PROMPTFL: Let Federated Participants Cooperatively Learn Prompts Instead of Models – Federated Learning in Age of Foundation Model

Tao Guo, Song Guo, Fellow, IEEE, Junxiao Wang, Xueyang Tang, and Wenchao Xu, Member, IEEE

Abstract—Quick global aggregation of effective distributed parameters is crucial to federated learning (FL), which requires adequate bandwidth for parameters communication and sufficient user data for local training. Otherwise, FL may cost excessive training time for convergence and produce inaccurate models. In this paper, we propose a brand-new FL framework, PROMPTFL, that replaces the federated model training with the federated prompt training, i.e., let federated participants train prompts instead of a shared model, to simultaneously achieve the efficient global aggregation and local training on insufficient data by exploiting the power of foundation models (FM) in a distributed way. PROMPTFL ships an off-the-shelf FM, i.e., CLIP, to distributed clients who would cooperatively train shared soft prompts based on very few local data. Since PROMPTFL only needs to update the prompts instead of the whole model, both the local training and the global aggregation can be significantly accelerated. And FM trained over large scale data can provide strong adaptation capability to distributed users tasks with the trained soft prompts. We empirically analyze the PROMPTFL via extensive experiments, and show its superiority in terms of system feasibility, user privacy, and performance.

Index Terms—Edge computing, prompt federated learning, vision-language model.

I. INTRODUCTION

The ever-growing edge devices, e.g., smart phones, autonomous vehicles, etc., have become the dominant computing platforms today [1]. These devices generate vast amounts of valuable data while providing hidden insights into the human world. Artificial intelligence (AI) nowadays has shown its success to mine the big edge data and produce accurate models that can replace human decisions timely and properly. However, analyzing large amounts of data using sophisticated machine learning algorithms requires significant computing power. Therefore, traditional AI paradigms require to gather all raw data to a cloud center for centralized training, which can incur significant communication overhead and potential privacy leakage, and thus are not desirable for edge users [2], [3], [4].

Federated learning (FL) [5], [6], [7] has emerged as a potential distributed machine learning that allows multiple edge users to jointly train a shared model without sharing their raw data, which has been demonstrated great success in many edge applications, e.g., input word prediction, voice assistant, etc. [8], [9], that can mine massive distributed data without exposing users’ privacy, and thus are widely applied in various edge scenarios. The FL training process comprises of two iterative phases, i.e., local training and global aggregation. Thus the learning performance is determined by both the effectiveness of the parameters from local training and smooth aggregation of them. However, these two requirements are not easy to satisfy in edge environment, i.e., edge users often have limited bandwidth and insufficient data, which can cause inefficient parameters aggregation, excessive training time and reduced model accuracy. Despite the rich opportunities offered by FL, fundamental research problems still need to be addressed before FL can be readily applied to real-world deployments.

Existing research efforts have focused on improving the FL optimization process [10], [11] or refining model architectures [12], [13], but this does not change that FL inherently entails a large number of communication rounds and a large amount of labeled data for training, which are often unavailable for edge users. Such challenges are particularly salient under the combined effect of a long training process and unfavorable factors such as non-IID and unbalanced data [14], limited communication bandwidth, and unreliable and limited device availability. Therefore, there is an urgent need to explore alternative solutions that can mitigate the challenges of existing FL paradigm and make it more feasible to edge users. We revisit the question of how FL mines the distributed data in iterative training rounds, and exploit the emerging foundation model (FM) to optimize the FL training. FM refers to large neural model that trained on large scale data and has strong adaptation capability for various downstream tasks. We let federated participants cooperatively learn prompts instead of models to unleash the power of FM
in a distributed way, whereby both the local training and global aggregation can be significantly accelerated. Our paper aims to provide a new perspective by rethinking if FMs can be applied to FL as a new paradigm of training.

We investigate the behavior of the nascent model in a standard FL setting using popular off-the-shelf FMs, e.g., CLIP, and methods for FM adaptation. We propose PROMPTFL, a framework that replaces existing federated model training with prompt training, i.e., FL clients train prompts instead of a model, which can simultaneously exploit the insufficient local data and reduce the aggregation overhead. PROMPTFL ships an off-the-shelf public CLIP to users and apply continuous prompts (a.k.a. soft prompts) for FM adaptation, which requires very few data samples from edge users. The framework is technically very simple but effective. The focus of our investigation is whether it meets the key principles:

- **Feasibility:** What are the system costs? We examine the feasibility of PROMPTFL on modern hardware, focusing conservatively on personal cell phones. We demonstrate the feasibility of the system in terms of overhead in communication, training, and inference dimensions.
- **Performance:** Are PROMPTFL competitive with FL? FL does not baseline against any such approach, so we implement a proof-of-concept in the framework, spanning a range of popular image classification tasks and benchmarks. We evaluate the generalization and personalization ability of PROMPTFL and observe competitive performance against strong FL baselines even in the heterogeneous environment.
- **Privacy:** Is PROMPTFL privacy-preserving? We show that PROMPTFL keeps data on each device private, aiming to learn global prompts updated only by communicating gradients rather than the data itself, and thus not less private than FL. Furthermore, despite leaving the raw images locally, sharing gradients instead of the whole model parameters further prevent malicious attack, which is a more privacy-preserving framework.

II. PRELIMINARIES

A. Foundation Model

AI is going through a paradigm shift with the rise of models (e.g., BERT, GPT-3, CLIP, DALL-E-2 [15], [16], [17], [18]) trained on broad data using self-supervision at scale that can be adapted to a wide range of downstream tasks. Researchers call these models foundation models (FMs) to emphasize their key core. From a technical standpoint, FMs are not new. However, the sheer size and scope of FMs over the past few years has expanded our imagination of what is possible. FMs are scientifically interesting for their impressive performance and capabilities, but what makes them critical to research is that they are rapidly being integrated into real-world deployments of AI systems, with profound implications for users.

**CLIP**: Contrastive Language-Image Pre-Training (CLIP) is a neural network trained on hundreds of millions of (image, caption) pairs [17]. CLIP encodes images and captions separately as vectors, enabling users with visual modality samples to retrieve, score, or classify samples from textual modalities. Models are often very fragile and only know very specific things you trained them to do. CLIP extends the knowledge of classification models to a wider range of things by leveraging semantic information in text. Standard classification models completely discard the semantic meaning of class labels and simply enumerate numeric classes behind the scenes; CLIP works by understanding the meaning of the classes. ALIGN is another CLIP-like vision-language pre-training [19].

**Image Classification with CLIP**: CLIP pre-trains an image encoder and a text encoder to predict which images are paired with which texts. We can use this behavior to convert the CLIP to an image classifier. We may convert all [class] to captions such as “picture of [class]” and predict the caption class CLIP estimates the best pairing with the given image. In many previous works, this has involved prompt template engineering, in which human engineers or algorithms search for the best template for each class [20], [21], [22], [23].

B. Federated Learning

Recent neural models require large amounts of training data [24], and users typically hold limited-scale labeled data. To address the challenge of lack of sufficient data for individual users, federated learning of data across multiple privacy spheres (i.e., users) has become a popular framework.

The term *federated learning* was introduced by [25]. In a centralized setting, the federated server initially sends global model parameters to each client. After training with local data, the participants are only required to share gradients for model updates. Then the server aggregates the gradients and transmits the updated model back to each client. More specifically, federated learning is a machine learning setting where a set of n clients (e.g., mobile devices) collaboratively train a model under the orchestration of a federated server (e.g., service provider), while the training data of clients is stored locally and not exchanged [26]. The federated server orchestrates the collaborative training process, by repeating the following steps until training is converged:

**Client Selection**: Given the unstable client availability, for the round t of federated learning, the federated server samples a small subset of m clients meeting eligibility requirements out of all n clients to participate in the learning.

**Local Training**: Upon notification of being selected at the round t, each selected client downloads the current parameters θ of global model and a training program from the federated server. Each selected client locally computes an update to the global model on its local training data by executing the training program. More specifically, the gradients updated at one client (denoted as G), are computed by ∂(X, y, θ, G) with X, y denote the batches of training data and corresponding labels, and ℓ(·) refers to the loss function.

The gradients G in typical federated learning settings are the minimum that must be shared to the server, corresponding to FedSGD method. In FedAVG [25], models are consecutively updated on more batches of local data, which can be several epochs of training, and then shared. We note that a common way is to share the updated model θ + G, but this practically amounts to sharing G since all participants know θ.
Global Aggregation: Upon having received local updates from \( m \) clients, the federated server aggregates these updates and update its global model, and initiates next round learning. In addition to the federated learning framework that relies on the centralized server node, there are also some federated learning implementations based on the decentralized framework [27], [28], [29]. This means that the aggregation of gradients does not necessarily occur in a fixed federation server, but may also occur in some clients.

III. PROMPT-BASED FEDERATED LEARNING

We hypothesize that an off-the-shelf public CLIP-like model is shipped to the user device. The CLIP-like model is a powerful image classifier that utilizes linguistic knowledge to classify images. In other words, CLIP already knows a lot about the content of images. Thus, only a few examples are demanded for adaptation, other than the large number of labeled data we need in the existing FL framework. But to unleash the power of CLIP in FL, we need to take advantage of something called prompt engineering that was mentioned in the preliminaries.

