Leveraging Foundation Models to Improve Lightweight Clients in Federated Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Federated Learning (FL) is a distributed training paradigm that enables clients scattered across the world to cooperatively learn a global model without divulging confidential data. However, FL faces a significant challenge in the form of heterogeneous data distributions among clients, which leads to a reduction in performance and robustness. A recent approach to mitigating the impact of heterogeneous data distributions is through the use of foundation models, which offer better performance at the cost of larger computational overheads and slower inference speeds. We introduce *foundation model distillation* to assist in the federated training of lightweight client models and increase their performance under heterogeneous data settings while keeping inference costs low. Our results show improvement in the global model performance on a balanced testing set, which contains rarely observed samples, even under extreme non-IID client data distributions. We conduct a thorough evaluation of our framework with different foundation model backbones on CIFAR10, with varying degrees of heterogeneous data distributions ranging from class-specific data partitions across clients to dirichlet data sampling, parameterized by values between 0.01 and 1.0.

1 Introduction

Federated learning (FL) is a decentralized training paradigm in machine learning [McMahan et al., 2017] that trains one global model across multiple clients while preserving the privacy of client data. A typical FL framework consists of a central server coordinating global model training by periodically aggregating clients' local models that are trained with locally-stored data. Similar to a variety of distributed learning approaches, one of the major challenges for FL is that locally-stored client data are heterogeneous, which can result from uneven distributions or unbalanced patterns [Li et al., 2020a]. This results in client-drift, commonly-seen accuracy drops, and non-convergence [Zhao et al., 2018, Karimireddy et al., 2020, Hsieh et al., 2020, Li et al., 2020a]. In addition, many FL clients in real-world deployments are edge devices which have strict limitations on inference speeds and compute. This, in turn, restricts clients to using small-scale models for inference.

To maintain model performance under heterogeneous data distributions, foundation models [Bommasani et al., 2021] present a potential solution. Their widely known benefits, such as their comprehensive knowledge, transferable representations across a broad range of downstream tasks [Radford et al., 2021, Wang et al., 2022], and strong robustness to distribution shifts [Ma et al., 2021] make them a strong candidate to mitigate the effects of heterogeneous distributions. With this in mind, multiple recent methods explore fine-tuning foundation models under federated settings [Qu et al., 2022, Chen et al., 2022a, Guo et al., 2023a, Chen et al., 2022b, Su et al., 2022, Guo et al., 2023b]. Among these, Chen et al. [2022b] have shown that under extreme non-IID conditions, federated fine-tuning of foundation models forces a performance worse than training on local data only. In addition, these methods incur a relatively large increase in inference time and compute by directly using foundation models instead of small-scale alternatives like EfficientNet Tan and Le [2019] or MobileNet Sandler et al. [2018] for inference.

In order to effectively leverage the performance of foundation models under non-IID data distributions while using small-scale backbones for faster inference, we propose an approach to *distill* knowledge from foundation models into the small-scale client models (proxy models). During training, foundation models are not directly fine-tuned, but rather are leveraged to update each client's proxy model; then proxy model updates are shared with and aggregated by the server model. At inference, only the proxy models are used. Thus, the proxy models offer low latency inference while knowledge from the foundation models helps reduce the bias and diversify the knowledge of the proxy models, especially under heterogeneous local data distributions. In addition, our proposed method is agnostic to the number and size of foundation models available to each client. This offers the option of personalization for each client, which can select the appropriate foundation model(s) based on the amount of storage and compute available, as well as local data characteristics; this is particularly beneficial when some clients have much more/less data than others. We use the concept of personalization to highlight important directions for future work.

Overall, our main contributions are as follows:

- This is the first approach to leverage foundation models in FL via distillation to help improve the performance and robustness achievable in small-scale client models (*e.g.* relative increase of 9.22% for EfficientNetB0, 8.69% for ResNet18 and 24.60% for MobileNetV2).
- Within the space of low latency models, we provide a federated learning solution robust to various heterogeneous client data. Our approach outperforms prior art across a variety of client data distributions, from IID to various parametrized dirichlet distributions and class-specific partitions.
- We explore the impact of leveraging representations from fine-tuned foundation models on local data versus pre-trained foundation models. Our results show that under IID data distributions, an initial step of fine-tuning foundation models offers no benefit over 0-shot foundation models, and significantly hinders accuracy as data heterogeneity increases, suggesting that directly fine-tuning foundation models leads to biased representations.
- Our framework is also the first to allow clients the flexibility in choosing their locally-stored foundation models (personalization) according to the scale of compute and data available. We study the impact of variable foundation model backbones and highlight the importance of combining disparate feature representations correctly.

