A Survey on Efficient Federated Learning Methods for Foundation Model Training

Herbert Woisetschläger[1] , **Alexander Erben**[1] , **Shiqiang Wang**[2] ,
Ruben Mayer[3] and **Hans-Arno Jacobsen**[4]
[1]Technical University of Munich
[2]IBM Research
[3]University of Bayreuth
[4]University of Toronto
{herbert.woisetschlaeger, alex.erben}@tum.de, wangshiq@us.ibm.com, ruben.mayer@uni-bayreuth.de,
jacobsen@eecg.toronto.edu

Abstract

Federated Learning (FL) has become an established technique to facilitate privacy-preserving collaborative training across a multitude of clients. However, new approaches to FL often discuss their contributions involving small deep-learning models only and focus on training full models on clients. In the wake of Foundation Models (FM), the reality is different for many deep learning applications. Typically, FMs have already been pre-trained across a wide variety of tasks and can be fine-tuned to specific downstream tasks over significantly smaller datasets than required for full model training. However, access to such datasets is often challenging. By its design, FL can help to open data silos. With this survey, we introduce a novel taxonomy focused on computational and communication efficiency, the vital elements to make use of FMs in FL systems. We discuss the benefits and drawbacks of parameter-efficient fine-tuning (PEFT) for FL applications, elaborate on the readiness of FL frameworks to work with FMs and provide future research opportunities on how to evaluate generative models in FL as well as the interplay of privacy and PEFT.

1 Introduction

Foundation Models (FMs) [Bommasani *et al.*, 2021] have conquered the deep learning world with unprecedented speed, enabling generative artificial intelligence for a broad audience. As FMs have been pre-trained on an extensive data basis and can be used in multi-modal applications, they perform well over a wide range of tasks. To specialize these models on a downstream task, we use fine-tuning that can either be done over the full model or with parameter-efficient fine-tuning techniques (PEFT) [Hu *et al.*, 2021; Lester *et al.*, 2021; Zaken *et al.*, 2021]. A major advantage is that fine-tuning requires orders of magnitude smaller datasets than pre-training but benefits from access to a variety of samples pertaining to a task.

Access to a breadth of data has always been challenging in deep learning, as data owners are typically reluctant to share their data with service providers. To tackle the data access challenge, McMahan *et al.* [2017] introduced *Federated Learning* (FL). FL enables privacy-preserving machine learning over decentralized data without the necessity of sharing input data and does not require high bandwidth client connections. Rather, a set of clients collectively train a model and send their local model updates to a server that subsequently aggregates the updates to a global model. In FL applications that involve small models with less than 1 million parameters, we may spend as much time on communication as we spend on computation [Yousefpour *et al.*, 2023]. However, it is desirable to design systems in such a way that computation takes up the majority of time. Luckily, the larger the models become, the time spent on training is larger than on communication [Ryabinin *et al.*, 2023; Woisetschläger *et al.*, 2023; Beutel *et al.*, 2020]. As such, scaling the model size in FL systems can be desirable, and this introduces beneficial properties that can aid us in training large models with several 100 million parameters and beyond. These properties render FL the perfect choice for fine-tuning FMs for downstream tasks. FL can provide access to a significantly larger and more diverse data basis while benefiting from the increased time spent on the computation of FMs over small models. Yet, the costs of transmitting model updates remain significant even with the increased computational load of FMs [Yousefpour *et al.*, 2023], making it a priority to jointly optimize the training and communication efficiency.

Our survey is the first to study advances in computational and communication-efficient methods for *FM training* in FL applications. Our work contains three distinct contributions:

- **A novel taxonomy on FL methods for FM training focused on the core challenges in computation and communication**. We discover a gap between FL methods to increase computational efficiency and techniques to improve communication efficiency. While we see research on computational efficiency for FM training and fine-tuning in FL applications, communication efficiency methods are predominantly tailored to full-model training. Our taxonomy aims to identify synergies between FL methods for FMs and efficient communication methods.

- **Holistic evaluation of existing FL computational efficiency methods for FMs and communication efficiency techniques**. We study how existing techniques can help drive the adoption of FMs in FL applications and what needs to be done to render FL frameworks ready for large models.
- **We thoroughly discuss future research directions**. We highlight future directions for research on computational and communication efficiency as these domains grow closer together. Also, we show what is necessary to make FMs in FL applications a reality, especially with regard to generative tasks and privacy considerations.

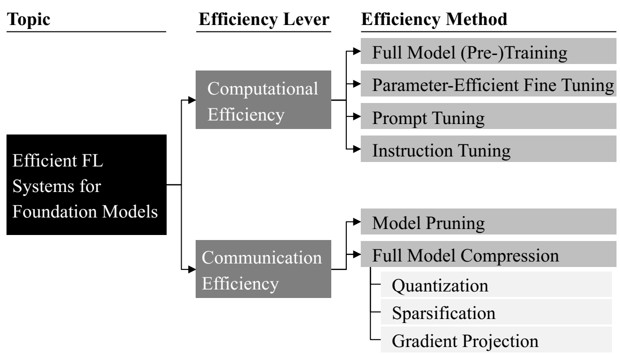

Figure 1: Our Taxonomy. Foundation Models, in conjunction with Federated Learning, require efficient computational and communication methods.

2 Taxonomy

Our taxonomy introduces a novel perspective focused on the current developments in the field of efficient computational and communication methods for FL applications intended for training and fine-tuning FMs. While communication efficiency has been studied extensively in FL, computational advancements in conjunction with large models are currently emerging. Our taxonomy is visualized in Figure 1.

2.1 Basics of Federated Learning

To understand the relevancy of our taxonomy, it is key to briefly introduce the fundamentals of FL and the notations of this paper.

Usually coordinated by a server, the overall goal of FL is to collaboratively and iteratively train a DL model across a set of clients N and minimize the global loss, where f denotes an objective function with parameters w,

$$\min_{w} f(w) := \frac{1}{|N|} \sum_{n=1}^{|N|} f_n(w). \tag{1}$$

With this, we create a model that generalizes across all clients $n \in N$. Specifically, at the beginning, we commonly initialize the model weights across clients. Then, clients train the model on their local data and return the updated model parameters to the server. At the same time, each client updates its local parameters with fixed minibatch size m over one or more epochs by applying gradient descent steps $\nabla l(w_t^n; m)$ to the model,

$$w_{t+1}^n = w_t - \eta \nabla l(w_t^n; m), \, \forall n \in N. \tag{2}$$

Subsequently, only the model parameters w_{t+1}^n are communicated back to the server.