A. Prompt Engineering

The prompting function \( f_{\text{prompt}}(\cdot) \) is applied to modify the class label \( y \) into a prompt \( y' = f_{\text{prompt}}(y) \). The most natural form of implementing a prompting function is to manually create an intuitive template based on human introspection. For example, as referred in [16] we may manually craft prefix prompts to handle an image classification task by using templates like “picture of [class]” or “a photo of a [class]”. Based on that, many approaches have been proposed to automate the template design process.

Specifically, the automated prompting can be further separated into discrete prompts (a.k.a. hard prompts), where the prompt is an actual text string, and continuous prompts (a.k.a. soft prompts), where the prompt is performed directly in the embedding space of the model [30]. Discrete prompts constraint that the embeddings of template words be the embeddings of natural language words [31], [32]. Thus, discrete prompting is a clear way to visualize what “word” are learned for the vectors [33].

Our paper adopts continuous prompts instead of discrete prompts in FL for the reason that (1) Our purpose of prompt construction is to find a way to enable FL to efficiently perform the image classification tasks, not for human interpretation, there is no need to limit prompts to human-interpretable natural language. (2) The templates have their own parameters that can be tuned based on training data from the user, which is a natural compatibility connecting FL and prompting. More related topics of continuous prompts can refer to [34], [35], [36], [37], [38].

B. Framework to Learn Prompts in FL

The framework of PROMPTFL is presented in Fig. 1. Each FL client consists of a prompt learner and an out-of-the-box CLIP model. PROMPTFL introduces only a small amount of trainable parameters in the prompt learner while keeping the CLIP backbone frozen. In other words, during local training, only the parameters of the prompt learner are updated while the
whole CLIP model turns off gradients in both the image and the text encoder. The federated server is designed to aggregate only the parameter updates of prompt learners over multiple users, and transmit the updated parameters back to each user. Thus, PromptFL evolves the goal of FL from model training to prompt learner training. To clearly depicts the training procedure in PromtoFL, we presents the algorithm in Algorithm 1.

The CLIP backbone comprises two encoders, one for images and the other for texts. The image encoder will map high-dimensional images into a low-dimensional embedding space. The network of the image encoder can take the form of a CNN such as ResNet50 [39] or Vision Transformer [40]. The text encoder will generate text representations from input. The network of the text encoder is a Transformer [41]. Regarding to the local training manner between traditional FL and PromptFL, PromptFL differs from the existing one from the information capturing. Traditional FL directly leverages one-hot labels and converts to the vectors for regression, PromptFL on the other hand leverages textual vectors for better information extraction and supervision. Thus, the main difference only lies in the supervision manner that act on the visual model, both the image processing part is the same. We presents the structure in Fig. 2.

**Prompt Learner:** Given a pre-trained CLIP backbone, the input to the text encoder is designed in the form of \([\text{prompt vectors}][\text{class}]\). Here we use Transformer as the text encoder and classification tasks as the training objective. Inspired by the simple and straightforward prompt design in [38], we introduce a set of \(p\) continuous embeddings of dimension \(d\) in the \([\text{prompt vectors}]. \(d\) is same as the dimension of word embeddings in the text encoder, thus 512 by default. \(p\) is a hyperparameter specifying the number of embeddings. In a word, \([\text{prompt vectors}]\) are \(p\) learnable \(d\)-dimensional vectors.

Given a batch of image-text pairs, CLIP will maximize the cosine similarity for matched pairs while minimize the cosine similarity for all other unmatched pairs. Since CLIP is pre-trained to predict whether an image matches a textual description, it can compute the classification loss and logits by aligning the two embedding spaces for images and texts (i.e., \([\text{prompt vectors}][\text{class}]\)) respectively. Formally, let \(g(\cdot)\) and \(h(\cdot)\) be the feature extraction function of the image and text encoder. \(w_i = h(P, K_i)\) be the weight vector generated by the text encoder, where \(i \in [1, k]\). \(k\) denotes the number of classes and each \((P, K_i)\) is derived from the prompt in the form of \([\text{prompt vectors}][\text{class}]\), where \([\text{class}]\) is replaced by the word embedding vector of specific class label name. Let \(\cos(\cdot)\) denote the cosine similarity used by CLIP. By forwarding a \((P, K_i)\) and an image \(x\), the classification prediction probability and logits are computed as

\[
p(y = i|x) = \frac{\exp(\cos(g(x)\cdot h(P, K_i)))}{\sum_{j=1}^{k}\exp(\cos(g(x)\cdot h(P, K_j)))},
\]

where \(P\) is the only part that is updated in local back propagation and aggregated in the federated server.

Prompting are particularly useful in the FL case, as using prompts to push the model in the correct direction is particularly effective. Other than traditional FL which trains from the scratch and tunes the whole model parameters with large amount of data and consecutive rounds, PromptFL leverages the pretrained knowledge for adaptation to the downstream tasks. This feature enables prompting to converge quickly in FL, requires less data per user, and is less affected by adverse factors in the process, e.g., non-IID and unbalanced data, limited communication bandwidth, and unreliable and limited device availability. In this paper, the prompt learner employed in PromptFL though simple and straightforward as a bridge to our core idea is easy to follow. We also envision that more complex and effective bridges would be there to replace the role and should be a valuable direction.

### C. System Feasibility

We examine the feasibility of PromptFL on modern hardware, focusing conservatively on personal cell phones. We notice that GPUs resources are served as standard configurations in existing mobile phones. Enterprise users have even more abundant resources. Without loss of generality, we take a 100 M parameter model for FL and 150 M parameter CLIP backbone for image similarity-search of PromptFL, which is effortless for a mobile device with terabyte storage to load on. The prompt learner introduces only a small number of parameters, that can be ignored, compared to the whole model parameters. We assume that the FL configures 32 local training batch size, 1 local training epoch, and 100 total communication rounds, which suggested in [12]. In practice however, PromptFL takes much less communication round than traditional federated learning as shown in Sec. V Fig. 6. We also assume that both FL and PromptFL configures 196 input sequence length and the full precision. The system cost comparison is summarized in Table I along the following dimensions:

** Communication:** The average download speed within the globe for mobile internet was 54 Mbps, and the average upload speed for mobile internet was 12 Mbps that reported by 2021 [43]. PromptFL requires locally downloading while FL requires communicating the model repeatedly between users and the federated server. Thus, the communication cost in terms of file transfer volume is that it takes only 1.4 minutes to transfer 600 MB for PromptFL, and 9 hours for FL to transfer 40 GB.

**Training and Inference:** FL requires FLOPs computed by \((2 \times 3 \times \text{model parameters} \times \text{local training epoch} \times \text{local training batch size} \times \text{input sequence length})\) for training, while the training FLOPs of PromptFL is much smaller and negligible compared to FL. For both PromptFL and FL, inference requires

![Fig. 2. Structure of CLIP-like FM on each client. Each backbone contains an image encoder and a textual encoder. Each encoder extracts feature representations of each modality, and maximize the cosine similarity of the real pairs while minimizing the incorrect pairs.](image-url)
FLOPs computed by \(2 \times \text{model parameters} \times \text{input sequence length}\), in the setting where the key and value vectors for attention computation are cached. Compared to the acceptable computational and storage costs, the RAM on the modern cell phones is a key bottleneck. However, such bottlenecks can be fully addressed in the near future with the following advanced techniques: \(1\) Out-of-the-box offloading inference \([44]\), \(2\) Trends for more RAM \([45]\) and tiny CLIPs \([46]\). \(3\) Inference with quantization methods \([47]\).

Compatibility: Apart from image classification, many different vision tasks are compatible with PROMPTFL, such as object detection \([48]\), video understanding \([49]\) and visual question answering \([50]\). This means that the system cost of PROMPTFL is shared by many tasks. The prompt learner incurs these costs per personal task specific user subset requires. PROMPTFL is thus competitive in terms of economics. Although sharing a similar learning process as the traditional federated learning, PROMPTFL shows advantages in different ways and provides compatibility with existing techniques. Compared with existing federated learning paradigm where models are required to train from the scratch, PROMPTFL leverages pretrained models as base models for adaptation. Moreover, traditional federated learning framework needs to train and transmit the huge amount of the parameters of the whole models with consecutive training rounds while PROMPTFL only needs to train on negligible parameters against the whole models with few rounds. Furthermore, PROMPTFL shares a similar learning process as existing federated framework makes it more convenient in integrating nowadays advanced techniques, e.g., existing secure aggregation techniques for FL. The compatibility of PROMPTFL empowers such framework with a more scalable and rapid integration ability on more practical application scenarios.

