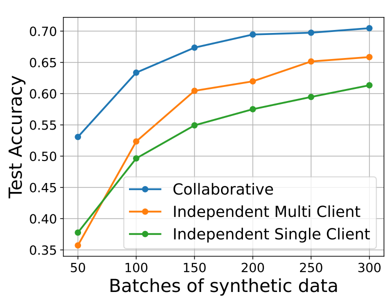

Figure 4. Comparison between collaborative and independent optimization.

Collaborative optimization versus aggregate knowledge extraction. We compare the performance of clients independently extracting knowledge (using Eq 1) against combining their individual datasets on the server. We plot the accuracy of the server model optimized from scratch and find the collaboratively optimized samples are significantly more sample efficient for training a student model than the independently optimized samples. For instance, we train a randomly initialized student model with 50 batches of synthesized *dreams* and show that collaboratively optimized *dreams* get $\sim 20\%$ higher test accuracy on average than independently optimized *dreams*.

Improving local computation cost. As noted in Sec 2.3, the key limitation of our technique is the additional computation cost incurred in synthesizing *dreams*. To address this, we identify some assumptions that help alleviate these computational costs. If the clients use the same model, we can synthesize the *dreams* in the activation space instead of data, resulting in faster optimization as only the gradients need to be backpropagated for the last few layers. Additionally, if we assume that a client has additional memory to support a generative model, then *dreams* can be synthesized by initializing them as the output of the generative model instead of random noise, significantly reducing the communication cost.

4. Conclusion

We introduced `Co-Dream`, a collaborative data synthesis approach where clients jointly optimize for an accuracy model-agnostic federated learning framework that leverages a knowledge extraction algorithm for gradient descent in the input space. We view this approach as a complementary technique to `FedAvg`, which performs gradient descent over model parameters. Our contributions were validated through comprehensive evaluations and ablation studies. Future work includes more empirical evaluation in data heterogeneous scenarios and theoretical analysis of federated optimization in data space. New privacy mechanisms catered for `Co-Dream` that have improved privacy-utility trade-off is another promising future avenue.

References

Afonin, A. and Karimireddy, S. P. Towards model agnostic federated learning using knowledge distillation. In *International Conference on Learning Representations*, 2022.

Allen-Zhu, Z. and Li, Y. Towards understanding ensemble, knowledge distillation and self-distillation in deep learning. *arXiv preprint arXiv:2012.09816*, 2020.

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

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

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

Chen, H.-Y. and Chao, W.-L. Fed{be}: Making bayesian model ensemble applicable to federated learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=dgtpE6gKjHn.

Goetz, J. and Tewari, A. Federated learning via synthetic data, 2020.

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

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

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

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

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

McMahan, H. B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017. URL http://arxiv.org/abs/1602.05629.

McMahan, H. B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data, 2023.

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

Rasouli, M., Sun, T., and Rajagopal, R. Fedgan: Federated generative adversarial networks for distributed data, 2020.

Reddi, S., Charles, Z., Zaheer, M., Garrett, Z., Rush, K., Konečnỳ, J., Kumar, S., and McMahan, H. B. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.

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

Xin, B., Yang, W., Geng, Y., Chen, S., Wang, S., and Huang, L. 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)*, pp. 2927–2931, 2020. doi: 10.1109/ICASSP40776.2020.9054559.

Yang, J., Shi, R., Wei, D., Liu, Z., Zhao, L., Ke, B., Pfister, H., and Ni, B. Medmnist v2-a large-scale lightweight benchmark for 2d and 3d biomedical image classification. *Scientific Data*, 10(1):41, 2023.

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

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

Zhang, L., Shen, L., Ding, L., Tao, D., and Duan, L.-Y. 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*, pp. 10174–10183, 2022b.

Zhang, Z. and Sabuncu, M. Self-distillation as instance-specific label smoothing. *Advances in Neural Information Processing Systems*, 33:2184–2195, 2020.