2.2 Taxonomy Explanation

Our taxonomy is centered on two major challenges in FL: the computational and communication efficiency levers. While these challenges have been studied in FL extensively in the past [Zhang *et al.*, 2021], existing approaches predominantly focus on small models with $< 10\text{M}$ parameters. With the emergence of FMs as the backbone for multi-modal applications, we need to combine computational and communication efficiency in FL as these FMs typically come with > 1 billion parameters [Zhang *et al.*, 2023a]; a growth factor of $100\times$ This introduces additional computational load for FL clients, while their resources are often scarce already [Beutel *et al.*, 2020]. At the same time, communication loads are also growing since many parameters need to be transmitted.

Computational Efficiency. We discuss computational efficiency levers along four major categories. Full model training is used to train large transformer models from the very beginning (Section 3.1). Parameter-efficient fine-tuning techniques can be utilized to improve a pre-trained FM for a specific downstream task (Section 3.2). Prompt tuning enables performance improvements of an FM without training the model itself but by designing textual prompts that we prepend to an input (Section 3.3). Instruction tuning enables fine-grained control over the model training process and allows for a high degree of model specialization for certain downstream tasks (Section 3.4).

Communication Efficiency. While computationally efficient methods for FM training in FL applications can already reduce the number of parameters to communicate, there is still a significant amount of data to be communicated between clients and servers. For instance, parameter-efficient fine-tuning methods only require 1–2% of parameters to be trainable. This still amounts to up to 14M parameters when working with Alpaca-7B [Zhang *et al.*, 2023a], larger than the majority of models (with $< 1\text{M}$ trainable parameters) currently being discussed in FL research [He *et al.*, 2020; Beutel *et al.*, 2020].

We discuss communication efficiency methods along two major categories. (I) *Model pruning* is a method to communicate parts of a model between clients and the server, which resemble the most important parameters for a client (Section 4.1). (II) *Full model compression* is divided into three sub-categories: First, quantization is a method to decrease the numeric precision of model parameters. Second, sparsification is used as a way to zero out less important model parameters. Third, gradient projection transforms high dimensional parameter matrices in client updates into scalar vectors for communication (Section 4.2).

Paper	Training Regime	Underlying FL Aggregation Strategy	Max. Model Params	Trainable Parameters	Comms Savings compared to FMT	Domain CV	NLP	Audio
Centralized Learning	FMT, PEFT, PT, IT	–	$\geq$ 7B	–	–	✓	✓	✓
FedBERT [Tian *et al.*, 2022]	FMT	Weighted Average	117M	110M	0%		✓	
FedCLIP [Lu *et al.*, 2023]	PEFT	Weighted Average	85M	530K	> 99%	✓		
FedPEFT [Sun *et al.*, 2023]	PEFT	Weighted Average	85M	230K	> 99%	✓		
FedPETuning [Zhang *et al.*, 2023b]	PEFT	Weighted Average	125M	1.25M	99%		✓	
SLoRA [Babakniya *et al.*, 2023]	PEFT	Weighted Average	67M	140K	> 99%		✓	
FedDPA [Yang *et al.*, 2024]	PEFT	Weighted Average	7B	N/A	> 99%		✓	
FedPrompt [Zhao *et al.*, 2022]	PT	Weighted Average	223M	–	> 99%		✓	
FedIT [Zhang *et al.*, 2023a]	IT	Weighted Average	7B	–	> 99%		✓	

Table 1: Computational Efficiency Methods for FL Systems and FM. FMs are generally multi-modal and provide a strong performance across a variety of domains that are well explored in centralized learning but not in FL [Dosovitskiy *et al.*, 2020; OpenAI, 2023; Yang *et al.*, 2023].

3 Computational Efficiency

This section discusses recent methods to train FMs with FL methods. We distinguish between full model training (FMT), as it has been studied frequently in the FL domain, PEFT, prompt tuning (PT), and instruction tuning (IT). Table 1 summarizes existing computational efficiency methods for FM training.

3.1 Full Model Training

Generally, full model training is referred to when training all parameters of a neural network. In this approach, we locally train a model on all clients with the objective of minimizing the loss l.

The BERT model is one of the first models to use the transformer layer architecture – the building blocks of FMs – to achieve state-of-the-art performance at the time of its release. Tian *et al.* [2022] discuss federated pre-training of BERT-family models with up to 117M parameters by applying masked language modeling (MLM). In MLM, the loss is defined over the sum of probabilities P of predicted tokens $\hat{x}$ over the representation function of a masked sentence $g\left(\frac{x}{M(x)}\right)$,

$$l(w_n^t, m) = -\sum_{\hat{x}\in\mathcal{M}(\mathrm{x})} \log P\left(\hat{x} \,\middle|\, g\left(\frac{x}{M(x)}\right)\right), \forall n \in N. \tag{3}$$

In FL applications, supervised learning can be challenging as we cannot ensure a proper data labeling process for supervised learning because we usually cannot access client data. Here, MLM can be beneficial since it is a self-supervised learning technique that masks parts of a sentence. Those masked parts will be used as the prediction target to automatically create an input and target sample. Federated pre-training provides a net benefit over training on local datasets [Ding *et al.*, 2023; Babakniya *et al.*, 2023; He *et al.*, 2020]. However, the perplexity, an indicator for quality in large models, yields four orders of magnitude worse results for large models (e.g., GPT-2) trained in a federated fashion than centralized training, which is a stark indicator of low model quality [Tian *et al.*, 2022; Radford *et al.*, 2019]. Nonetheless, for use cases involving sensitive data and strict privacy regulation, full model pre-training allows the creation of foundation models based on federated data.