### IV. THEORETICAL ANALYSIS

#### A. Definition of PromptFL

The objective function of prompt federated learning is given by

\[
\min_P \mathcal{L}(P) := \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_i(P),
\]

where \(\mathcal{L}_i(P) := \mathbb{E}_{(x,y) \sim D_i} [\ell(x, y; P)]\), \(\ell(\cdot)\) denotes the adopted loss function and \(i \in [n]\). Besides, the detailed training algorithm is given as follows:

- **Server Update:** At the beginning of the training process, the server initializes the global prompt as \(P^0\). During each communication round \(t\), server first selects a random client setround \(S^t\) of \(m\) clients and broadcasts the global prompt \(P^t\) to the participating clients \((S^t)\). After the participating clients finish conducting **Client Update**, server aggregates the received local prompts \(\{P^t_{i,R} | i \in S^t\}\) by \(P^{t+1} = \frac{1}{m} \sum_{i \in S^t} P^t_{i,R}\). With the obtained \(P^{t+1}\), server can starts the next communication round.

- **Client Update:** During each communication round \(t\), the participating client \(i \in S^t\) first initializes the local prompt with \(P^t_{i,0} = P^t\). Then, it can conduct local gradient update for \(R\) iterations. At each local iteration \(r\), the client \(i\) update the local prompt by \(P^t_{i,r+1} = P^t_{i,r} - \eta \nabla \mathcal{L}_i(P^t_{i,r})\) using its local dataset \(D_i\). After finishing local update for \(R\) iterations, client \(i \in S^t\) uploads the local prompt \(P^t_{i,R}\) to the server.
B. Convergence Rate

To derive the convergence rate of our algorithm, we first give the assumptions required for the theoretical analysis.

Assumption 1: For any \(i \in [n]\), loss function \(L_i\) is \(L\)-smooth with respect to \(P\), as follows when \(\forall P, P'\):

\[
\|\nabla L_i(P) - \nabla L_i(P')\| \leq L\|P - P'\|, 
\]

where \(L\) is a finite constant.

Assumption 2: (Bounded variance) The variance of local gradients to the aggregated average is upper-bounded by

\[
\frac{1}{n} \sum_{i=1}^{n} \|\nabla L_i(P) - \nabla L(P)\|^2 \leq \delta_L^2, 
\]

where \(\delta_L^2\) is a finite constant.

According to the definition of loss function, \(L_i(P)\) depends on the local data distribution of \(D_i\), \(\forall i \in [n]\). In this way, Assumption 3 serves as a valuable scheme to measure the degree of data heterogeneity among local clients.

Lemma 1: (Aggregation variance) When Assumption 2 holds, the gradient bias caused by random client selection is upper-bounded by

\[
E_{S_t} \left[ \left\| \frac{1}{m} \sum_{i \in S_t} \nabla L_i(P^t) - \nabla L(P^t) \right\|^2 \right] \leq \frac{n/m - 1}{n - 1} \delta_L^2. \tag{3}\]

Proof: We can write that

\[
E_{S_t} \left[ \left\| \frac{1}{m} \sum_{i \in S_t} \nabla L_i(P^t) - \nabla L(P^t) \right\|^2 \right] = E_{S_t} \left[ \left\| \frac{1}{m} \sum_{i \in S_t} \left( \nabla L_i(P^t) - \nabla L(P^t) \right) \right\|^2 \right] = \frac{1}{m^2} E_{S_t} \left[ \left\| \sum_{i \in S_t} \left( \nabla L_i(P^t) - \nabla L(P^t) \right) \right\|^2 \right] = \frac{1}{m^2} E_{S_t} \left[ \sum_{i \in S_t} \left\| \nabla L_i(P^t) - \nabla L(P^t) \right\|^2 \right. 
\]

\[
+ \sum_{i \in S_t} \sum_{j \neq i} \left( \nabla L_i(P^t) - \nabla L(P^t), \nabla L_j(P^t) - \nabla L(P^t) \right) \right] = \frac{1}{m^2} E_{S_t} \left[ \sum_{i \in S_t} \left\| \nabla L_i(P^t) - \nabla L(P^t) \right\|^2 \right. 
\]

\[
+ \sum_{i \in S_t} \sum_{j \neq i} \left( \nabla L_i(P^t) - \nabla L(P^t), \nabla L_j(P^t) - \nabla L(P^t) \right) \right], 
\]

where \(I_{i \in S_t} = 1\) if \(i \in S_t\); \(I_{i \in S_t} = 0\) otherwise. Since every client \(i \in [n]\) is randomly sampled with identical probability at each communication round \(t\), we have

\[
E_{S_t} \left[ I_{i \in S_t} \right] = p(i \in S^t) = \frac{m}{n} \quad \text{and} \quad E_{S_t} \left[ I_{i \in S_t} I_{j \in S_t} \right] = p(i, j \in S_t) = \frac{m(m-1)}{n^2} \quad \text{for} \quad i \neq j. 
\]

According to the definition in equation 2, we know

\[
\begin{align*}
&\left\| \frac{1}{n} \sum_{i=1}^{n} \nabla L_i(P) - \nabla L(P) \right\|^2 \\
&= \frac{1}{n^2} \left\| \sum_{i=1}^{n} \left( \nabla L_i(P) - \nabla L(P) \right) \right\|^2 \\
&= \frac{1}{n^2} \sum_{i=1}^{n} \left\| \nabla L_i(P) - \nabla L(P) \right\|^2 \\
&+ \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j \neq i} \left( \nabla L_i(P) - \nabla L(P), \nabla L_j(P) - \nabla L(P) \right) \\
&= 0
\end{align*}
\]

Thus, we can write that

\[
E_{S_t} \left[ \sum_{i \in [n]} \sum_{j \neq i} \left( \nabla L_i(P) - \nabla L(P), \nabla L_j(P) - \nabla L(P) \right) \right] = \sum_{i \in [n]} \sum_{j \neq i} \left( \nabla L_i(P) - \nabla L(P), \nabla L_j(P) - \nabla L(P) \right) \\
= \sum_{i \in [n]} \sum_{j \neq i} \left( \nabla L_i(P) - \nabla L(P), \nabla L_j(P) - \nabla L(P) \right) \\
= \frac{m(m-1)}{n(n-1)} \sum_{i=1}^{n} \left\| \nabla L_i(P) - \nabla L(P) \right\|^2 
\]

Therefore, we can derive that

\[
E_{S_t} \left[ \left\| \frac{1}{m} \sum_{i \in S_t} \nabla L_i(P^t) - \nabla L(P^t) \right\|^2 \right] = \frac{1}{m^2} \left\{ \sum_{i=1}^{n} \left[ E_{S_t} \left[ \nabla L_i(P^t) - \nabla L(P^t) \right]^2 \right] \\
- \frac{m(m-1)}{n(n-1)} \sum_{i=1}^{n} \left\| \nabla L_i(P^t) - \nabla L(P^t) \right\|^2 \right\} \\
= \frac{1}{m^2} \left\{ \frac{m}{n} \sum_{i=1}^{n} \left\| \nabla L_i(P^t) - \nabla L(P^t) \right\|^2 \\
- \frac{m(m-1)}{n(n-1)} \sum_{i=1}^{n} \left\| \nabla L_i(P^t) - \nabla L(P^t) \right\|^2 \right\} \\
\leq \frac{n/m - 1}{n - 1} \delta_L^2. 
\]

Proof ends.

Lemma 2: (Local update) When Assumption 1 is satisfied and the learning rate \(\eta \leq \frac{1}{\sqrt{2L}}\), the following inequality holds
for any \( i \in [n] \):

\[
\frac{1}{R} \sum_{r=0}^{R-1} \left\| P_{i,r}^t - P^t \right\|^2 \leq 8 R^2 \eta^2 \| \nabla \mathcal{L}(P^t) \|^2. \tag{4}
\]

**Proof:** As discussed in Section IV-A, we know \( P_{i,r}^t = P_{i,r-1}^t - \eta \nabla \mathcal{L}_i(P_{i,r-1}^t), \forall r \geq 1 \). Therefore, we can write

\[
\left\| P_{i,r}^t - P^t \right\|^2
\]

\[
= \left\| P_{i,r-1}^t - \eta \nabla \mathcal{L}_i(P_{i,r-1}^t) - P^t \right\|^2
\]

\[
= \left\| P_{i,r-1}^t - \eta \nabla \mathcal{L}_i(P_{i,r-1}^t) + \eta \nabla \mathcal{L}_i(P^t) - \eta \nabla \mathcal{L}_i(P^t) - P^t \right\|^2
\]

\[
\leq \left( 1 + \frac{1}{R} \right) \left\| P_{i,r-1}^t - P^t - \eta \nabla \mathcal{L}_i(P^t) \right\|^2
\]

\[
+ (1 + R) \left\| \eta \nabla \mathcal{L}_i(P^t) - \eta \nabla \mathcal{L}_i(P_{i,r-1}^t) \right\|^2
\]

\[
\leq \left( 1 + \frac{1}{R} \right) \left\{ \left( 1 + \frac{1}{2R} \right) \left\| P_{i,r-1}^t - P^t \right\|^2 + (1 + 2R) \right\} \left\| \eta \nabla \mathcal{L}_i(P^t) \right\|^2 \]

\[
(1 + R) \eta^2 L^2 \left\| P_{i,r-1}^t - P^t \right\|^2
\]

\[
+ \left( 1 + \frac{1}{R} \right) \left( 1 + 2R \right) \eta^2 \| \nabla \mathcal{L}_i(P^t) \|^2.
\]