2 Related Work

Federated Learning with Heterogenous Client Data Federated learning is a distributed machine learning scheme which enables multiple clients to train a shared model while keeping their data private. Typically a central server federates the training procedure by periodically aggregating model updates from clients [McMahan et al., 2017]. Frequently, client data can have non-identical distributions which causes naive aggregation methods to not be able to guarantee global model convergence to a local minimum [Zhao et al., 2018, Li et al., 2020b, Hsieh et al., 2020, Li et al., 2020a]. To tackle this challenge, FL-algorithms such as FedProx [Li et al., 2020a] add a proximal term to the local training objective to protect models in each client from over-fitting to the local data distribution; other approaches such as regularization [T Dinh et al., 2020], model mixture [Deng et al., 2020, Mansour et al., 2020, Hanzely and Richtárik, 2020], clustering clients [Sattler et al., 2020, Cho et al., 2021], multi-task learning [Smith et al., 2017], and meta-learning [Fallah et al., 2020] have been introduced to stabilize the trained models. In this work, we tackle the issue of heterogeneous data distributions by distilling knowledge from foundation models to proxy models, to help mitigate this issue without the need for additional data.

Foundation Models in FL The past few years have witnessed the rapid development of foundation models with the integration of language [Radford et al., 2018, Devlin et al., 2018, Radford et al., 2021], vision [Bao et al., 2021], and audio modalities [Tang et al., 2023] across many tasks. In FL, foundation models have been used to improve the robustness of clients to distribution shifts and heterogeneous data distributions [Qu et al., 2022] or the overall performance of the system [Chen et al., 2022b, Guo et al., 2023a, Zhao et al., 2023, Lu et al., 2023, Guo et al., 2023b]. However, existing works do not fully address the increase in computational overhead nor inference time that follow the use of foundation models. In addition, even compressed foundation models [Sanh et al., 2019, Wu et al., 2023] do not fully match the latency requirements of clients, which hinders their deployment in real-world settings. Therefore, we propose the use of small-scale proxy models and distillation to leverage the performance of foundation models while keeping inference costs low.

Distillation Knowledge distillation is a teaching technique that transfers valuable insights and generalization capabilities from a trained teacher model to a student model [Hinton et al., 2015, Anil et al., 2018, Zhang et al., 2018, 2021]. Within the domain of FL, Lin et al. [2020] explore adaptable aggregation methods with ensemble distillation at the server, while Sattler et al. [2021] use an auxiliary dataset to weight and ensemble local models from each client. FedDistill [Seo et al., 2022] extracts statistics related to the logit-vector from different client models and shares them with the remaining clients to help with distillation. Zhu et al. [2021] present a data-free knowledge distillation approach by training a generative model at the server, using information from clients. They proceed to use the generative model to create synthetic data which is used to train client models. Cho et al. [2021] propose a co-distillation-based personalized FL method to allow cross-architecture training. In our approach, we study the impact of knowledge distillation Hinton et al. [2015] on the performance of small-scale client models without the use of excessive data, augmentations or model sharing so as to maintain privacy. We hope to provide guidance with respect to how foundation models can be effectively used in FL.

3 Distilling Foundation Models in Federated Learning

3.1 Federate Learning: Setup

Our core FL scheme follows FedAvg [McMahan et al., 2017], which consists of a central server and multiple clients, indexed as $i = 1, 2, ..., N$. Each client-i has its local private dataset $\mathcal{D}_i$. We denote the local loss function of interest for the i-th client as $\mathcal{L}(D_i; \theta)$, where $\theta \in \mathbb{R}^d$ are the parameters of the trainable client model. The overall optimization problem considered at the server is denoted as,

$$\min_{\theta \in \mathbb{R}^d} \mathcal{L}(\theta) := \sum_{i=1}^{N} p_i \mathcal{L}(D_i; \theta). \tag{1}$$

Here, p_i is a re-weighting factor conditioned as $p_i \geq 0$ and $\sum_i p_i = 1$. Typically, p_i is assigned as $p_i = \frac{|\mathcal{D}_i|}{\sum_{j \in \mathcal{S}_t} |\mathcal{D}_j|}$ where S_t denotes the set of clients communicating with the server at round t. With this setup in mind, the FL framework repeats the following steps until a desired end condition is achieved: 1) The server broadcasts the current global model to selected clients; 2) Each client resets its local model with the received model, performs local training based on its data, and sends the updated weights/gradients to the server; 3) The central server updates the global model by aggregating the received weights/gradients.