While FL pre-training has shown some promise, it is brittle and has shown to be worse at model sizes above $100\mathrm{M}$ parameters [Tian *et al.*, 2022]. the applicability of model pre-training is currently limited due to the significantly lower model quality than in centralized training.

3.2 Parameter Efficient Fine-Tuning

Generally, PEFT is used to improve the performance of large models already trained on a large data corpus further and provide good performance across various tasks. This is especially effective since the data required for fine-tuning is orders of magnitude smaller than for pre-training. When applying PEFT, additional fully connected layers are inserted into the pre-trained model between the transformer blocks. While the original model weights are frozen, only the newly added layers are trained, typically resulting in $\geq 98\%$ less communication [Hu *et al.*, 2021]. This renders PEFT techniques well-suited for FL applications since they address computation and communication alike. However, Babakniya *et al.* [2023] show in their study that PEFT is more sensitive towards non-IID data than FMT, but this sensitivity can be mitigated. PEFT can be applied in FL applications as follows.

Sparse fine-tuning of pre-trained model parameters. As communication is a key concern in FL, reducing the number of parameters to communicate between client and server has become a priority. One of the most used approaches to achieve this is BitFit [Zaken *et al.*, 2021]. The technique freezes almost all model parameters w_t and only trains the bias term b and the final classification layer w_t^{final} over the input features a_t, where the next-layer input features $a_{t+1} = a_t \cdot w_t + b$. With this technique, it is only required to communicate b and w_t^{final}. The communication load is reduced by $\geq 99\%$, i.e., instead of communicating 100 million parameters, only 100 thousand parameters are sent over the network.

Sun *et al.* [2023] propose FedPEFT, a framework for federated transformer fine-tuning that freezes the model weights to retain upstream knowledge within the model and adjust the systematic error for the downstream task. Their experimental results on vision transformers (ViT-B with 85M parameters) show on-par performance compared to full model fine-tuning on non-IID data, all the while reducing communication by 99.8%.

Adapter-based fine-tuning of additionally added parameters. When aiming to maintain a pre-trained model while introducing task-specific knowledge, adapter-based fine-tuning techniques provide strong performance and on-par efficiency compared to sparse fine-tuning techniques [Houlsby *et al.*, 2019]. Here are two adapter layers introduced in each transformer block of a foundation model. The adapter layer a^a_{t+1} is calculated based on a downstream projection w_t^{down} of input feature a^a_t into a lower-dimensional space $w_t^{\text{down}} \in \mathcal{R}^{d\times r}$ followed by an upstream projection $w_t^{\text{up}} \in \mathcal{R}^{r\times u}$, resulting in

$$a^a_{t+1} = w_t^{\text{up}} \cdot h(w_t^{\text{down}} \cdot a^a_t). \tag{4}$$

With this, we can fine-tune an FM over much fewer dimensions than when fully fine-tuning a model. This saves both computational resources and communication costs in the same range as FedPEFT.

An improvement with regard to computational and communication efficiency over additional adapters is low-rank adapters (LoRA) [Hu *et al.*, 2021]. The technique uses a lower dimensional representation $\mathbf{A} \in \mathcal{R}^{r\times u}$, where u is the dimension of the next layer after the LoRA adapter and $\mathbf{B} \in \mathcal{R}^{d\times r}$, where d is the dimension of the previous LoRA adapter. The weight updates are calculated with $w_{t+1} = w_t + \Delta w = w_t + \mathbf{BA}$. $r \ll \min(d, u)$ casts the weight update matrices into a much lower dimensionality than in the original transformer module without the necessity of adding additional adapters, i.e., LoRA builds an adapter for existing parameters. However, as $\mathbf{A}$ is initialized randomly to a Gaussian distribution and $\mathbf{B}$ as a zero matrix, this works well for centralized settings with IID data [Hu *et al.*, 2021]. For FL settings, this initialization method bears the risk of slowing down the fine-tuning process over non-IID data.

FedCLIP [Lu *et al.*, 2023] introduces a PEFT method with an adjusted FedAvg-based adapter aggregation technique. Their approach yields significant performance improvements over vanilla FedAvg and FedProx.

Zhang *et al.* [2023b] provide a systematic benchmark study on adapter-based fine-tuning methods in privacy-preserving FL systems. Their results show that fine-tuning with additional adapters and LoRA both yield the same benchmark results regarding model accuracy. However, LoRA requires 66% less communication than additional adapters.

However, both FedCLIP and FedPETuning yield a worse accuracy than full fine-tuning. SLoRA [Babakniya *et al.*, 2023] and FedDPA [Yang *et al.*, 2024] are LoRA-based techniques to fine-tune models in non-IID FL settings. Their approach parameterizes the weight update based on r,

$$w_{t+1} = w_t + \frac{\beta}{r}\mathbf{BA}. \tag{5}$$

As β depends on r, scaling the ratio helps control the weight update impact of a single client. Subsequently, this can be used to control inconsistent training updates caused by non-IID data. To practically achieve this, Babakniya *et al.* [2023] make use of a two-stage process: First, they used singular vector decomposition on $\mathbf{A}$ and $\mathbf{B}$ to obtain a common initialization point for LoRA across all clients in an FL system. Second, the training is facilitated with the commonly initialized low-rank representations. Their approach achieves on-par performance with full model fine-tuning. However, they require a warmup time of approximately 100 FL rounds for stage 1, which can be very expensive in FL settings as clients are often unavailable consecutively for such a large number of FL rounds [McMahan *et al.*, 2017].

3.3 Prompt Tuning

Prompt tuning is another efficient method for tuning pre-trained models to a downstream task. Here, we use binary sentiments subsequent to masked-language-modeling to achieve high-quality results [Lester *et al.*, 2021]. As such, the model remains entirely frozen, and we only tweak the prompts (a very small number of tokens) that are being prepended to each embedded input query to improve the output quality. In contrast to fine-tuning, this method does not interfere with the model architecture or parameters. Lester *et al.* [2021] show that the effects of prompt tuning on model performance in centralized settings become better with larger models, i.e., for large FMs, prompt tuning bears significant potential.