When \( \eta \leq \frac{1}{8RT} \), we have \( R \eta^2 L^2 \leq \frac{1}{2R} \). Furthermore, we can get

\[
\left\| P_{i,r}^t - P^t \right\|^2
\]

\[
\leq \left( 1 + \frac{1}{R} \right)^2 \left\| P_{i,r-1}^t - P^t \right\|^2 + \left( 1 + \frac{1}{R} \right) \]

\[
\times (1 + 2R) \eta^2 \| \nabla \mathcal{L}_i(P^t) \|^2.
\]

Since \( P_{i,0}^t = P^t \), we can derive the following inequality for any \( r \geq 1 \):

\[
\left\| P_{i,r}^t - P^t \right\|^2 \leq \sum_{s=0}^{r-1} \left( 1 + \frac{1}{R} \right)^2 \left( 1 + \frac{1}{R} \right)
\]

\[
\times (1 + 2R) \eta^2 \| \nabla \mathcal{L}_i(P^t) \|^2
\]

\[
= \left( 1 + \frac{1}{R} \right) \left( 1 + 2R \right) \eta^2 \| \nabla \mathcal{L}_i(P^t) \|^2 \left( 1 + \frac{1}{R} \right)^2 r
\]

\[
\leq \left( 1 + \frac{1}{R} \right) \left( 1 + 2R \right) \eta^2 \| \nabla \mathcal{L}_i(P^t) \|^2 \left( 1 + \frac{1}{R} \right)^2 r
\]

\[
= R^2 \left( 1 + \frac{1}{R} \right) \left( 1 + 2R \right) \eta^2 \| \nabla \mathcal{L}_i(P^t) \|^2 \left( 1 + \frac{1}{R} \right)^2 r
\]

\[
= R(1 + R) \eta^2 \| \nabla \mathcal{L}_i(P^t) \|^2 \left( 1 + \frac{1}{R} \right)^2 r.
\]

Therefore, based on the above inequality we can write that

\[
\frac{1}{R} \sum_{r=0}^{R-1} \left\| P_{i,r}^t - P^t \right\|^2
\]

\[
\leq R(1 + R) \eta^2 \| \nabla \mathcal{L}_i(P^t) \|^2 \left( 1 + \frac{1}{R} \right)^2 r
\]

\[
\leq (1 + R) \eta^2 \| \nabla \mathcal{L}_i(P^t) \|^2 \left( 1 + \frac{1}{R} \right)^2 r
\]

\[
\leq (1 + R) \eta^2 \| \nabla \mathcal{L}_i(P^t) \|^2 \left( 1 + \frac{1}{R} \right)^2 r
\]

\[
\leq \frac{1}{2} R(1 + R) \eta^2 \| \nabla \mathcal{L}_i(P^t) \|^2 \left( 1 + \frac{1}{R} \right)^2 r.
\]

We know that \( (1 + \frac{1}{R})^R \leq \lim_{R \to \infty} (1 + \frac{1}{R})^R = e \) and \( e^2 < 8 \). Thus, we can get that

\[
\frac{1}{R} \sum_{r=0}^{R-1} \left\| P_{i,r}^t - P^t \right\|^2 \leq \frac{1}{2} R(1 + R) \eta^2 \| \nabla \mathcal{L}_i(P^t) \|^2 e^2
\]

\[
\leq e^2 R^2 \eta^2 \| \nabla \mathcal{L}_i(P^t) \|^2
\]

\[
< 8 R^2 \eta^2 \| \nabla \mathcal{L}_i(P^t) \|^2, \forall R \geq 1.
\]

**Proof ends.**

**Theorem 1:** Suppose Assumptions 1 and 2 hold. If the learning rate \( \eta \) always satisfies that \( \eta \leq \frac{1}{8RT} \), then the convergence rate of the global prompt is described by

\[
\mathbb{E}[\| \nabla \mathcal{L}(P^*) \|^2]
\]

\[
\leq \mathcal{O}\left( \frac{\epsilon_1}{\beta RT} + \frac{\epsilon_2^2 L^2 \delta_1^2}{T^2} + \frac{\epsilon_3^2 L^2 \delta_1^2}{T^2} + \frac{\sqrt{(n-m)\epsilon_1 L \delta_2^2}}{m(n-1)T} \right),
\]

where \( \epsilon_1 := \mathbb{E}[\| \mathcal{L}(P^t) - \mathcal{L}(P^*) \|], \beta \) is a positive constant that satisfies \( \beta \leq \frac{1}{8RT} \), and \( t^* \) is uniformly sampled from \( \{0, 1, \ldots, T-1\} \).

**Remark 1:** Theorem 1 shows that our algorithm can achieve a convergence rate of \( \mathcal{O}(1/\sqrt{T}) \) under the situation where partial clients are selected at each round and local data distributions are Non-IID. In particular, the convergence rate can reach \( \mathcal{O}(1/T^{2/3}) \) when all clients participate to learn the global prompt at every round. If the local data distributions are IID, the convergence rate can be \( \mathcal{O}(1/T) \) because \( \delta_1^2 = 0 \) under IID case. The detailed proof of Theorem 1 is provided in the following part.

**Proof:** Since the local prompt is updated by \( P_{i,r+1}^t = P_{i,r}^t - \eta \nabla \mathcal{L}_i(P_{i,r}^t), \forall i, r \), we can get \( \sum_{r=0}^{R-1} \eta \nabla \mathcal{L}_i(P_{i,r}^t) = P_{i,0}^t - P_{i,R}^t \). That is,

\[
P_{i,R}^t = P^t - \eta \sum_{r=0}^{R-1} \nabla \mathcal{L}_i(P_{i,r}^t)
\]
After the global aggregation, we can get the updated global prompt at communication round \( t + 1 \) as

\[
P^{t+1} = \frac{1}{m} \sum_{i \in S^t} P^t_{i,R} \]

\[
= \frac{1}{m} \sum_{i \in S^t} \left\{ P^t - \eta \sum_{r=0}^{R-1} \nabla L_i(P^t_{i,r}) \right\}
\]

\[
= P^t - \frac{\eta R}{m} \sum_{i \in S^t} \sum_{r=0}^{R-1} \nabla L_i(P^t_{i,r})
\]

\[
= P^t - \eta \sum_{i \in S^t} \sum_{r=0}^{R-1} \nabla L_i(P^t_{i,r}) \text{,}
\]

When Assumption 1 holds, loss function \( L_i(P) \), \( \forall i \in [n] \) is \( L \)-smooth. We can derive the following inequality:

\[
\|\nabla L(P) - \nabla L(P')\| = \left\| \frac{1}{n} \sum_{i=1}^{n} \nabla L_i(P) - \frac{1}{n} \sum_{i=1}^{n} \nabla L_i(P') \right\|
\]

\[
= \left\| \frac{1}{n} \sum_{i=1}^{n} \left\{ \nabla L_i(P) - \nabla L_i(P') \right\} \right\|
\]

\[
\leq \frac{1}{n} \sum_{i=1}^{n} \|\nabla L_i(P) - \nabla L_i(P')\|
\]

\[
\leq \left\| \nabla L(P) - \nabla L(P') \right\|
\]

\[
\leq \left\| L(P) - P' \right\|, \forall P, P',
\]

which means that the global loss function \( L(P) \) is also \( L \)-smooth. Therefore, we can write

\[
\mathbb{E}_{S^t}[L(P^{t+1}) - L(P^t)]
\]

\[
\leq \mathbb{E}_{S^t}\left[ \|\nabla L(P^t)\| L^t(P^t + 1 - P^t) \right] + \frac{L}{2} \mathbb{E}_{S^t}\left[ \|P^{t+1} - P^t\|^2 \right]
\]

\[
= \mathbb{E}_{S^t}\left[ \|\nabla L(P^t)\| - \alpha \|\psi\| \right] + \frac{L}{2} \mathbb{E}_{S^t}\left[ \|\alpha \|\psi\|^2 \right]
\]

\[
= \alpha \mathbb{E}_{S^t}\left[ \|\nabla L(P^t)\|, \nabla L(P^t) - \psi - \nabla L(P^t) \right] + \frac{\alpha^2 L}{2}
\]

\[
\mathbb{E}_{S^t}[\|\psi\|^2]
\]

\[
= -\alpha \mathbb{E}_{S^t}\left[ \|\nabla L(P^t)\| \right] + \alpha \mathbb{E}_{S^t}\left[ \|\nabla L(P^t)\|, \psi - \nabla L(P^t) \right]
\]

\[
+ \frac{\alpha^2 L}{2} \mathbb{E}_{S^t}[\|\psi\|^2]
\]

\[
\leq -\alpha \mathbb{E}_{S^t}[\|\nabla L(P^t)\|] + \alpha \mathbb{E}_{S^t}[\|\nabla L(P^t)\|]
\]

\[
+ \frac{\alpha^2 L}{2} \mathbb{E}_{S^t}[\|\psi\|^2]
\]

\[
+ \frac{\alpha^2 L}{2} \mathbb{E}_{S^t}\left[ \left\| \frac{1}{nR} \sum_{i=1}^{n} \sum_{r=0}^{R-1} \nabla L_i(P^t_{i,r}) - \frac{1}{n} \sum_{i=1}^{n} \nabla L_i(P^t) \right\|^2 \right]
\]

\[
\leq -\alpha \mathbb{E}_{S^t}[\|\nabla L(P^t)\|] + \alpha \mathbb{E}_{S^t}[\|\nabla L(P^t)\|]
\]

\[
+ \frac{\alpha^2 L}{2} \mathbb{E}_{S^t}[\|\psi\|^2]
\]

\[
+ \frac{\alpha}{2} \mathbb{E}_{S^t}\left[ \left\| \frac{1}{nR} \sum_{i=1}^{n} \sum_{r=0}^{R-1} \nabla L_i(P^t_{i,r}) - \frac{1}{n} \sum_{i=1}^{n} \nabla L_i(P^t) \right\|^2 \right]
\]

\[
+ \frac{\alpha}{2} \mathbb{E}_{S^t}[\|\nabla L(P^t)\|] + \frac{\alpha^2 L}{2} \mathbb{E}_{S^t}[\|\psi\|^2]
\]

