Local Superior Soups: A Catalyst for Model Merging in Cross-Silo Federated Learning

Minghui Chen
The University of British Columbia
Vector Institute

Meirui Jiang
The Chinese University of Hong Kong

Xin Zhang
Meta

Qi Dou
The Chinese University of Hong Kong

Zehua Wang
The University of British Columbia

Xiaoxiao Li*
The University of British Columbia
Vector Institute
xiaoxiao.li@ece.ubc.ca

Abstract

Federated learning (FL) is a learning paradigm that enables collaborative training of models using decentralized data. Recently, the utilization of pre-trained weight initialization in FL has been demonstrated to effectively improve model performance. However, the evolving complexity of current pre-trained models, characterized by a substantial increase in parameters, markedly intensifies the challenges associated with communication rounds required for their adaptation to FL. To address these communication cost issues and increase the performance of pre-trained model adaptation in FL, we propose an innovative model interpolation-based local training technique called "Local Superior Soups." Our method enhances local training across different clients, encouraging the exploration of a connected low-loss basin within a few communication rounds through regularized model interpolation. This approach acts as a catalyst for the seamless adaptation of pre-trained models in in FL. We demonstrated its effectiveness and efficiency across diverse widely-used FL datasets.

1 Introduction

Federated learning (FL) [33] has emerged as a promising methodology for leveraging the power of private data without the need for centralized data governance. However, data heterogeneity in FL poses significant challenges to the design of efficient training for global convergence. With the emergence of the pre-training and fine-tuning paradigm in various applications [14, 18], recent studies [35, 2] have attempted to address the problem of FL under data heterogeneity with pre-trained initialization. Although pre-trained federated learning can speed up convergence compared to random initialization, it still requires a significant number of communication rounds between the server and clients, often amounting to hundreds of rounds [35]. Existing pre-trained models [39, 45] often have an enormous parameter scale, and following the neural scaling law [23], there is a continuous trend toward increasing model parameters. Deploying models with such a large parameter size in FL introduces significant communication overhead. This greatly hampers the flexibility and scalability of model updates. Reducing FL communication overhead can be approached by reducing

*corresponding author

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

the scale of model parameters involved in distributed training [57] or reducing communication rounds [33]. Comparing with reducing model parameters, reducing communication rounds typically leads to a more efficient reduction of network congestion [16], decreased energy consumption on client devices [31], and a lower risk of privacy breaches [59]. *In this paper, we focus on reducing communication rounds in FL with pre-trained model as initialization.*

Typically, increasing the number of local training steps can effectively reduce communication rounds. However, there is an upper limit to the extent of local training step increments. This limitation arises due to the presence of data heterogeneity, where the consistency of optimization among different clients deteriorates with the increasing number of local steps. This optimization inconsistency leads to a discrepancy between local and global models and decelerates the convergence rate of FL. The discrepancy is often called client drift [24]. Previously, some FL methods [24, 43] attempted to introduce proximal terms to regularize local training, with the aim of reducing local overfitting and minimizing the problem of client drift. While these methods can accelerate convergence, they restrict the progress of each local training steps towards the optimal solution, impeding the attainment of FL with more aggressive communication round reductions. Furthermore, these client drift mitigation methods can alleviate local overfitting to some extent, but cannot guarantee that global aggregated models perform well. This situation arises when individual local clients become trapped in isolated low-loss valleys. More specifically, as illustrated in Figure 1, two models from clients 'A' and 'B', even if their optimal model distance is small, still result in a poorly performing aggregated global model. Moreover, the preceding FL methods aimed at minimizing communication rounds exclusively address scenarios involving random initialization, lacking a customized approach tailored to pre-trained models. Recent proposed centralized fine-tuning methods (*e.g.*, model soups [50] and DiWA [41] – a greedy model selection version of model soups) based on model interpolation (averages of a large number of model weights) are effective approaches to seek large connected low-loss region, which are promising for applying in FL to reduce communication rounds. These methods can prevent individual clients from being trapped in isolated low-loss valleys by positioning the global model centrally within a larger low-loss region by overlapping the low-loss regions among clients, as shown in Fig. 1 (right). However, their training efficiency is exceedingly low, requiring complete retraining of numerous models, leading to *significant computational overhead* on clients and intolerable communication costs when applied in FL, due to two aspects: First, they involve a time-consuming model selection phase within the model pool, which consists of all candidate models available for weight interpolation. Secondly, model soups entail an extensive number of model training iterations, lacking prior guidance and relying on brute-force, random, and often redundant training. Many of the trained models end up unused.