Specifically, in FL settings, for each client n the likelihood P for a desired output $\hat{x}$ is calculated over prepending trainable embedded prompts x_p to each embedded input x. In the interactive training process, the prompt is optimized in such a way that it optimally resembles the local objective of a client,

$$\max(P_n(\hat{x}|[x_p; x])), \forall n \in N. \tag{6}$$

Zhao *et al.* [2022] introduce FedPrompt, a method to efficiently communicate federally generated prompts only and aggregate them such that the global model performance of a pre-trained model improves for a downstream task. Their experimental evaluation shows a general sensitivity of prompt tuning towards data heterogeneity as the model performance degrades by 5 – 10% for the 100M parameter BERT model compared to a centrally trained baseline. However, with RoBERTa Base (124M parameters), the sensitivity diminishes, and the FL results are on par with centralized training. The larger T5 Base model (223M parameters) follows this trend, showing that prompt tuning becomes more effective with larger model sizes [Lester *et al.*, 2021].

3.4 Instruction Tuning

Some applications work with highly sensitive and protected data or require a very high model performance. The previously mentioned fine-tuning techniques may not yield sufficient results in these cases.

This is where instruction tuning comes into play as a technique that uses high-quality data. For instance, GPT-2 [Radford *et al.*, 2019] uses Reinforcement Learning with Human Feedback (RLHF). RLHF is a multi-stage process where an FM is initially trained on supervised data. In the second step - reward model training - the FM generates outputs over a given prompt, which a user then ranks based on their preference. With this, the model learns human preferences. In the third step - proximal policy optimization - the model trains self-supervised for a maximum reward [Zheng *et al.*, 2023].

With FedIT, Zhang *et al.* [2023a] study instruction tuning on LLaMA-7B in an FL application over heterogeneous client tasks, e.g., learning brainstorming and text summarization on different clients in a single system at the same time. Their results show that the additional context on a downstream task generated with federated instruction tuning provides net benefits over central training only, even in heterogeneous settings. However, these results were only produced over a single dataset, Dollybricks-15k.

3.5 Discussion

As more open-source FMs become available for unrestricted use (e.g., Alpaca [Taori *et al.*, 2023], Falcon [Penedo *et al.*, 2023]), it is unlikely that full model training will be a common use case since training an FM from scratch is very challenging, even in a centralized setting.

Therefore, we see a priority in improving upon PEFT for downstream tasks, for instance, by introducing new and enhancing existing algorithms to remove the currently required warm-up times to improve the performance of LoRA in non-IID data environments. Computer Vision (CV) applications may benefit from exploring prompt tuning for vision transformers [Jia *et al.*, 2022]. Also, we find significant challenges for instruction tuning as data quality is a general issue in FL [Longpre *et al.*, 2023], and access to human preferences, as it is required for RLHF, is hardly available in a real-world federated setting without incentive schemes.

Furthermore, PEFT also positively affects communication efficiency by significantly reducing the number of trainable (and thus communicable) parameters. As such, we now see computational and communication efficiency growing closer for FL and FMs, but not to a sufficient degree. As such, there is still a stark need to develop new approaches to use PEFT to improve computational and communication efficiency.

4 Communication Efficiency

With FMs, models have exponentially grown from a few million to several billion parameters to be able to serve multi-modal tasks [Bommasani *et al.*, 2021]. For FL, this specifically means that the communication load between clients and servers has grown significantly, even though with PEFT, there is not necessarily the need to communicate an entire model. However, when fine-tuning a billion-parameter FM with adapter-based methods, we still need to facilitate communication for millions of parameters. For cross-device scenarios involving more than 1,000 clients per training round, the data traffic can quickly overstrain a server's network capacity and potentially incur significant communication costs for data transfer from the edge to a cloud [Xu *et al.*, 2023; Erben *et al.*, 2023]. Therefore, efficient communication and training design is vital for future FL systems. We distinguish two major efficiency methods: model pruning and full model compression. A detailed overview of studies on efficient communication in FL is provided in Table 2.

4.1 Model Pruning

The objective of model pruning (MP) is to retain and communicate only parts of a DL model that are relevant to a certain task. The reduction of parameters with this technique reduces the communication effort [Zhu and Gupta, 2017]. However, the success of pruning highly depends on the underlying data. Pruning client models without coordination may deny convergence with heterogeneous non-IID data in FL systems. Jiang *et al.* [2022] introduce PruneFL, a two-stage procedure to realize model pruning in FL systems. The first stage is carried out on a powerful client to find a common initialization for the model and generate the importance-based pruning mask. This mask is then iteratively refined over multiple FL training rounds under the consideration of all clients. The experimental results with PruneFL show two remarkable results: (I) The models have a shorter time to accuracy over the same task with PruneFL than with FedAvg, which is attributable to the higher degree of model specialization. (II) Since the model size is reduced, one would expect additional effects on faster computation, but there is limited hardware support for sparse matrix multiplications in training as they are required in PruneFL. Therefore, PruneFL has no computational benefits to this point. However, this may change with new hardware, such as sparse Tensor cores that support PruneFL's dynamic pruning approach [Zhu *et al.*, 2019]. With FedTiny, Huang *et al.* [2022] present an approach that works identically to PruneFL, except for them swapping the common initialization procedure with using batch normalization values of clients to choose a common initialization. Furthermore, FjORD [Horváth *et al.*, 2021] and HeteroFL [Diao *et al.*, 2020] provide similar approaches to pruning.

Isik *et al.* [2022] choose a similar approach for pruning models based on the lottery ticket hypothesis, first introduced to FL by Li *et al.* [2020]. Instead of commonly initializing a pruned model for an FL system, they initialize a random binary mask based on a shared seed on each client. This reduces computational efforts in the ramp-up phase. After an FL training round, each client communicates their binary mask to the server, which creates a global model based on the weighted average of those binary masks. With this, an approximate weight estimate replaces the parameters on the global model. From client to server, FedPM achieves significant communication efficiencies. However, the full model still has to be communicated from the server to the clients, lowering the net benefit.

Model pruning has also been discussed extensively for FM fine-tuning outside of FL [Lagunas *et al.*, 2021; Sanh *et al.*, 2020]. As existing pruning approaches have shown strong benefits to delivering on-par model performance compared to fine-tuning the full model, this is a promising direction to combine federated PEFT with highly efficient pruning techniques to further enhance communication. Along with pruning, sparse tensor hardware can lower computational loads.