We first deal with the third term on the right side of above inequality as follows:

\[
\mathbb{E}_{S^t}[\|\psi - \nabla L(P^t)\|^2]
\]

\[
= \mathbb{E}_{S^t}\left[ \left\| \frac{1}{nR} \sum_{i=1}^{n} \sum_{r=0}^{R-1} \nabla L_i(P^t_{i,r}) - \frac{1}{n} \sum_{i=1}^{n} \nabla L_i(P^t) \right\|^2 \right]
\]

\[
= \mathbb{E}_{S^t}\left[ \left\| \frac{1}{nR} \sum_{i=1}^{n} \sum_{r=0}^{R-1} \nabla L_i(P^t_{i,r}) - \frac{1}{n} \sum_{i=1}^{n} \nabla L_i(P^t) \right\|^2 \right]
\]

\[
+ \frac{1}{nR} \sum_{i=1}^{n} \sum_{r=0}^{R-1} \mathbb{E}_{S^t}[\|\nabla L_i(P^t_{i,r}) - \nabla L_i(P^t)\|^2]
\]

\[
\leq \frac{1}{nR} \sum_{i=1}^{n} \sum_{r=0}^{R-1} \mathbb{E}_{S^t}[\|\nabla L_i(P^t_{i,r}) - \nabla L_i(P^t)\|^2]
\]

\[
+ 2 \mathbb{E}_{S^t}\left[ \left\| \frac{1}{nR} \sum_{i=1}^{n} \sum_{r=0}^{R-1} \nabla L_i(P^t_{i,r}) - \frac{1}{n} \sum_{i=1}^{n} \nabla L_i(P^t) \right\|^2 \right]
\]

\[
+ \frac{1}{nR} \sum_{i=1}^{n} \sum_{r=0}^{R-1} \mathbb{E}_{S^t}[\|\nabla L_i(P^t_{i,r}) - \nabla L_i(P^t)\|^2]
\]

\[
+ 2 \mathbb{E}_{S^t}\left[ \left\| \frac{1}{nR} \sum_{i=1}^{n} \sum_{r=0}^{R-1} \nabla L_i(P^t_{i,r}) - \frac{1}{n} \sum_{i=1}^{n} \nabla L_i(P^t) \right\|^2 \right]
\]
\[ \leq 2E_{S^t} \left[ \frac{1}{m R} \sum_{i \in S^t} \sum_{r=0}^{R-1} \left\| \nabla L_i(P_{i,r}^t) - \nabla L_i(P^t) \right\|^2 \right] \]
\[ + 2E_{S^t} \left[ \left\| \frac{1}{m} \sum_{i \in S^t} \nabla L_i(P^t) - \frac{1}{n} \sum_{i=1}^n \nabla L_i(P^t) \right\|^2 \right] \]
\[ \leq 2E_{S^t} \left[ \frac{1}{m} \sum_{i \in S^t} \sum_{r=0}^{R-1} \left\| P_{i,r}^t - P^t \right\|^2 \right] \]
\[ + 2E_{S^t} \left[ \left\| \frac{1}{m} \sum_{i \in S^t} \nabla L_i(P^t) - \nabla L(P^t) \right\|^2 \right]. \]

Using the inequalities in Lemmas 1 and 2, we can get
\[ E_{S^t} \left[ \left\| \psi^t - \nabla L(P^t) \right\|^2 \right] \]
\[ \leq 16R^2\eta^2 L^2 E_{S^t} \left[ \frac{1}{m} \sum_{i \in S^t} \left\| \nabla L_i(P^t) \right\|^2 \right] + \frac{2(n/m - 1)}{n - 1} \delta_L^2 \]
\[ \leq 32R^2\eta^2 L^2 E_{S^t} \left[ \frac{1}{m} \sum_{i \in S^t} \left\| \nabla L_i(P^t) - \nabla L(P^t) \right\|^2 \right] \]
\[ + 2(n/m - 1) \delta_L^2 \]
\[ = 32R^2\eta^2 L^2 E_{S^t} \left[ \sum_{i \in S^t} \left\| \nabla L_i(P^t) - \nabla L(P^t) \right\|^2 \right] \]
\[ + 2(n/m - 1) \delta_L^2 , \]
\[ \leq \frac{\alpha}{2} \left( 1 - 2\alpha L - 16\alpha^2 L^2 - 64\alpha^3 L^3 \right) E_{S^t} \left[ \left\| \nabla L(P^t) \right\|^2 \right] \]
\[ + 8\alpha^3 L^2 \delta_L^2 + 32\alpha^4 L^3 \delta_L^2 + \frac{2(n/m - 1)\alpha^2 L^2 \delta_L^2}{n - 1} . \]

When \( \eta \leq \frac{1}{|S_T|} \), we have
\[ 1 - 2\alpha L - 16\alpha^2 L^2 - 64\alpha^3 L^3 \geq 1 - 1 \frac{1}{4} - 1 \frac{1}{8} > 1 , \]
\[ \forall R \geq 1 . \]

Thus, we can derive that
\[ E_{S^t} \left[ \left\| \nabla L(P^{t+1}) - \nabla L(P^t) \right\|^2 \right] \leq \frac{\alpha}{8} E_{S^t} \left[ \left\| \nabla L(P^t) \right\|^2 \right] + 32\alpha^4 L^3 \delta_L^2 + \frac{2(n/m - 1)\alpha^2 L^2 \delta_L^2}{m(n - 1)} . \]

In other words, we have
\[ \frac{1}{2T} \sum_{t=0}^{T-1} E_{S^t} \left[ \left\| \nabla L(P^t) \right\|^2 \right] \leq \frac{C_1}{\eta_0 R T} + C_2 R^2 \eta^3 \]
\[ + 3 C_3 R^2 \eta^2 + C_4 R \eta . \]

Using the schemes adopted in [54], [55], [56], we consider the following two cases:
- When \( \eta_0 \leq \min \left\{ \left( \frac{C_1}{C_3 R^2 T^2} \right)^{\frac{1}{4}}, \left( \frac{C_1}{C_4 R T} \right)^{\frac{1}{2}} \right\} \), we choose \( \eta = \eta_0 \). Then, we have
\[ \frac{1}{2T} \sum_{t=0}^{T-1} E_{S^t} \left[ \left\| \nabla L(P^t) \right\|^2 \right] \]
\[ \leq \frac{C_1}{\eta_0 R T} + \frac{C_4^2 C_2^2}{T^4} + \frac{C_3^2 C_4^2}{T^4} + \frac{C_2^4}{T^4} . \]
- When \( \eta_0 \geq \max \left\{ \left( \frac{C_1}{C_3 R^2 T^2} \right)^{\frac{1}{4}}, \left( \frac{C_1}{C_4 R T} \right)^{\frac{1}{2}} \right\} \), we choose \( \eta = \min \left\{ \left( \frac{C_1}{C_3 R^2 T^2} \right)^{\frac{1}{4}}, \left( \frac{C_1}{C_4 R T} \right)^{\frac{1}{2}} \right\} \). Then, we have
\[ \frac{1}{2T} \sum_{t=0}^{T-1} E_{S^t} \left[ \left\| \nabla L(P^t) \right\|^2 \right] \leq \frac{2C_1^2 C_4^2}{T^4} + \frac{2C_1^2 C_2^2}{T^4} \]
\[ + \frac{2C_2^4}{T^4} . \]

Combining these two cases, we can get
\[ \frac{1}{T} \sum_{t=0}^{T-1} E \left[ \left\| \nabla L(P^t) \right\|^2 \right] \]
\[ \leq O \left( \frac{C_1}{\eta_0 R T} + \frac{3C_4^2 C_2^2}{T^4} + \frac{3C_3^2 C_4^2}{T^4} + \frac{3C_2^4}{T^4} \right) . \]
We select a representative collection of recognition
and fine-grained classification from diversified categories.

Our experiments aim to answer the following research questions that are important for the practical deployment of FL methods, while also contributing to our understanding of the PROMPTFL paradigm.