3.2 Fed-LPFM

Setup Unique to our framework, we consider the scenario where each client has access to local pre-trained foundation models. Similar to each client's training dataset, these foundation models are only accessible by the client and not other entities in FL. We assume that in the FL system each client contains two sets of local models: (a) a set M_i of pre-trained foundation models (private): $\mathcal{M}_i^1, \mathcal{M}_i^2, \ldots, \mathcal{M}_i^{M_i}$, and (b) one trainable small-scale proxy model parameterized by θ_i. Since the foundation models are private, only the proxy models are circulated among the clients and server to facilitate the exchange of knowledge across the system. Our goal is to minimize the objective in Eq. 1, where the θ to be optimized represents the parameters of the small-scale proxy model while the foundation models are left unmodified.

Local Training In our algorithm, the client uses its locally stored data along with the knowledge from its private foundation models to supervise local training. For this purpose we use the following loss function,

$$\mathcal{L}(D_i; \theta) = \lambda \mathcal{L}_{CE}(D_i; \theta) + (1 - \lambda) \mathcal{L}_{Distill}(D_i; \theta, \mathcal{M}_i^1, \ldots, \mathcal{M}_i^{M_i}). \tag{2}$$

Algorithm 1 Fed-LPFM

Input: Dataset $\mathcal{D}_i$, frozen and private pre-trained foundation models: $\mathcal{M}_i^1, \mathcal{M}_i^2, \ldots, \mathcal{M}_i^{M_i}$ and proxy model θ_0 for each client $i \in [N]$.

Server:
for Round $t = 0, 1, 2, \ldots, T-1$ **do**
  Send θ_t to connected clients $\mathcal{S}_t \subset [N]$. Let $P_t = \sum_{i \in \mathcal{S}_t} |\mathcal{D}_i|$.
  for Client $i \in \mathcal{S}_t$ in parallel **do**
    $\theta_t^i \leftarrow$ **LocalUpdate**(θ_t, i)
    Send the updated model θ_t^i to the central server
  end for
  Server-end aggregation: $\theta_{t+1} = \sum_{i \in \mathcal{S}_t} \frac{|D^i|}{P_t} \theta_t^i$
end for
return: θ_T

LocalUpdate(θ_t, i)
$\theta_t^i = \theta_t$
for epoch $q = 0, 1, \ldots, Q-1$: $\theta_{t,q+1}^i = \theta_{t,q}^i - \eta \tilde{\nabla} \mathcal{L}(\theta_{t,q}^i; \mathcal{M}_1^i, \ldots, \mathcal{M}_{M_i}^i; \mathcal{D}_i)$
return: $\theta_t^i = \theta_{t,Q}^i$

Here, the first term is the local cross entropy loss, denoted as

$$\mathcal{L}_{CE}(D_i; \theta) = \mathbb{E}_{(x,y) \sim \mathcal{D}_i} \ell_{CE}(h(x; \theta), y), \tag{3}$$

where $h(\cdot)$ denotes the outcome of a forward pass through the proxy model. The second term $\mathcal{L}_{Distill}$ is used to distill the knowledge between the proxy model and the pre-trained foundation models. Typically, the Kullback Leibler (KL) Divergence loss is used for this purpose.

$$\mathcal{L}_{Distill}(D_i; \theta, \mathcal{M}_i^1, \ldots, \mathcal{M}_i^{M_i}) = \sum_{m=1}^{M_i} \mathbb{E}_{(x,y) \sim \mathcal{D}_i} \ell_{KL}[h(x; \theta) || \mathcal{M}_i^m(x)] \tag{4}$$

The parameter λ controls the proportion of knowledge distilled from foundation model in comparison to ground-truth labels.