4.2 Full Model Compression

Model pruning is prone to omit segments of a DL model that may become relevant at a later stage. This originates from domain shifts [Peng *et al.*, 2020] and might require preserving the full model with all its parameters. For this, three frequently discussed techniques for full model compression in FL systems are Quantization, Sparsification, and Gradient Projection.

Quantization (Q). The first work towards dynamic quantization is FedPAQ [Reisizadeh *et al.*, 2020], which combines FedAvg with strong quantization guarantees, where Q represents the quantization term for a local model update,

$$w_{t+1} = w_t + \frac{1}{|N|} \sum_{n=1}^{|N|} Q(w^n_{t+1} - w_t). \tag{7}$$

Algorithm	Enhancement Method(s)	Underlying FL Aggregation Strategy	Max. Model Parameters	Communication Savings vs. FedAvg	Domain CV	NLP	Audio
Centralized Learning	MP, Q, S, GP	–	≥ 7B	–	✓	✓	✓
FedKSeed [Qin *et al.*, 2023]	GP	Weighted Average	3B	≥ 99%		✓	
FedOBD [Chen *et al.*, 2023]	Q	Weighted Average	17M	89%	✓	✓	
PruneFL [Jiang *et al.*, 2022]	MP	Weighted Average	132M	80%	✓		
FedPM [Isik *et al.*, 2022]	MP	Weighted Average	12M	98%	✓		
FedTiny [Huang *et al.*, 2022]	MP	Weighted Average	132M	97%	✓		
SoteriaFL [Li *et al.*, 2022]	S	FedSGD	0.05M	N/A	✓		
FjORD [Horváth *et al.*, 2021]	MP	Weighted Average	11M	98%	✓	✓	
FedPAQ [Reisizadeh *et al.*, 2020]	Q	FedSGD	0.2 M	90%	✓		
HeteroFL [Diao *et al.*, 2020]	MP	Weighted Average	11M	98%	✓		
LotteryFL [Li *et al.*, 2020]	MP	Weighted Average	138M	50%	✓		

Table 2: Communication-efficient FL methods. Their centralized learning pendants are often tied to specific domains: CV [Habib *et al.*, 2023], NLP [He *et al.*, 2021], and Audio [Perez *et al.*, 2020].

While DL models often operate on full precision (32-bit), this high degree of detail is not necessarily required [Zhou *et al.*, 2018]. FedPAQ leverages this to reduce the communication intensity of FL applications. Q calculates the optimal float precision of a model update to preserve all required information: $Q(w) = \|w\| \cdot \text{sign}(w) \cdot \xi(w, s)$, as proposed in QSGD by Alistarh *et al.* [2017]. ξ formulates a stochastic process to dynamically tune s, the level of precision. FedPAQ has a significantly lower time to accuracy than QSGD, which is attributable to dynamizing s. However, it must be noted that dynamic quantization only yields benefits for communication. Depending on the infrastructure, the model updates have to be cast back to full precision, creating additional computational overhead on the client and server. Also, the method has been only tested with small models ($< 100\text{K}$ parameters).

FedOBD [Chen *et al.*, 2023] quantizes models with transformer block dropout, i.e., the random removal of entire model blocks. The dropout mechanism is carried out during training by each client and returns only the top-k most important model blocks to communicate. Additionally, FedOBD includes the ideas discussed in Alistarh *et al.* [2017] and Reisizadeh *et al.* [2020] but proposes an optimization problem out of the stochastic quantization where the trade-off originates from entropy and update size. The communication required for FL with a 17M parameter transformer model shows FedOBD to cut communication cost by $2\times$ vs. FedPAQ and by $8\times$ compared to vanilla FedAvg [Chen *et al.*, 2023].

Sparsification (S). While model pruning and sparsification technically have the same objective, pruned models do not necessarily resemble sparse models. A model is sparse once more than 50% of weights are set to 0 [Frankle and Carbin, 2019]. However, pruning can also change the model architecture, i.e., not return the full model. Since it lends its idea from [Frankle and Carbin, 2019], FedPM [Isik *et al.*, 2022] (see Section 4.1) can be considered as a model sparsification technique but does not necessarily lead to sparse networks and may return partial networks. SoteriaFL [Li *et al.*, 2022] guarantees sparse networks while maintaining differential privacy. Equation 7 is amended in such a way that $Q(w^n_{t+1} - w_t)$ is replaced by $C(w^n_{t+1} + \mathcal{N}(0, \sigma^2\mathcal{I}))$ with C resembling a sparse client update through shifted compression that has proven to improve convergence of DL models in FL settings compared to direct compression [Mitra *et al.*, 2021; Mishchenko *et al.*, 2019]. Overall, SoteriaFL mitigates the trade-off between model utility and compression, i.e., the differentially private models converge faster in stricter compression regimes than previously existing non-compressed differentially private approaches.

Gradient projection (GP). Qin *et al.* [2023] introduce FedKSeed tailored to efficiently train FMs in FL applications. They do so by using seeds in the form of scalar vectors to create gradient projections. As only the scalar vectors have to be transmitted, the total amount of communication is reduced by $\geq 99\%$ compared to applications that would send the original multi-dimensional gradients. This is the first technique that enables communication efficient FL applications with FMs, regardless of the training regime (pre-training or fine-tuning).

4.3 Discussion

To date, advancements in communication-efficient methods for FL systems have predominantly focused on training small, full models. The communication paradigm shifts with the emergence of FMs in FL applications. With fine-tuning tasks, we only need to train a small fraction of model parameters, and thus, only trainable parameters have to be communicated. However, each trainable parameter usually contains a high degree of information for a downstream task. As such, the effectiveness of model pruning techniques is unclear as they would cut away fine-tuned parameters. An unexplored space in FL research is the pruning of FMs with subsequent fine-tuning for downstream tasks. Pruning FMs can lead to smaller transformer layers and, consequently, smaller PEFT layers with fewer parameters. In turn, this could positively affect computational and communication efficiency. Full-model compression techniques do not alter the model structure but rather reduce the parameter precision. Thus, these techniques can be used with FL applications and FMs in order to further reduce the size of the communicated updates. Furthermore, Qin *et al.* [2023] have shown that gradient projection is a promising direction to compress model updates without sacrificing significant information. This can be beneficial for PEFT applications as we have small specific adapters for downstream tasks. However, the remaining key challenge is the effect of non-IID data on PEFT. Potential compounding effects of communication compression and lossy compression remain open for investigation.