- Is PROMPTFL able to train a competitive performance in FL as compared to which have been the de-facto method on image classification tasks?
- Is PROMPTFL capable of handling heterogeneous data distributions (a.k.a. non-IID settings) across clients?
- Is PROMPTFL competitive with the de-facto method in terms of computational communication overhead?
- What is the difference between PROMPTFL and the fine-tuning of visual pre-trained models in FL?
- What practical tips help the service provider and participants deploy PROMPTFL in FL?

A. Experimental Setup

Datasets: We select a representative collection of recognition datasets used in CLIP as our benchmarks. General Objects: Caltech101 [57] for general object detection. Fine-grained Categories: Flowers102 [58], OxfordPets [59] and Food101 [60] for fine-grained classification from diversified categories. Action Recognition: UCF101 [61]. Texture Classification: DTD [62]. Scene Recognition: Sun397 [63] for scene recognition.

Baselines: As compared to our proposed PROMPTFL, we choose current representative framework in FL, FedAVG, by updating and averaging the weights model weighs collaboratively among server and clients. We compare both training from the scratch and fine-tuning with pretrained models as our baseline method. We select the most prevailing models, Vitb16 and Resnet50, as our backbone in both our image encoder of PROMPTFL and the corresponding backbone in the baseline method. We use Transformer as the text encoder of PROMPTFL, consistent with CLIP.

Fine-tuning versus Prompting: How does the prompting differ from the existing adaptation method in FL? Currently in vision, the standard adaptation method is fine-tuning. Therefore we consider fine-tuning as the de-facto way of adapting visual pre-trained models in FL. Fine-tuning is highly flexible in its usage: it can adapt the pre-trained models to new input domains or new tasks with different output semantics. Yet it also requires some level of access to the pre-trained models: often entire parameters. Unlike fine-tuning, prompting adapts the inputs to a pre-trained model by modifying the model’s inputs. This opens up unique applications: the input-space adaptation puts control in the hands of the FL user; FL users only need to find the prompts, they don’t need to control the pre-trained model itself while training and testing. In this way, FL users can provide adapted images and prompts to an online API that can only operate on their inputs. On the other hand, fine-tuning is typically conditioned on inputs. Its update also directly contains some embeddings of visual feature information. In contrast, the prompts we explore in this paper are input-agnostic across the training data. So the prompting can prevent leaking of user’s private information from FL update to a certain extent.

CLIP PROMPTFL: For CLIP, an image-language model, PROMPTFL organizes users to collaboratively learn prompts as the CLIP’s output transformation function. Given a frozen pretrained CLIP $F$ and a task dataset $D\{(x_m, y_m)\}$ across clients, the target of PROMPTFL is to learn a single, static, task-specific prompting $f_{prompt}$ on class space parameterized by [prompt vectors]. Image classes are represented by labels (e.g., ‘panda’) which are then prompted (i.e., ‘[prompt vectors][panda]’) to specify the context of the user’s task. We follow CLIP’s protocol and compute the cosine similarity of the embeddings for each class, normalized to a probability distribution via softmax. The class with the highest probability is selected as the model output. The prompting is added to the class space to form a prompted output $y + v_f$. During training, PROMPTFL will maximize the likelihood of the correct label $y$,

$$
\max_{f_{prompt}} P_{f_{prompt}}(y + v_f|x),
$$

while the gradient updates are applied only to the [prompt vectors] $v_f$ and the CLIP parameters $F$ remain frozen. During validation, the optimized prompt is added to all test-time classes, $D_{test}\{(x_m, y_m + v_f)\}$, which will be then processed through the frozen $F$.

Training Details: To validate the effectiveness of our method, we compare the performance of PROMPTFL with existing framework by 1) training the collaborative model from the scratch and 2) fine-tuning the full model with pretrained weights. We evaluate the performance on seven representative datasets used in CLIP across various categories like general objects, fine-grained classification, action recognition, texture classification and scene recognition. We report the performance with two representative and influential backbone, Resnet50 (38.3 M parameters) and Vitb16 (86.6 M parameters). All experiments are conducted with Pytorch on GeForce RTX 3090 GPU. Training is performed with SGD with 0.001 learning rate.

For the evaluation metrics, we select three aspects to assess the performance of each method, 1) representative Top-1 accuracy on the test set, 2) F1 score to measure the weighted and unified average of precision and recall, which is more useful especially on unbalanced class distribution, 3) as well as the computational and communication cost reported in Fig. 6. We presuming that higher result on accuracy and F-1 score as well as lower result on computation latency will lead to better a framework, detailed comparison results show the superior if PROMPTFL in Tables II and III.

Main Results: We measure the overall performance of PROMPTFL against existing framework from the perspective of two data distribution settings in Tables II and III. We record the test accuracy and the corresponding F-1 scores for performance.
TABLE II

| Method | Model | Acc | F-1 | F1 | Para |
|--------|-------|-----|-----|----|------|
| From Scratch | Rs50 | 32.4 | 10.5 | 100 | |
|         | Vit   | 32.5 | 12.9 | 100 | |
| Finetuning | Rs50 | 90.0 | 84.7 | 100 | |
|         | Vit   | 93.1 | 89.1 | 100 | |
| PROMPTFL | Rs50 | 90.2 | 86.1 | 0.1 | |
|         | Vit   | 94.7 | 91.8 | 0.01| |

| Method | Model | Acc | F-1 | F1 | Para |
|--------|-------|-----|-----|----|------|
| From Scratch | Rs50 | 33.2 | 25.7 | 100 | |
|         | Vit   | 38.0 | 32.5 | 100 | |
| Finetuning | Rs50 | 92.6 | 91.6 | 100 | |
|         | Vit   | 91.9 | 90.7 | 100 | |
| PROMPTFL | Rs50 | 88.2 | 87.6 | 0.1 | |
|         | Vit   | 90.5 | 90.1 | 0.01| |

The table report the accuracy, F-1 score and learnable parameters according to the corresponding backbone and method under the iid data distribution, where the ‘Para’ represents the percentage of learnable parameters against the whole model parameters. We report the best score of each group with respect to method and model and annotate in bold. Compared with finetuning and training from the scratch, PromptFL only update 0.01% - 0.1% parameters, however, still outperforms other methods in most cases. Despite encountering suboptimal cases, our method still approaches the optimal performance with small gap.

1) For the iid setting, each client shares the same classes, and the shots for each class on client is identical. We can see that from Table II, PROMPTFL obtains superior results with similar or better accuracy and F1 value, but with only 0.01% ~ 0.1% learnable parameters with the iid setting. Specially, for Vit-b16 served backbone, PROMPTFL surpasses the alternatives over the average across benchmarks with only 1% of the learnable parameters compared to the others. While for the Resnet50, although PROMPTFL does not achieve the optimal performance on all datasets, the gap is negligible over the average benchmarks. We also notice that for certain datasets PROMPTFL does not surpass finetuning method, such phenomenon lies in the scarcity and the rareness of these datasets. For example, DTD serves for the texture classification tasks, which is hard to encounter in the real world other than some general datasets. Thus, it is more easier to learn with a bunch of data samples with iterative training rounds such as finetuning method.

2) For the extreme non-iid setting, each client owns the independent and non-overlapping classes. Since fedavg can not address the heterogeneity properly, we also compare PROMPTFL with the advanced algorithms [10], [54] which target on the non-iid situations. Since such algorithms are developed on cnn-based architecture in the beginning, we experiment on resnet-50 backbone as representation. We can observe that from Table III that PROMPTFL achieves competitive performance on both accuracy and efficiency, and outbeats the existing framework comprehensively across all benchmarks by a large margin under the with the non-iid setting.

Superior outcome on both settings manifest the advantage of our proposed PROMPTFL. What’s more, the outstanding generalization ability exhibited in PROMPTFL under the non-iid scenarios further validate the effectiveness of our method. We further analyze the ability with the non-iid setting in the following sections. On the contrary, existing framework shows miserable stability when encountering shifted class distribution other than unified mode by observing the Table III.