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

	Overview	Resources			Flexibility / Utility		Security	
	What is shared?	Comm.	Comp.	Memory	Heterogeneous models	Heterogeneous tasks	Compatible with Secure Agg.	Levels of Privacy
Federated Learning	Predictive Model[1]	Baseline	Baseline	Baseline	No	Yes	Yes	1
Federated Generative modeling	Generative Model[1]	High	High	High	No	Yes	Yes	1
Synthetic Data Sharing	Synthetic Data[2]	Low	High	High	Yes	Yes	No	2
Data-Free KD	Predictive Model[1]	High	High	High	Yes	No	No	1
Co-Dream	Dreams[2]	Same	High	Same	Yes	Yes	Yes	2

Figure 5. **Landscape of FL techniques.** By levels of privacy, we mean how distant the shared updates are from the raw data. Sharing synthetic data[2] and *dreams*[2] are two levels of indirection away from the raw data than sharing models[1].

A. Related Work

The problem of collaborative data synthesis has been previously explored using generative modeling and federated learning techniques. Figure 5 compares existing decentralization solutions regarding shared resources, utility, and privacy. We refer the reader to Supplementary for a more detailed discussion of existing works.

Generative modeling techniques either pool locally generated data on the server (Song et al., 2022; Goetz & Tewari, 2020) or use `FedAvg` with generative models (Rasouli et al., 2020; Xin et al., 2020). `FedAvg` over generative models lead to the same problem `FedAvg` over predictive models. While we share the idea of generative modeling of data, we do not expose individual clients' updates or models directly to the server.

Knowledge Distillation in Federated Learning 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 & Karimireddy, 2022; Chen & 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 et al., 2021). These data-free KD approaches are not amenable to secure aggregation and must use the same architecture for the generative model.

However, all these existing approaches lack active client collaboration in the knowledge synthesis process. Clients share their local models 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. Therefore, we introduce `Co-Dream`, which enables clients to synthesize dreams collaboratively while remaining compatible with secure aggregation techniques.

B. Preliminaries

Federated Learning aims to minimize 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 (Konečný et al., 2016; McMahan et al., 2023). In the absence of access to the true distribution, FL aims to optimize the empirical risk instead $\min_\theta \sum_{k\in K} \frac{1}{|\mathcal{D}_k|}\ell(\mathcal{D}_k,\theta)$. Here, each $\mathcal{D}_k$ is owned by client k in the federation and $\mathcal{D}$ is assumed to be partitioned across K clients $\mathcal{D} = \cup_{k\in K}\mathcal{D}_k$. The optimization proceeds with the server broadcasting θ^t to each user k that locally optimizes $\theta_k^{t+1} = \arg\min_{\theta^t} \ell(\mathcal{D}_k, \theta^t)$ for r rounds and sends local updates either in the form of θ_k^{t+1} or $\theta_k^{t+1} - \theta_k^t$ (*pseudo-gradient*) to the server to aggregate local updates and send the aggregated weights back to the clients.

[1]Aggregation of local updates occurs in the model parameters space
[2]Aggregation of local updates occurs in the data space

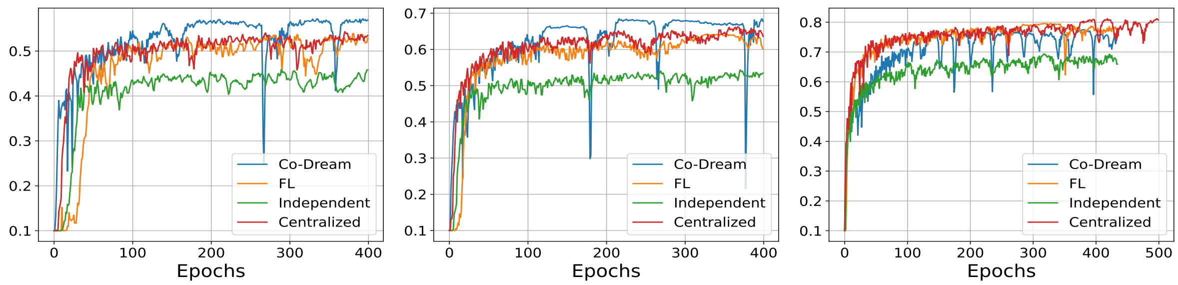

Figure 6. Comparing test-accuracy (on y-axis) for CIFAR-10 between `FedAvg` and `Co-Dream` for different samples per client ratios. We include centralized and independent baselines for reference.