Framework	Secure Aggregation	Training Efficiency	Communication Efficiency	FM Training / Fine-Tuning	Edge Ready
FLARE	DP, HEC		Currently, none of the Frameworks implements communication efficient FL methods.	✓	
FedML	DP, HEC	PEFT		✓	✓
FederatedScope		PEFT		✓	✓
Flower	DP, SMPC	PEFT		✓	✓
FATE	HEC, SMPC	PEFT		✓	✓
Substra					
PySyft	DP				
OpenFL					
TFF	DP				✓
IBM FL	HEC				

Table 3: Current capabilities of state-of-the-art FL framework with respect to our taxonomy and their ability to run in resource-limited environments. Key: DP = Differential Privacy, HEC = Homomorphic Encryption, SMPC = Secure Multi-Party Computation.

5 Are FL Frameworks Ready for FMs?

The backbones for making FL applications available to a broad audience are FL frameworks implementing recent advancements in FL research. We investigate widely used frameworks for their FM readiness and progress in integrating computational and communication efficiency (Table 3).

In theory, all FL frameworks could handle FMs with sufficient hardware availability. Yet, only some implement efficient training methods. To further drive the adoption of FL in times of FMs, the frameworks need to improve both computationally- and communication-efficient methods for training. FL frameworks that are characterized by their active open-source community, FLARE [Roth *et al.*, 2022], FATE [Fan *et al.*, 2023; Liu *et al.*, 2021], FedML [He *et al.*, 2020], TensorFlow Federated (TFF) [Google, 2019], FederatedScope [Kuang *et al.*, 2023; Xie *et al.*, 2023] and Flower [Beutel *et al.*, 2020], have adopted recent advancements in FL research. The frameworks especially allow for PEFT of FMs with LoRA. Substra [Galtier and Marini, 2019], PySyft [Ziller *et al.*, 2021], OpenFL [Foley *et al.*, 2022], and IBM FL [Ludwig *et al.*, 2020], in their versions as of 2023, focus on training smaller FL tasks with 100K up to a 10M parameters and, therefore, do not provide adapters for FM workloads with more than 100M parameters. Yet, a consistent observation across all frameworks is their lack of efficient communication techniques (e.g., FedOBD). Workloads with FMs will significantly increase communication costs, and the growing use cases involving resource-constrained edge and IoT devices require high efficiency for computation and communication. As such, only those FL frameworks enabling training efficiency are viable choices for working with FMs.

6 Related Work

While there are ample surveys that provide a broad perspective on FL [Li *et al.*, 2023; Banabilah *et al.*, 2022; Liu *et al.*, 2022; Nguyen *et al.*, 2021; Zhang *et al.*, 2021; Aledhari *et al.*, 2020], there are two closely related surveys to our work as they also focus on FMs.

In their survey, Zhuang *et al.* [2023] introduce a broad and general perspective on FMs and FL. They extensively discuss data modalities. This includes access to data across a large number of highly distributed clients and the quality of data that lives on these clients. Currently, FM training or fine-tuning requires datasets with high data quality, i.e., the instructions or texts used for MLM must be curated very carefully. Thus, their survey identifies a stark need for methods to train or fine-tune FMs on a scattered data basis with (highly) varying data quality. Further, Zhuang *et al.* [2023] discuss approaches to integrate FL applications into the lifecycle of FMs, i.e., how FMs can benefit from a continuously evolving system. While their survey briefly touches upon computational efficiency, our study provides an in-depth overview of state-of-the-art training and fine-tuning techniques to render FMs in FL applications a reality. Furthermore, our study includes a comprehensive overview of communication techniques that can enhance the adoption of FMs in communication resource-constrained environments.

Yu *et al.* [2023] provide an overview of FMs and FL with a special focus on privacy, an integral component of FL. Their survey includes a comprehensive overview of different fields of application for FMs, which they divide into once-off training and continual learning. The authors elaborate on technical challenges that may arise for specific use cases, such as robustness towards unreliable clients, varying data quality, the degree of non-IID data, and scalability. In contrast, our survey provides an application-agnostic, in-depth study of existing methods suitable for FM training. Our focus is to outline the technical challenges that currently hinder the operationalization of FM for use cases in federated applications.

7 Conclusions & Future Directions

In this paper, we survey the current landscape of computational and communication efficiency methods in FL systems and introduce a novel taxonomy based on the key techniques. While efficient FL methods have been separate topics on their own in the past, they become closely intertwined as we start using FL systems to leverage FMs. Consequently, the following three questions arise:

What are good and realistic evaluation strategies for generative downstream tasks in FL settings where we do not have control of data? Fine-tuning generative FM requires high-quality data. However, we do not have access to data on the clients to monitor data quality before or during training. As such, estimating data quality is of utmost importance.

How does hyperparameter optimization work for FMs in continuously evolving FL systems? While hyperparameter optimization in FL has been a key challenge, PEFT adds additional complexity. As such FL systems must adapt to the era of FMs by introducing adaptive parameterization for PEFT techniques that can account for changing environmental conditions.

We must develop an understanding of the interplay between PEFT and privacy in FL systems. Communication-efficient FL techniques have been studied for their effect on privacy, but this is still an open topic for PEFT, PT, and IT. While it is proven that PEFT is more sensitive to data heterogeneity, the effects of perturbation through differential privacy are still subject to further studies. The same is true for PT and IT, as both techniques require precise prompts and instructions, respectively. As such, noise may have significantly negative effects here, as well.

Acknowledgments

This work is partially supported by the Bavarian Ministry of Economic Affairs, Regional Development and Energy (Grant: DIK0446/01).