Aggregation scheme After local training, the server synchronizes with the available clients and aggregates the locally updated proxy models. The local models are aggregated with the following re-weighting scheme,

$$\theta_{t+1} = \sum_{i \in S_t} \frac{|D_i|}{\sum_{j \in S_t} |\mathcal{D}_j|} \theta_t^i, \tag{5}$$

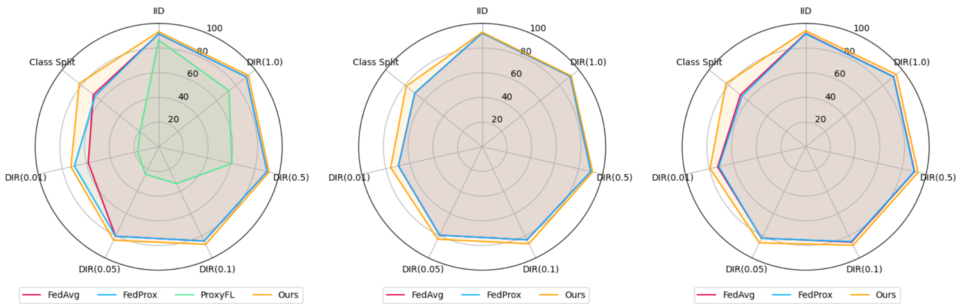
where t denotes the communication round. After the aggregation is complete, the server broadcasts the updated model to clients and the entire process is repeated until a desired end condition is met. Algorithm 1 provides a step-by-step explanation of our FL scheme.

4 Experiments

4.1 Experiments Setup

Data Settings We evaluate our algorithm on the CIFAR-10 dataset with 10 clients across seven data partitions at various levels of heterogeneity, including both IID and non-IID. For the non-IID data partitions we use (1) Dirichlet distribution, denoted as Dir(α) with $\alpha = 1.0,\ 0.5,\ 0.1,\ 0.05,\ 0.01$; (2) Class Split, where each client's data is sampled from 2 of the 10 classes. We evaluate all algorithms over the balanced CIFAR-10 test set and report average accuracy over three trials to mitigate randomness of the runs.

Network Architectures For the choice of foundation models, we employ CLIP Radford et al. [2021] with backbones ViT-Base/32 (default) and RN50, while we use MobileNet-v2, EfficientNetB0, and ResNet18 as our proxy models. In each of the proxy models, we replace the batch normalization layers with group normalization (8 groups) and train it from random initialization.



(a) MobileNetV2 (b) EfficientNetB0 (c) ResNet18

Figure 1: Performance comparisons against existing works across a variety of data settings and proxy model backbones. Fed-LPFM consistently outperforms prior art, by a large margin, under extremely heterogeneous data distributions. Larger area covered indicates a stronger FL approach.

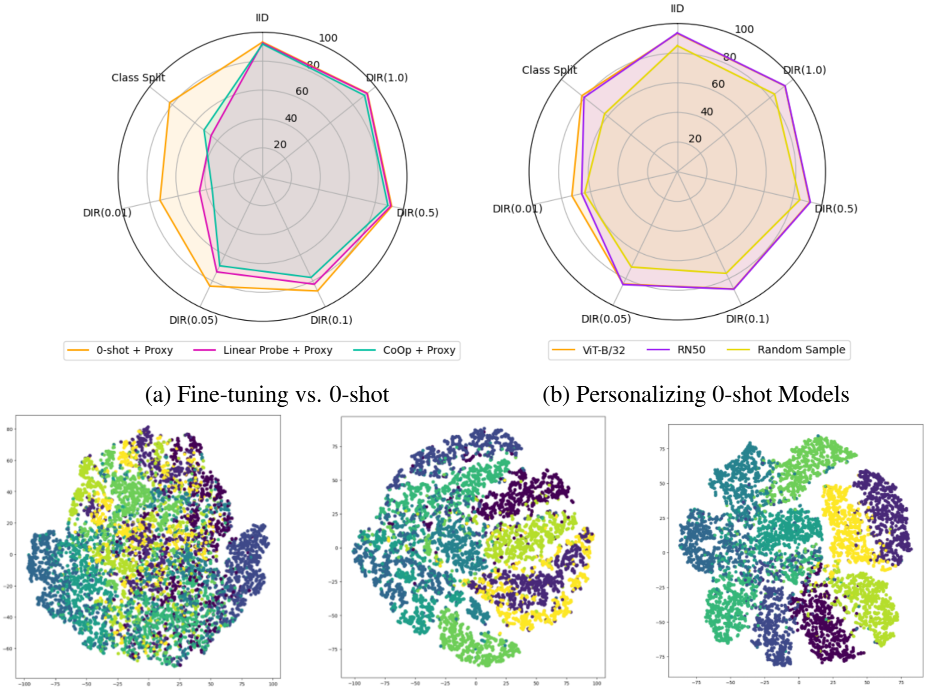
Training Setup and Hyper-parameters Throughout all algorithms and experiments, we use an SGD optimizer for training. We train the proxy models for 600 epochs using a learning rate of 0.01, weight decay of 5e-4, and a step learning rate scheduler with a scale factor of 0.1 at epoch 200. In ablation studies where we additionally consider directly fine-tuning foundation models, we train for 200 epochs with a learning rate of 2e-3, weight decay of 5e-4, and a cosine learning rate scheduler with 1 epoch of warmup. For comparisons against the SOTA algorithms we train the proxy models up to 500 epochs in FedAvg and FedProx, and 600 epochs in FML.