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 (Buciluă et al., 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 et al., 2015) first showed that features learned DL 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

$$\min_{\hat{x}} \mathsf{CE}\left(f_\theta(\hat{x}),\ y\right)\ +\ \mathcal{R}(\hat{x}), \tag{5}$$

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:

$$\mathcal{R}_{bn}(\hat{x}) = \sum_l \left\|\mu_l(\hat{x}) - \mathbb{E}_{\mathcal{D}}\left[\mu_l(x)\right]\right\|_2\ +\ \left\|\sigma_l(\hat{x}) - \mathbb{E}_{\mathcal{D}}\left[\sigma_l(x)\right]\right\|_2, \tag{6}$$

where $x \in \mathcal{X}$ are the original samples from a dataset and $\mu_l(\cdot)$ and $\sigma_l(\cdot)$ are mean and variance for the l'th layer's feature maps for a given batch. The value $\mathbb{E}_{\mathcal{D}}[\mu_l(x)]$ can be approximated using the running mean and variance of the batch normalization layers stored in a model.

C. Additional Experiments

Comparison with FL We evaluate the performance on the CIFAR10 dataset when samples are IID among clients. For baselines, we compare against `FedAvg`, and include `Independent`, and `Centralized` training baseline for reference. In the `Centralized` baseline, all the data from the clients are aggregated in a single place. In the case of `Independent`, we train models only on the client's local dataset. We also experiment with varying the number of samples per client. We plot the results across communication rounds in Fig C and report the maximum accuracy across multiple rounds in Table 1. We find that `Co-Dream` consistently performs closer to the centralized baseline and outperforms centralized in two out of the three cases. We posit that the reason `Co-Dream` outperforms the centralized baseline is due to the self-distillation phenomenon (Allen-Zhu & Li, 2020; Zhang & Sabuncu, 2020).

Table 1. Performance on IID clients for CIFAR-10

#samples	1k/client	2k/client	3k/client
`Centralized`	0.543	0.6627	0.81
`Independent`	0.458	0.543	0.694
`FedAvg`	0.538	0.644	**0.796**
`Co-Dream`	**0.571**	**0.689**	0.775

Federated averaging versus distributed optimization. Similar to `FedAvg`, our approach reduces client communication by increasing the number of local steps performed on the client device. Therefore, we quantify the tradeoff between the number of local steps and the reduction in the quality of the co-dreams. We note that our knowledge extraction approach is sensitive to the optimizer and usually performs better with Adam (Kingma & Ba, 2014) over SGD.

This presents a unique challenge when performing multiple steps of local optimization locally as the server can not perform adaptive optimization anymore. Therefore, we utilize the same approach as adaptive federated optimization (Reddi et al., 2020) that treats the server aggregation step as an optimization problem and replaces the simple averaging (i.e. FedAvg) with adaptive averaging with learnable parameters on the server.

We compare three methods of optimization: 1) *DistAdam* where the clients share gradients at every step and the server applies Adam optimizer on the aggregated gradients, 2) *FedAvg* where clients apply Adam optimizer locally for m steps and the server averages the *pseudo-gradients* as described in Eq 4, and 3) *FedAdam* where clients apply Adam optimizer locally for m steps and the server performs adaptive optimization on the aggregated *pseudo-gradients* based on the formulation by (Reddi et al., 2020).

Table 2. Comparison of different optimization techniques for `Co-Dream`. m refers to the total number of communication rounds.

#Optimization	m	MNIST	CIFAR10
DistAdam	2k	0.763	0.644
FedAvg	400	0.1826	0.5919
DistAdam	400	0.7978	0.5949
FedAdam	400	0.7831	0.6439

We show qualitative results in the Supplementary and quantitative difference in Table 2. We find that the naive *FedAvg* approach reduces the student performance even with a minor increase in the number of local computation steps; however, when we apply *FedAdam* (Reddi et al., 2020), we see similar performance as *DistAdam* with reduced global steps.