References

[Aledhari *et al.*, 2020] Mohammed Aledhari, Rehma Razzak, Reza M. Parizi, and Fahad Saeed. Federated learning: A survey on enabling technologies, protocols, and applications. *IEEE Access*, 8:140699–140725, 2020.

[Alistarh *et al.*, 2017] Dan Alistarh, Demjan Grubic, et al. Qsgd: Communication-efficient sgd via gradient quantization and encoding. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[Babakniya *et al.*, 2023] Sara Babakniya, Ahmed R. Elkordy, et al. Slora: Federated parameter efficient fine-tuning of language models, 2023.

[Banabilah *et al.*, 2022] Syreen Banabilah, Moayad Aloqaily, et al. Federated learning review: Fundamentals, enabling technologies, and future applications. *Information Processing & Management*, 59(6), 2022.

[Beutel *et al.*, 2020] Daniel J. Beutel, Taner Topal, et al. Flower: A friendly federated learning research framework, 2020.

[Bommasani *et al.*, 2021] Rishi Bommasani, Drew A. Hudson, et al. On the opportunities and risks of foundation models, 2021.

[Chen *et al.*, 2023] Yuanyuan Chen, Zichen Chen, et al. Fedobd: Opportunistic block dropout for efficiently training large-scale neural networks through federated learning. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*. IJCAI Org., 2023.

[Diao *et al.*, 2020] Enmao Diao, Jie Ding, et al. Heterofl: Computation and communication efficient federated learning for heterogeneous clients, 2020.

[Ding *et al.*, 2023] Ning Ding, Yujia Qin, et al. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235, March 2023.

[Dosovitskiy *et al.*, 2020] Alexey Dosovitskiy, Lucas Beyer, et al. An image is worth 16x16 words: Transformers for image recognition at scale, 2020.

[Erben *et al.*, 2023] Alexander Erben, Ruben Mayer, and Hans-Arno Jacobsen. How can we train deep learning models across clouds and continents? an experimental study, 2023.

[Fan *et al.*, 2023] Tao Fan, Yan Kang, et al. Fate-llm: A industrial grade federated learning framework for large language models, 2023.

[Foley *et al.*, 2022] Patrick Foley, Micah J Sheller, et al. Openfl: the open federated learning library. *Physics in Medicine & Biology*, 67(21), 2022.

[Frankle and Carbin, 2019] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *ICLR*, 2019.

[Galtier and Marini, 2019] Mathieu N Galtier and Camille Marini. Substra: a framework for privacy-preserving, traceable and collaborative machine learning, 2019.

[Google, 2019] Inc. Google. Tensorflow federated: Machine learning on decentralized data. https://www.tensorflow.org/federated, 2019. Accessed: 2023-12-01.

[Habib *et al.*, 2023] Gousia Habib, Tausifa Jan Saleem, and Brejesh Lall. Knowledge distillation in vision transformers: A critical review, 2023.

[He *et al.*, 2020] Chaoyang He, Songze Li, et al. Fedml: A research library and benchmark for federated machine learning, 2020.

[He *et al.*, 2021] Haoyu He, Xingjian Shi, et al. Distiller: A systematic study of model distillation methods in natural language processing, 2021.

[Horváth *et al.*, 2021] Samuel Horváth, Stefanos Laskaridis, et al. FjORD: Fair and accurate federated learning under heterogeneous targets with ordered dropout. In *Advances in Neural Information Processing Systems*, 2021.

[Houlsby *et al.*, 2019] Neil Houlsby, Andrei Giurgiu, et al. Parameter-efficient transfer learning for nlp, 2019.

[Hu *et al.*, 2021] Edward J. Hu, Yelong Shen, et al. Lora: Low-rank adaptation of large language models, 2021.

[Huang *et al.*, 2022] Hong Huang, Lan Zhang, et al. Distributed pruning towards tiny neural networks in federated learning, 2022.

[Isik *et al.*, 2022] Berivan Isik, Francesco Pase, et al. Sparse random networks for communication-efficient federated learning, 2022.

[Jia *et al.*, 2022] Menglin Jia, Luming Tang, et al. Visual prompt tuning, 2022.

[Jiang *et al.*, 2022] Yuang Jiang, Shiqiang Wang, et al. Model pruning enables efficient federated learning on edge devices. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[Kuang *et al.*, 2023] Weirui Kuang, Bingchen Qian, et al. Federatedscope-llm: A comprehensive package for fine-tuning large language models in federated learning, 2023.

[Lagunas *et al.*, 2021] Francois Lagunas, Ella Charlaix, et al. Block pruning for faster transformers. In *Proceedings of the 2021 Conference on EMNLP*, Online and Punta Cana, Dominican Republic, November 2021. ACL.

[Lester *et al.*, 2021] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning, 2021.

[Li *et al.*, 2020] Ang Li, Jingwei Sun, et al. Lotteryfl: Personalized and communication-efficient federated learning with lottery ticket hypothesis on non-iid datasets, 2020.

[Li *et al.*, 2022] Zhize Li, Haoyu Zhao, et al. Soteriafl: A unified framework for private federated learning with communication compression. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

[Li *et al.*, 2023] Qinbin Li, Zeyi Wen, Zhaomin Wu, Sixu Hu, Naibo Wang, Yuan Li, Xu Liu, and Bingsheng He. A survey on federated learning systems: Vision, hype and reality for data privacy and protection. *IEEE Transactions on Knowledge and Data Engineering*, 35(4):3347–3366, April 2023.

[Liu *et al.*, 2021] Yang Liu, Tao Fan, et al. Fate: An industrial grade platform for collaborative learning with data protection. *JMLR*, 22(1), 2021.

[Liu *et al.*, 2022] Ji Liu, Jizhou Huang, et al. From distributed machine learning to federated learning: a survey. *Knowledge and Information Systems*, 64(4), 2022.

[Longpre *et al.*, 2023] Shayne Longpre, Le Hou, et al. The flan collection: Designing data and methods for effective instruction tuning, 2023.

[Lu *et al.*, 2023] Wang Lu, Xixu Hu, et al. Fedclip: Fast generalization and personalization for clip in federated learning, 2023.