4.2 Main results

SOTA Algorithm Comparison We compare our approach against FedAvg [McMahan et al., 2017], FedProx [Li et al., 2020b], and FML [Shen et al., 2020], under multiple data heterogeneity partitions. We visualize our results in Fig. 1, where each data partition is represented as a vertex on the polar plot and accuracy is plotted along the radius. From Fig. 1, we observe that Fed-LPFM robustly outperforms prior work across a variety of data distributions. *In particular, our algorithm improves over FedProx (the best among prior art) by a wide margin, especially under the most extreme heterogeneous distributions (class split and dirichlet sampling with $\alpha = 0.01, 0.05$).* In addition, we highlight that using MobileNet as the backbone for both the private and proxy models, mimicing the setup in FML, performs poorly. We hypothesize that fine-tuning on the local data begins to bias the representations learned across both models, thus lending to significantly worsening performances as the data heterogeneity increases.

Proxy Model To establish the applicability of our approach to a variety of proxy model backbones, we evaluate across EfficienetNetB0, ResNet18, and MobileNetV2. We report and visualize the results in Figs. 1b and 1c. We observe that our approach outperforms FedAvg and FedProx across the entire selection of proxy models under various data heterogeneity settings, especially the severe non-IID cases. In addition, we also observe that the improvement in performance from FedProx diminishes across both ResNet and EfficienNet, when compared to MobileNet. FedAvg and FedProx perform similar to one another.

Fine-Tuned vs. 0-shot Our Fed-LPFM method uses pre-trained foundation models with no available fine-tuning. To explore the impact of prior knowledge and how it affects distillation, we compare our 0-shot approach with first fine-tuning each client's foundation model(s) on local data (linear probing and prompt tuning). Fig. 2a illustrates how the 0-shot CLIP case outperforms the fine-tuned CLIP models. Our conjecture of this behavior is that *fine-tuning the foundation model on local data results in a more personalized and biased knowledge representation which decreases the performance on a balanced test set.* In addition, the more the data distribution is heterogeneous, the more knowledge encoded locally is personalized and biased. However, under more homogeneous settings there is a significant boost in the performance of clients when leveraging knowledge from both 0-shot and fine-tuned foundation models. We believe that the improvement shown when distilling from 0-shot models is largely due to the impact of strong diversity in its feature embeddings when



(a) Fine-tuning vs. 0-shot (b) Personalizing 0-shot Models

(c) Linear Probing (d) Prompt Tuning (e) 0-shot

Figure 2: (**2a**) Fine-tuning foundation models on local data forces significantly worse performances under non-IID conditions. (**2b**) Maintaining consistent foundation model backbones improves the synergy in information shared across clients. (**Bottom**) When compared to fine-tuned models, 0-shot models offer more diverse feature embeddings that reduce the bias of proxy models towards local data distributions.

compared to fine-tuned foundation models. We use tSNE plots to observe the spread of the encoded knowledge representations from foundation models. From Fig. 2, we can see that features from 0-shot foundation models cover a wider area when compared to fine-tuned models.

Personalizing Foundation Models By keeping foundation model(s) private, Fed-LPFM allows each client to personalize them. From Fig. 2b, we see that maintaining consistent backbones across the foundation models yields the highest improvement in performances while having a random sampling of backbones, between ViT-B/32 and RN50, forces a drop in performance. We believe this behavior stems from following a naive strategy in combining the information presented by multiple proxy models. The root of this behavior can be attributed to differences in knowledge/understanding of foundation models with disparate backbones. Instead, utilizing our approach along with personalized FL ([T Dinh et al., 2020, Fallah et al., 2020, Li et al., 2021, Ghosh et al., 2020, Cho et al., 2021], etc.) could potentially boost the overall performance.