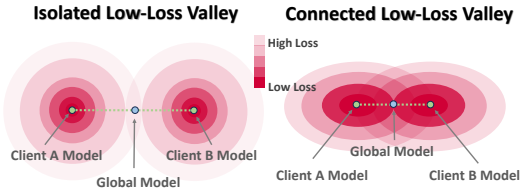


Figure 1: Illustration on isolated (left) and connected low-loss valley with larger regions in dark red (right).

To enjoy the connected low-loss valley benefits of model soup-based methods [50, 41] without burdening local training, we propose an efficient and local model interpolation-based method, called **Local Superior Soups (*LSS*)**. To address the first issue, we propose a **sequential random model interpolation** method during training. This eliminates the need for subsequent model selection steps and ensures that the models slated for interpolation reside within the same low-loss valley during training (Sec. 3.3.1). For the second issue, we introduce two quantifiable indicators of candidate model quality: **diversity** (Sec. 3.3.2) and **affinity** (Sec. 3.3.3). Specifically, the *diversity indicator* quantifies the diversity among models in the model pool with their pairwise model distance, where larger distances denote higher diversity, signifying better model quality for interpolation. As illustrated in Figure 2 (left), a low-loss region, supported by models with low diversity, can be effectively covered with only a few candidate models positioned near its periphery. Thus, we propose incorporating the diversity metric as a regularization term during training to maximize the expansion of low-loss regions, thereby increasing the utilization of trained models. The *affinity indicator* measures the affinity of each candidate model in the model pool to the initial model. Smaller distances indicate greater affinity, indicating better model quality for interpolation. This affinity is also incorporated as

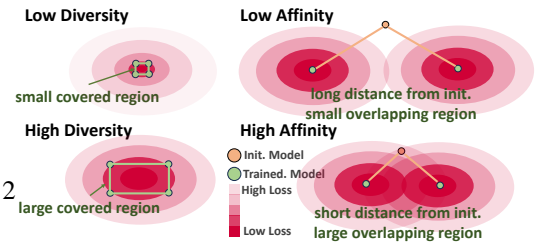


Figure 2: Illustration on diversity (left) and affinity (right) regularization.

a regularization term during training to prevent the expansion of low-loss regions from deviating too far from the shared initialization point, thus increasing the likelihood of overlapping connected regions (as depicted on the right side of Fig. 2). These two indicators facilitate the efficient inclusion of models into the model pool, preventing wasteful training of models that may ultimately go unused. In experiments, we found that our proposed method greatly reduces communication rounds, and we achieved the performance of models fused after multiple rounds of communication in other FL methods with only a few rounds of communication.

In summary, our contributions are as follows.

(1) We reveal the importance of regularizing local client models in the connected low-loss valleys for reducing communication rounds when initializing FL with pre-trained models. (2) We introduce an innovative and efficient model soups-based method for FL, called Local Superior Soups (*LSS*) that eliminates the need for time-consuming model selection and redundant model training in the existing soups-based approaches, while expanding connected low-loss valleys of client models for faster convergence. (3) In experimental evaluations, *LSS* demonstrates a significant reduction in communication rounds, achieving superior performance with only a few rounds of communication, exceeding baseline FL methods significantly in four datasets and two types of distribution shifts.

2 Related Work

2.1 Heterogeneous Federated Learning

FL struggles with Non-IID data, leading to various proposed algorithms. FedProx [27] uses proximal term to regularize local training, preventing client divergence. Scaffold [24] adds variance reduction to combat "clients-drift." MOON [26] employs mode-level contrastive learning to stabilize local training. Personalized FL [44] targets high local performance on Non-IID data. FedBN [29] applies local batch normalization to mitigate feature shift before model averaging. Recent one-shot communication round FL methods utilize server-side techniques like prediction ensembles [13] or generating data [54, 17] for centralized training, improving aggregated model performance. Few-round communication round FL, based on meta-learning [37], may not align with practical FL scenarios due to data partition concerns.