[Ludwig *et al.*, 2020] Heiko Ludwig, Nathalie Baracaldo, et al. Ibm federated learning: an enterprise framework white paper v0.1, 2020.

[McMahan *et al.*, 2017] Brendan McMahan, Eider Moore, et al. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*. PMLR, 2017.

[Mishchenko *et al.*, 2019] Konstantin Mishchenko, Eduard Gorbunov, et al. Distributed learning with compressed gradient differences, 2019.

[Mitra *et al.*, 2021] Aritra Mitra, Rayana Jaafar, George J. Pappas, and Hamed Hassani. Linear convergence in federated learning: Tackling client heterogeneity and sparse gradients. In *Advances in Neural Information Processing Systems*, 2021.

[Nguyen *et al.*, 2021] Dinh C. Nguyen, Ming Ding, et al. Federated learning for internet of things: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 23(3), 2021.

[OpenAI, 2023] OpenAI. GPT-4 technical report, 2023.

[Penedo *et al.*, 2023] Guilherme Penedo, Quentin Malartic, et al. The refinedweb dataset for falcon llm: Outperforming curated corpora with web data, and web data only, 2023.

[Peng *et al.*, 2020] Xingchao Peng, Zijun Huang, et al. Federated adversarial domain adaptation. In *ICLR*, 2020.

[Perez *et al.*, 2020] Andres Perez, Valentina Sanguineti, et al. Audio-visual model distillation using acoustic images. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2020.

[Qin *et al.*, 2023] Zhen Qin, Daoyuan Chen, et al. Federated full-parameter tuning of billion-sized language models with communication cost under 18 kilobytes, 2023.

[Radford *et al.*, 2019] Alec Radford, Jeff Wu, et al. Language models are unsupervised multitask learners. 2019.

[Reisizadeh *et al.*, 2020] Amirhossein Reisizadeh, Aryan Mokhtari, et al. Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108. PMLR, 2020.

[Roth *et al.*, 2022] Holger R. Roth, Yan Cheng, et al. Nvidia flare: Federated learning from simulation to real-world. 2022.

[Ryabinin *et al.*, 2023] Max Ryabinin, Tim Dettmers, et al. Swarm parallelism: Training large models can be surprisingly communication-efficient, 2023.

[Sanh *et al.*, 2020] Victor Sanh, Thomas Wolf, and Alexander Rush. Movement pruning: Adaptive sparsity by fine-tuning. In *Advances in Neural Information Processing Systems*, volume 33. Curran Associates, Inc., 2020.

[Sun *et al.*, 2023] Guangyu Sun, Matias Mendieta, et al. Exploring parameter-efficient fine-tuning for improving communication efficiency in federated learning, 2023.

[Taori *et al.*, 2023] Rohan Taori, Ishaan Gulrajani, et al. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023.

[Tian *et al.*, 2022] Yuanyishu Tian, Yao Wan, et al. Fedbert: When federated learning meets pre-training. *ACM Transactions on Intelligent Systems and Technology*, 13(4), 2022.

[Woisetschläger *et al.*, 2023] Herbert Woisetschläger, Alexander Erben, et al. Federated fine-tuning of llms on the very edge: The good, the bad, the ugly, 2023.

[Xie *et al.*, 2023] Yuexiang Xie, Zhen Wang, et al. Federatedscope: A flexible federated learning platform for heterogeneity. *Proceedings of the VLDB Endowment*, 16(5), 2023.

[Xu *et al.*, 2023] Zheng Xu, Yanxiang Zhang, et al. Federated learning of gboard language models with differential privacy. 2023.

[Yang *et al.*, 2023] Dongchao Yang, Jinchuan Tian, et al. Uniaudio: An audio foundation model toward universal audio generation, 2023.

[Yang *et al.*, 2024] Yiyuan Yang, Guodong Long, Taoshu Shen, Jing Jiang, and Michael Blumenstein. Dual-personalizing adapter for federated foundation models. *ArXiv*, abs/2403.19211, 2024.

[Yousefpour *et al.*, 2023] Ashkan Yousefpour, Shen Guo, et al. Green federated learning, 2023.

[Yu *et al.*, 2023] Sixing Yu, J. Pablo Muñoz, and Ali Jannesari. Federated foundation models: Privacy-preserving and collaborative learning for large models, 2023.

[Zaken *et al.*, 2021] Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models, 2021.

[Zhang *et al.*, 2021] Chen Zhang, Yu Xie, et al. A survey on federated learning. *Knowledge-Based Systems*, 216, 2021.

[Zhang *et al.*, 2023a] Jianyi Zhang, Saeed Vahidian, et al. Towards building the federated gpt: Federated instruction tuning, 2023.

[Zhang *et al.*, 2023b] Zhuo Zhang, Yuanhang Yang, et al. FedPETuning: When federated learning meets the parameter-efficient tuning methods of pre-trained language models. In *Findings of the Association for Computational Linguistics: ACL 2023*. ACL, 2023.

[Zhao *et al.*, 2022] Haodong Zhao, Wei Du, et al. Fedprompt: Communication-efficient and privacy preserving prompt tuning in federated learning, 2022.

[Zheng *et al.*, 2023] Rui Zheng, Shihan Dou, et al. Secrets of rlhf in large language models part i: Ppo, 2023.

[Zhou *et al.*, 2018] Yiren Zhou, Seyed Moosavi-Dezfooli, et al. Adaptive quantization for deep neural network. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), 2018.

[Zhu and Gupta, 2017] Michael Zhu and Suyog Gupta. To prune, or not to prune: exploring the efficacy of pruning for model compression, 2017.

[Zhu *et al.*, 2019] Maohua Zhu, Tao Zhang, et al. Sparse tensor core: Algorithm and hardware co-design for vector-wise sparse neural networks on modern gpus. In *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO '52. ACM, 2019.

[Zhuang *et al.*, 2023] Weiming Zhuang, Chen Chen, and Lingjuan Lyu. When foundation model meets federated learning: Motivations, challenges, and future directions, 2023.

[Ziller *et al.*, 2021] Alexander Ziller, Andrew Trask, et al. *PySyft: A Library for Easy Federated Learning*. Springer International Publishing, 2021.