Data Distribution Analysis: After obtaining the decent performance in both extreme iid and non-iid settings, we hope to further testify the stability of PROMPTFL and figure out the impact of different data distribution on clients to the performance of PROMPTFL. Inspired by the previous personalized work, we consider the pathological non-iid setting in our experiments. Here we set $n = 50$ clients with $r = 10\%$ participation. To observe the intermediate status, we select a fixed number $p$ of classes on each client, ranging from 5 to 10 to 20, which means that random number of $p$ classes appears on each client. Fig. 3 reports the general test accuracy and F1 with corresponding distribution. From the result, we observe that as the number of classes on each client increases, the performance of both test accuracy and the corresponding F1 value improves. The observation implies that the lack of certain classes may empirically affect the overall performance, which is in consistent with our observations.
TABLE III
PERFORMANCE OF PROMPTFL AGAINST EXISTING FL FRAMEWORK WITH NON-IID DATA DISTRIBUTION

(a) Caltech101

| Method   | Model | Acc | F-1 | Para |
|----------|-------|-----|-----|------|
| From Scratch | Ra50  | 11.2 | 2.8 | 100  |
|          | Vit   | 12.1 | 3.5 | 100  |
| Finetuning | Ra50  | 29.8 | 12.2 | 100 |
|          | Vit   | 29.9 | 12.2 | 100 |
| FedProx  | Ra50  | 63.8 | 42.3 | 100  |
|          | Vit   | 71.7 | 60.1 | 100  |
| PROMPTFL | Ra50  | 88.7 | 84.0 | 0.1  |
|          | Vit   | 94.1 | 90.5 | 0.01 |

(b) Flowers102

| Method   | Model | Acc | F-1 | Para |
|----------|-------|-----|-----|------|
| From Scratch | Ra50  | 16.4 | 6.1 | 100  |
|          | Vit   | 18.4 | 7.9 | 100  |
| Finetuning | Ra50  | 24.4 | 10.7 | 100 |
|          | Vit   | 24.5 | 11.2 | 100 |
| FedProx  | Ra50  | 52.7 | 38.7 | 100  |
|          | Vit   | 64.8 | 60.2 | 100  |
| PROMPTFL | Ra50  | 66.3 | 60.1 | 0.1  |
|          | Vit   | 74.8 | 69.1 | 0.01 |

(c) OxfordPets

| Method   | Model | Acc | F-1 | Para |
|----------|-------|-----|-----|------|
| From Scratch | Ra50  | 8.3  | 3.8 | 100  |
|          | Vit   | 6.8  | 3.4 | 100  |
| Finetuning | Ra50  | 24.8 | 11.3 | 100 |
|          | Vit   | 25.3 | 11.9 | 100 |
| FedProx  | Ra50  | 56.5 | 48.0 | 100  |
|          | Vit   | 44.4 | 37.6 | 100  |
| PROMPTFL | Ra50  | 87.0 | 86.9 | 0.1  |
|          | Vit   | 89.5 | 88.5 | 0.01 |

(d) Food101

| Method   | Model | Acc | F-1 | Para |
|----------|-------|-----|-----|------|
| From Scratch | Ra50  | 14.1 | 7.1 | 100  |
|          | Vit   | 11.9 | 5.9 | 100  |
| Finetuning | Ra50  | 22.9 | 10.2 | 100 |
|          | Vit   | 23.8 | 10.7 | 100 |
| FedProx  | Ra50  | 29.2 | 21.4 | 100  |
|          | Vit   | 22.4 | 10.8 | 100  |
| PROMPTFL | Ra50  | 78.1 | 78.0 | 0.1  |
|          | Vit   | 85.9 | 85.8 | 0.01 |

(e) DTD

| Method   | Model | Acc | F-1 | Para |
|----------|-------|-----|-----|------|
| From Scratch | Ra50  | 7.4  | 2.9 | 100  |
|          | Vit   | 8.5  | 3.2 | 100  |
| Finetuning | Ra50  | 42.4 | 37.3 | 100 |
|          | Vit   | 36.3 | 31.2 | 100 |
| FedProx  | Ra50  | 34.2 | 27.7 | 100  |
|          | Vit   | 30.9 | 24.2 | 100  |
| PROMPTFL | Ra50  | 44.4 | 42.3 | 0.1  |
|          | Vit   | 47.5 | 45.4 | 0.01 |

(f) UCF101

| Method   | Model | Acc | F-1 | Para |
|----------|-------|-----|-----|------|
| From Scratch | Ra50  | 9.6  | 4.6 | 100  |
|          | Vit   | 9.8  | 4.4 | 100  |
| Finetuning | Ra50  | 42.0 | 35.6 | 100 |
|          | Vit   | 36.5 | 29.5 | 100 |
| FedProx  | Ra50  | 34.6 | 27.0 | 100  |
|          | Vit   | 33.8 | 26.5 | 100  |
| PROMPTFL | Ra50  | 64.0 | 60.7 | 0.1  |
|          | Vit   | 70.6 | 68.0 | 0.01 |

(g) Sun397

| Method   | Model | Acc | F-1 | Para |
|----------|-------|-----|-----|------|
| From Scratch | Ra50  | 6.5  | 2.7 | 100  |
|          | Vit   | 5.8  | 2.4 | 100  |
| Finetuning | Ra50  | 23.5 | 15.0 | 100 |
|          | Vit   | 22.1 | 12.2 | 100 |
| FedProx  | Ra50  | 23.0 | 15.1 | 100  |
|          | Vit   | 23.6 | 14.8 | 100  |
| PROMPTFL | Ra50  | 61.1 | 59.9 | 0.1  |
|          | Vit   | 66.9 | 65.5 | 0.01 |

(h) Average

| Method   | Model | Acc | F-1 | Para |
|----------|-------|-----|-----|------|
| From Scratch | Ra50  | 10.6 | 4.3 | 100  |
|          | Vit   | 10.5 | 4.4 | 100  |
| Finetuning | Ra50  | 30.0 | 18.9 | 100 |
|          | Vit   | 28.3 | 17.0 | 100 |
| FedProx  | Ra50  | 42.0 | 31.6 | 100  |
|          | Vit   | 41.7 | 33.5 | 100  |
| PROMPTFL | Ra50  | 70.0 | 67.4 | 0.1  |
|          | Vit   | 75.6 | 73.3 | 0.01 |

The table report the accuracy and F-1 score according to the corresponding backbone and method under the Non-Iid data distribution, where the ‘Param’ represents the percentage of learnable parameters against the whole model parameters. Apart from finetuning and training from scratch, we also compare the existing advanced techniques that target on Ono-Iid settings. We report the best score of each group with respect to method and model and annotate in bold. Other than the iid scenario in Table II, our method surpasses the alternatives method by a significant margin across all datasets under the Non-Iid settings, with only updating 0.01% ~ 0.1% parameters. By contrast, all the baselines cannot well address the shifted class distribution problem caused by Non-Iid setting.

Fig. 3. Performance of PROMPTFL with different class distribution. We evaluate the performance on seven datasets and record the average performance. X-axis represents the number of classes on each client. Bars represent accuracy and lines indicate F-1 score. We place random fixed number of classes on each client and range the number from 5 to 10 to 20. As the number of classes on client gets larger, performance on both the accuracy and the F1 value improve. Furthermore, as the number of classes becomes sufficient, the improvement speed gets slower.
Fig. 4. Performance of PROMPTFL with different shots. We deploy the experiments on seven datasets and record the average one. X-axis represents the number of shots for each class, ranging from 1 to 2 to 4 to 8. Bars represent the local accuracy and lines represent the global accuracy, which implies the personalization and generalization ability respectively. As the number of shots increasing, the local performance improves. However, global performance is not affected much by the variation of number of shots, as we can observe that the global performance remains stable as shots increase.

Fig. 5. Performance of PROMPTFL with different clients. X-axis represents the number of clients during the training, ranging from 20 to 50 to 100. Bars represent the local accuracy and lines represent the global accuracy, which implies the personalization and generalization ability respectively. For each setting, we set the same participation rate of \( r = 10\% \). The overall performance does not obey the strict increasing or decreasing trend as the number of clients changes. We observe that the personalization ability may be affected when the number of clients gets larger, since the clients which do not engage in the training increase. Also, too few clients may lead to insufficient diverse of the training classes, thus lead to under representative of generalization ability.

training logistic. We also notice that as the data for the whole training system reaches sufficient status, the performance becomes stable, as we can see that the gap decreases as the number of classes large enough.

Impact of Number of Shots: Following the few-shot evaluation setting adopted in CLIP, we further explore the effect of number of shots within PROMPTFL. We select fixed number of shots on each client from 1, 2, 4, 8 during the training process and validate the performance with corresponding test sets. We not only observe the generalization ability of PROMPTFL, but also place significant emphasis on the personalization aspect. We record the two indicators as global accuracy and local accuracy. From the result in Fig. 4, we observe that as the number of training examples per class increases, personalized local performance of PROMPTFL enhanced. However, general global performance remains stable as the number of shots fluctuates. The observation implies that the number of training data only influence the personalization performance of the local model, while has little impact over the generalization ability.

Comparison With Different Clients: Further, to explore the possible impact caused by different clients, we further study the performance of PROMPTFL with different clients from 20 to 50 to 100, with the non-iid data distribution and \( r = 10\% \) participation for each mode. We set the fixed shots for different mode, here we set \( s = 2 \). From the result in Fig. 5, we observe that for different number of clients, performance with relatively large clients will be harmed. To analyze both the generalization ability as well as the personalization ability against different number of clients, we obtain two indicators separately. We record the average local personalized test accuracy in representing the personalization
ability and the global test accuracy as the generalization ability. In Fig. 5, the bars show the local accuracy tendency and the dash lines show the global accuracy tendency. This phenomenon is more severe in the local personalized data performance than the general global performance. The reason is that as the number of clients becomes larger, the number of clients which may not be chosen during the training increases. Thus, the local performance is more likely to be affected. On the other hand, for some tasks such as fine-grained categories, action recognition and texture classification, limited clients implies shortage of available data sources which may restrict the diversity of training data. Thus, the general performance in some cases reaches unsatisfactory with fewer clients. However, the number of clients will not influence the performance trend caused by different data distribution or number of shots as shown in Figs. 4 and 3.