5 Conclusions

Overall, we establish Fed-LPFM as an approach to leverage foundation models and help improve the performance and robustness achievable in small-scale models under the FL setting. Distillation from pre-trained foundation models, as opposed to fine-tuned foundation models, provides the diversity in feature representations required to reduce the bias towards local distributions and thus, improve performance of clients across a variety of heterogeneous data distributions. The use of logit-level distillation allows clients the flexibility to choose their local foundation models according to their individual constraints. In doing so, we establish an important direction of future work; find an approach that, in a synergistic way, combines the information from disparate knowledge representations towards improved model performance.

References

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

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, 2020a.

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.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International conference on machine learning*, pages 5132–5143. PMLR, 2020.

Kevin Hsieh, Amar Phanishayee, Onur Mutlu, and Phillip Gibbons. The non-iid data quagmire of decentralized machine learning. In *International Conference on Machine Learning*, pages 4387–4398. PMLR, 2020.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

Wenhui Wang, Hangbo Bao, Li Dong, Johan Bjorck, Zhiliang Peng, Qiang Liu, Kriti Aggarwal, Owais Khan Mohammed, Saksham Singhal, Subhojit Som, et al. Image as a foreign language: Beit pretraining for all vision and vision-language tasks. *arXiv preprint arXiv:2208.10442*, 2022.

Teli Ma, Shijie Geng, Mengmeng Wang, Jing Shao, Jiasen Lu, Hongsheng Li, Peng Gao, and Yu Qiao. A simple long-tailed recognition baseline via vision-language model. *arXiv preprint arXiv:2111.14745*, 2021.

Liangqiong Qu, Yuyin Zhou, Paul Pu Liang, Yingda Xia, Feifei Wang, Ehsan Adeli, Li Fei-Fei, and Daniel Rubin. Rethinking architecture design for tackling data heterogeneity in federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10061–10071, 2022.

Hong-You Chen, Cheng-Hao Tu, Ziwei Li, Han Wei Shen, and Wei-Lun Chao. On the importance and applicability of pre-training for federated learning. In *The Eleventh International Conference on Learning Representations*, 2022a.

Tao Guo, Song Guo, Junxiao Wang, Xueyang Tang, and Wenchao Xu. Promptfl: Let federated participants cooperatively learn prompts instead of models-federated learning in age of foundation model. *IEEE Transactions on Mobile Computing*, 2023a.

Jinyu Chen, Wenchao Xu, Song Guo, Junxiao Wang, Jie Zhang, and Haozhao Wang. Fedtune: A deep dive into efficient federated fine-tuning with pre-trained transformers. *arXiv preprint arXiv:2211.08025*, 2022b.

Shangchao Su, Mingzhao Yang, Bin Li, and Xiangyang Xue. Cross-domain federated adaptive prompt tuning for clip. *arXiv preprint arXiv:2211.07864*, 2022.

Tao Guo, Song Guo, and Junxiao Wang. pfedprompt: Learning personalized prompt for vision-language models in federated learning. In *Proceedings of the ACM Web Conference 2023*, pages 1364–1374, 2023b.

Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

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.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020b.

Canh T Dinh, Nguyen Tran, and Josh Nguyen. Personalized federated learning with moreau envelopes. *Advances in Neural Information Processing Systems*, 33:21394–21405, 2020.

Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Adaptive personalized federated learning. *arXiv preprint arXiv:2003.13461*, 2020.

Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020.

Filip Hanzely and Peter Richtárik. Federated learning of a mixture of global and local models. *arXiv preprint arXiv:2002.05516*, 2020.

Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE transactions on neural networks and learning systems*, 32(8):3710–3722, 2020.

Yae Jee Cho, Jianyu Wang, Tarun Chiruvolu, and Gauri Joshi. Personalized federated learning for heterogeneous clients with clustered knowledge transfer. *arXiv preprint arXiv:2109.08119*, 2021.

Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. *Advances in neural information processing systems*, 30, 2017.

Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. *Advances in Neural Information Processing Systems*, 33:3557–3568, 2020.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

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.

Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

Zineng Tang, Ziyi Yang, Chenguang Zhu, Michael Zeng, and Mohit Bansal. Any-to-any generation via composable diffusion. *arXiv preprint arXiv:2305.11846*, 2023.