**Generalization Ability of PROMPTFL:** Apart from the previous analysis by considering the general performance of the global accuracy, we further analyze the generalization ability of PROMPTFL by observing the performance of the unseen classes in the non-iid settings. We split the data into two groups and only train the model on the base datasets. We record the performance on both base and new classes.

**Computation and Communication Cost Analysis:** We also analyse the efficiency of PROMPTFL with regard to the computation and communication cost during training. We measure the communication cost by the size of uploaded data per round, and the total round to be transmitted. For the computation cost, we calculate the GPU memory utilization and training GPU time for given steps. Fig. 6 shows the comparison between existing finetuning framework and our proposed PROMPTFL. We observe that PROMPTFL can save at most 110 times communication cost per round compared to existing prevailing method, let alone that PROMPTFL takes half of rounds to reach convergence, which makes a wider disparity in communication cost between them. As for the computation cost, we report the comparison of GPU time as in the same given steps, where PROMPTFL remains outperform existing framework around 3 times. Further more, there is huge advantage that PROMPTFL consumes far less GPU memory during training, which can alleviate the system burden in practical.

**Privacy & Security Protection:** We further explore the privacy protection ability in PROMPTFL. We adopt a representative gradient inversion attack method in current distributed learning to reconstruct the raw training images. We compare the prompt scenario and the traditional model based scenario, and leverage dummy data and labels by training 100 iterations in both scenarios. We record the reconstruction results for every 20 epochs and illustrate the process in Fig. 7. We can observe that for the model-based method, the raw images can be recovered after epochs. However, for the prompt-based scenarios, the
reconstructed images closely resemble white noise images, thereby indicating that the sharing of prompts can still uphold privacy and security protection.

VI. CONCLUSION AND FUTURE DIRECTION

Overall, there are many unknowns about PROMPTFL and this paper sets out to investigate its feasibility. In summary: (1) We demonstrate the system feasibility of PROMPTFL on modern hardware, in terms of overhead in communication, training, and inference. (2) We show that PROMPTFL keeps data on each device private, aiming to learn global prompts updated only by communicating gradients rather than the data itself, and thus not less private than FL. (3) We implement a proof-of-concept in the framework, spanning a range of popular image classification tasks. We find PROMPTFL to be competitive with strong FL baselines.

We also summarize the opportunities and future directions for PROMPTFL. Besides the benefits that PROMPTFL provides in computation, communication and privacy aspects, we further consider the possibility for even more tiny devices. Such promising insights can be accomplished with advanced model compression techniques, e.g., quantization, knowledge distillation. Future research may explore the opportunities in obtaining more tiny FMs without compromising the model performance, which will encourage a more practical and widespread application of PROMPTFL.

REFERENCES

[1] X. Wu, H. Wang, Z. Lu, C. Hua, N. Cheng, and S. Guo, “Mobile collaborative learning over opportunistic Internet of Vehicles,” IEEE Trans. Mobile Comput., early access, May 5, 2023, doi: 10.1109/TMC.2023.3273425.

[2] A. Qu et al., “Rethinking architecture design for tackling data heterogeneity in federated learning,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2023, pp. 1083–1092.

[5] M. Tsimpoukelli, J. L. Menick, S. Cabi, S. Eslami, O. Vinyals, and F. Hill, “Warp: A more expressive architecture for adversarial reprogramming,” in Proc. NeurIPS Workshop, 2021, pp. 1–10.

[6] A. Vaswani et al., “Attention is all you need,” in Advances in Neural Information Processing Systems, 2017, pp. 5998–6008.

[7] A. Hard et al., “Federated learning for mobile keyboard prediction,” 2018, arXiv:1811.03064.

[8] P. Li et al., “Federated machine learning: Concept and applications,” ACM Trans. Intell. Syst. Technol., vol. 10, no. 2, pp. 1–19, 2019.

[9] A. Hard et al., “Federated learning for mobile keyboard prediction,” 2018, arXiv:1811.03064.

[10] P. Li et al., “Federated learning: Concept and applications,” 2020, arXiv:2001.01523.

[11] T. Li et al., “Federated optimization in heterogeneous networks,” in Proc. Mach. Learn. Syst., vol. 2, pp. 429–450, 2020.

[12] Y. Zhao, M. Li, L. Lai, N. Suda, D. Cavin, and V. Chandra, “Federated learning with non-IID data,” 2018, arXiv:1806.06582.

[13] L. Qiu et al., “Rethinking architecture design for tackling data heterogeneity in federated learning,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2022, pp. 10051–10061.

[14] T. Guo et al., “Tree learning: Towards promoting coordination in scalable multi-client training acceleration,” IEEE Trans. Mobile Comput., 2023, doi: 10.1109/TMC.2023.3259007.

[15] Q. Li, Y. Diao, Q. Chen, and B. He, “Federated learning on non-IID data silos: An experimental study,” in Proc. IEEE 38th Int. Conf. Data Eng., 2022, pp. 965–978.

[16] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” 2018, arXiv:1810.04805.

[17] T. Brown et al., “Language models are few-shot learners,” in Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 1877–1901.

[18] A. Radford et al., “Learning transferable visual models from natural language supervision,” in Proc. Int. Conf. Mach. Learn., 2021, pp. 8748–8763.

[19] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, “Hierarchical text-conditional image generation with clip latents,” 2022, arXiv:2204.06125.

[20] C. Jia et al., “Scaling up visual and vision-language representation learning with noisy text supervision,” in Proc. Int. Conf. Mach. Learn., 2021, pp. 4904–4916.

[21] A. Fürst et al., “CLOOB: Modern hopfield networks with infoloob outperform clip,” 2021, arXiv:2110.11316.

[22] Y. Li et al., “Supervision exists everywhere: A data efficient contrastive language-image pre-training paradigm,” in Proc. Int. Conf. Learn. Representations, 2021, pp. 1–17.

[23] A. Singh et al., “FLAVA: A foundational language and vision alignment model,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2022, pp. 15617–15629.

[24] L. Yuan et al., “Florence: A new foundation model for computer vision,” 2021, arXiv:2111.11432.

[25] J. Dodge, G. Iharce, R. Schwartz, A. Farhadi, H. Hajishirzi, and N. Smith, “Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping,” 2020, arXiv:2002.06305.

[26] B. McMahen, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in Proc. Int. Conf. Artif. Intell. Statist., 2017, pp. 1273–1282.

[27] P. Kairouz et al., “Advances and open problems in federated learning,” Foundations Trends Mach. Learn., vol. 14, no. 1-2, pp. 1–210, 2021.

[28] A. G. Roy, S. Siddiqui, S. Pölsterl, N. Navab, and C. Wachinger, “Brain-Torrent: A peer-to-peer environment for decentralized federated learning,” 2019, arXiv:1905.06731.

[29] A. Lalitha, S. Shekhar, T. Javid, and F. Koushanfar, “Fully decentralized federated learning,” in Proc. NeurIPS Workshop Bayesian Deep Learn., 2018, pp. 1–9.

[30] C. Hu, J. Jiang, and Z. Wang, “Decentralized federated learning: A segmented gossip approach,” 2019, arXiv:1908.07782.

[31] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, “Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing,” 2022, arXiv:2207.11586.

[32] T. Shin, Y. Raazeghi, R. L. Logan IV, E. Wallace, and S. Singh, “Auto-Prompt: Eliciting knowledge from language models with automatically generated prompts,” in Proc. Conf. Empirical Methods Natural Lang. Process., 2020, pp. 4222–4235.

[33] T. Gao, A. Fisch, and D. Chen, “Making pre-trained language models better for few-shot learners,” in Proc. Annu. Meeting Assoc. Comput. Linguistics, 2021, pp. 3816–3830.

[34] M. Deng et al., “RLPrompt: Optimizing discrete text prompts with reinforcement learning,” 2022, arXiv:2205.12548.

[35] X. Li et al. and P. Liang, “Prefix-tuning: Optimizing continuous prompts for generation,” in Proc. Annu. Meeting Assoc. Comput. Linguistics, 2021, pp. 4582–4597.

[36] B. Lester, R. Al-Rfou, and N. Constant, “The power of scale for parameter-efficient prompt tuning,” in Proc. Conf. Empirical Methods Natural Lang. Process., 2021, pp. 3045–3059.

[37] M. Tsimpoukelli, J. L. Menick, S. Cabi, S. Eslami, O. Vinyals, and F. Hill, “Multimodal few-shot learning with frozen language models,” in Proc. Adv. Neural Inf. Process. Syst., 2021, pp. 200–212.

[38] K. Hambardzumyan, H. Khachattian, and J. May, “WARP: Word-level adversarial reprogramming,” in Proc. Annu. Meeting Assoc. Comput. Linguistics, 2021, pp. 4921–4933.

[39] K. Zhou, J. Yang, C. C. Loy, and Z. Liu, “Learning to prompt for vision-language models,” 2021, arXiv:2109.01134.

[40] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2016, pp. 770–778.

[41] A. Dosovitskiy et al., “An image is worth 16x16 words,” Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 5825–5837.

[42] A. E. Freedman, “Apple A15 bionic powers iPhone 13 and iPad mini,” 2021, Accessed: Aug. 11, 2023, [Online]. Available: https://www.tomshardware.com/news/ipad-iphone-13-a15-bionic.
Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.