Haodong Zhao, Wei Du, Fangqi Li, Peixuan Li, and Gongshen Liu. Fedprompt: Communication-efficient and privacy-preserving prompt tuning in federated learning. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

Wang Lu, HU Xixu, Jindong Wang, and Xing Xie. Fedclip: Fast generalization and personalization for clip in federated learning. In *ICLR 2023 Workshop on Trustworthy and Reliable Large-Scale Machine Learning Models*, 2023.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

Kan Wu, Houwen Peng, Zhenghong Zhou, Bin Xiao, Mengchen Liu, Lu Yuan, Hong Xuan, Michael Valenzuela, Xinggang Wang, Hongyang Chao, et al. Tinyclip: Clip distillation via affinity mimicking and weight inheritance. *arXiv preprint arXiv:2309.12314*, 2023.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Rohan Anil, Gabriel Pereyra, Alexandre Passos, Robert Ormandi, George E Dahl, and Geoffrey E Hinton. Large scale distributed neural network training through online distillation. *arXiv preprint arXiv:1804.03235*, 2018.

Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4320–4328, 2018.

Haoran Zhang, Zhenzhen Hu, Wei Qin, Mingliang Xu, and Meng Wang. Adversarial co-distillation learning for image recognition. *Pattern Recognition*, 111:107659, 2021.

Tao Lin, Lingjing Kong, Sebastian U Stich, and Martin Jaggi. Ensemble distillation for robust model fusion in federated learning. *Advances in Neural Information Processing Systems*, 33:2351–2363, 2020.

Felix Sattler, Tim Korjakow, Roman Rischke, and Wojciech Samek. Fedaux: Leveraging unlabeled auxiliary data in federated learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

Hyowoon Seo, Jihong Park, Seungeun Oh, Mehdi Bennis, and Seong-Lyun Kim. 16 federated knowledge distillation. *Machine Learning and Wireless Communications*, page 457, 2022.

Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-free knowledge distillation for heterogeneous federated learning. In *International conference on machine learning*, pages 12878–12889. PMLR, 2021.

Tao Shen, Jie Zhang, Xinkang Jia, Fengda Zhang, Gang Huang, Pan Zhou, Fei Wu, and Chao Wu. Federated mutual learning. *CoRR*, abs/2006.16765, 2020. URL `https://arxiv.org/abs/2006.16765`.

Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through personalization. In *International Conference on Machine Learning*, pages 6357–6368. PMLR, 2021.

Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An efficient framework for clustered federated learning. *Advances in Neural Information Processing Systems*, 33:19586–19597, 2020.

A Appendix

The experimental results used to plot Figures in the main paper are shown in tabular form below. Our algorithm (i.e., Fed-LPFM) consistently outperform other algorithms. We observed that Fed-LPFMshows the largest improvement in the class split and Dirichlet distribution (α = 0.01 and 0.05), which are the most heterogeneous data distributions among different clients. For example, in the class split, Fed-LPFM has a 21.6% increase compared with Fedavg (82.29% vs 67.67%).

Data Settings	FedAvg	FedProx	FML	Fed-LPFM (MobileNetV2)
Class Split	67.67 ± 1.88	65.97 ± 5.07	19.29 ± 0.28	$\mathbf{82.29 \pm 1.12}$
Dir(0.01)	58.49 ± 15.72	69.92 ± 2.07	17.50 ± 1.22	$\mathbf{72.88 \pm 9.11}$
Dir(0.05)	80.48 ± 1.07	80.48 ± 1.07	24.75 ± 3.15	$\mathbf{84.05 \pm 0.99}$
Dir(0.1)	84.78 ± 1.65	84.78 ± 1.65	33.13 ± 2.58	$\mathbf{87.73 \pm 1.48}$
Dir(0.5)	90.05 ± 0.21	90.05 ± 0.21	60.77 ± 3.98	$\mathbf{91.72 \pm 0.31}$
Dir(1.0)	90.86 ± 0.13	90.86 ± 0.13	72.84 ± 1.56	$\mathbf{92.87 \pm 0.17}$
IID	91.48 ± 0.31	91.61 ± 0.34	86.49 ± 0.17	$\mathbf{93.26 \pm 0.17}$

Table 1: The values of Figure 1(a) are shown in the current table.

Under different proxy model backbones, similar results can be observed. For both EfficientNet and ResNet case, we observed that Fed-LPFM outperforms other methods cross all data heterogeneity settings.

Data Settings	FedAvg	FedProx	Fed-LPFM (EfficientNetB0)
Class Split	69.81 ± 2.64	69.81 ± 2.64	$\mathbf{78.70 \pm 2.28}$
Dir(0.01)	69.56 ± 2.20	69.56 ± 2.20	$\mathbf{75.98 \pm 1.29}$
Dir(0.05)	79.53 ± 0.95	79.53 ± 0.95	$\mathbf{83.12 \pm 0.76}$
Dir(0.1)	83.67 ± 1.23	83.67 ± 1.23	$\mathbf{87.24 \pm 1.46}$
Dir(0.5)	90.21 ± 0.08	90.21 ± 0.08	$\mathbf{91.80 \pm 0.30}$
Dir(1.0)	91.41 ± 0.35	91.41 ± 0.35	$\mathbf{92.17 \pm 0.09}$
IID	92.22 ± 0.17	92.22 ± 0.17	$\mathbf{92.84 \pm 0.04}$

Table 2: The values of Figure 1(b) are shown in the current table.

Data Settings	FedAvg	FedProx	Fed-LPFM(ResNet18)
Class Split	67.56 ± 4.66	65.99 ± 6.41	$\mathbf{82.50 \pm 2.00}$
Dir(0.01)	73.06 ± 2.10	72.13 ± 1.38	$\mathbf{79.41 \pm 0.95}$
Dir(0.05)	82.22 ± 0.79	82.40 ± 0.46	$\mathbf{86.42 \pm 1.33}$
Dir(0.1)	85.74 ± 1.38	85.23 ± 0.89	$\mathbf{88.59 \pm 1.54}$
Dir(0.5)	90.41 ± 0.30	90.095 ± 0.54	$\mathbf{93.30 \pm 0.10}$
Dir(1.0)	91.13 ± 0.13	90.82 ± 0.25	$\mathbf{93.92 \pm 0.14}$
IID	91.94 ± 0.22	91.45 ± 0.17	$\mathbf{94.00 \pm 0.20}$

Table 3: The values of Figure 1(c) are shown in the current table.

As supporting materials for Fig. 2a, we report the numerical results for testing fine-tuned CLIP (linear probing and prompt tuning) and zero-shot CLIP cross different data heterogeneity levels. We observed that zero-shot CLIP offers best and more robust performances when compared to fine-tuned methods.

As supporting materials for Fig. 2b, we report the results of using CLIP: ResNet50 and CLIP:ViT-B/32 as well as random sampling of them, with uniform prior, as foundation models. It shows that random selection of pre-trained models offers worst performances when compared to the other two.

Data Settings	Linear Probing	Prompt Tuning	0-shot (Ours)
Class Split	45.63 ± 9.48	51.85 ± 4.58	$\mathbf{82.29 \pm 1.12}$
Dir(0.01)	44.76 ± 3.56	35.86 ± 6.03	$\mathbf{72.88 \pm 9.11}$
Dir(0.05)	73.07 ± 0.86	68.39 ± 1.65	$\mathbf{84.05 \pm 0.99}$
Dir(0.1)	82.54 ± 3.21	77.35 ± 3.75	$\mathbf{87.73 \pm 1.48}$
Dir(0.5)	91.08 ± 0.24	89.01 ± 0.12	$\mathbf{91.72 \pm 0.31}$
Dir(1.0)	92.53 ± 0.48	90.32 ± 0.33	$\mathbf{92.87 \pm 0.17}$
IID	92.56 ± 0.17	91.85 ± 0.23	$\mathbf{93.26 \pm 0.17}$

Table 4: The values of Figure 2(a) are shown in the current table.

Data Settings	CLIP: RN50	CLIP: ViT-B/32	Random Selection
Class Split	80.46 ± 4.08	$\mathbf{82.29 \pm 1.12}$	62.75 ± 3.24
Dir(0.01)	66.17 ± 1.19	$\mathbf{72.88 \pm 9.11}$	64.02 ± 3.52
Dir(0.05)	$\mathbf{84.42 \pm 0.41}$	84.05 ± 0.99	71.33 ± 1.52
Dir(0.1)	$\mathbf{87.90 \pm 1.58}$	87.73 ± 1.48	76.05 ± 3.53
Dir(0.5)	$\mathbf{92.04 \pm 0.25}$	91.72 ± 0.31	84.82 ± 0.45
Dir(1.0)	$\mathbf{93.00 \pm 0.10}$	92.87 ± 0.17	83.99 ± 0.99
IID	$\mathbf{93.63 \pm 0.30}$	93.26 ± 0.17	85.01 ± 0.47

Table 5: The values of Figure 2(b) are shown in the current table.