2.2 Fine-tuning and Model Interpolation

Fine-tuning leverages pre-trained models to enhance task performance [6]. FedFTG [55] proposes knowledge distillation for global model fine-tuning in FL. Personalized FL employs fine-tuning to adapt global models to local ones, e.g., FedBABU [36], FTFA, and RTFA [4]. However, this focus on local performance neglects global generalization. Inspired by linear mode connectivity [34, 11], Model Soups [50] combines runs with varied hyper-parameters. DiWA [41] extends this concept, emphasizing diverse hyper-parameter training. Soups-based methods [50, 41] aggregate diverse models for better generalizability. Some methods induce diversity through high learning rates [32], cosine similarity minimization [49], tempered posteriors [19], or auxiliary dataset-trained model soups [40]. We depict the difference of different model ensemble-based methods in our appendix 7.

3 Method

The structure of this Section is as follows: firstly, we provide the problem definition and corresponding notions to be used (Sec. 3.1); secondly, we reveal the dilemma for existing federated learning methods on reducing communication rounds (Sec. 3.2); finally, we propose a regularized model interpolation-based method as a solution, provide corresponding analysis (Sec. 3.3), and present the overall algorithm flow.

3.1 Notions and Problem Definition

Notions. Let $\mathcal{X}$ be the input space of data, $\mathcal{Y}$ be the label space. Consider a FL setting with M clients, τ local steps and R communication rounds. Let $\mathcal{D} := \{\mathcal{D}_i\}_{i=1}^{M}$ be a set of M domain, each of which is a distribution over the input space $\mathcal{X}$. For each client, we have access to n training data points in the form of $(\mathcal{X}_i, \mathcal{Y}_i) = \{(x_j^i, y_j^i)\}_{j=1}^{n}$, where y_j^i denotes the target label for input x_j^i. Let $f \in \mathbb{R}^m$ represents the parameter for the global model, $\ell_i : \mathbb{R}^m \to \mathbb{R}$ denotes the local objective function at client i, and $\mathcal{P}$ denotes a distribution on the entire set of clients. We provide a notation table in Appendix A to clarify the meanings of the corresponding notations.

Problem definition. We aim to address the challenge of optimizing the global performance on $\mathcal{D}$ of aggregated models fine-tuned from different clients with data heterogeneity, while minimizing communication rounds between the clients and the server in data Non-IID setting. In terms of global performance, we perform empirical risk minimization (ERM) on the sampled data $\mathcal{D}_i$ for $i \in [M]$,

$$\mathcal{L}(f) = \sum_{i=1}^{M} p_i \mathcal{L}_i(f), \quad \text{where } \mathcal{L}_i(f) = \frac{1}{|\mathcal{D}_i|} \sum_{\xi \in \mathcal{D}_i} \ell_i(f, \xi) \text{ and } \sum_{i=1}^{M} p_i = 1. \tag{1}$$

3.2 Effect of Regularization and Local Steps on FL Convergence under Data Heterogeneity

In this section, we present a theoretical analysis to understand how communication rounds, local steps, and our introduced regularization terms affect the convergence bound in federated learning. Formally, we present the error term and posit the following assumptions for the purpose of analysis. Our analysis follows the assumptions and the convergence bound in [48], which is restated as follows. Formal statements are detailed in Appedinx B.1.

Theorem 3.1 (Convergence Rate for Convex Local Functions with Affinity and Diversity Constraint)**.** *Under Convexity and Smoothness Assumption on β-smooth loss function, Bounded Variance of Stochastic Gradient and Bounded Variance of Local and Global Gradient assumptions, when the client learning rate is chosen properly as, $\eta = \min\{\frac{1}{4\beta}, \frac{M^{\frac{1}{2}} d}{\tau^{\frac{1}{2}} R^{\frac{1}{2}} \sigma}, \frac{d^{\frac{2}{3}}}{\tau^{\frac{2}{3}} R^{\frac{1}{3}} \beta^{\frac{1}{3}} \sigma^{\frac{2}{3}}}, \frac{d^{\frac{2}{3}}}{\tau R^{\frac{1}{3}} \beta^{\frac{1}{3}} (\zeta + c)^{\frac{2}{3}}}\}$ we define $\epsilon = \mathbb{E}\left[\frac{1}{\tau R} \sum_{r=0}^{R-1} \sum_{k=1}^{\tau} \beta(\overline{f}^{(r,k)}) - \beta(f^\star)\right]$, and have*

$$\epsilon \leq \frac{2\beta R^2}{\tau R} + \frac{2\sigma d}{\sqrt{M\tau R}} + \frac{5\beta^{\frac{1}{3}} \sigma^{\frac{2}{3}} d^{\frac{4}{3}}}{\tau^{\frac{1}{3}} R^{\frac{2}{3}}} + \frac{15\beta^{\frac{1}{3}} (\zeta + c)^{\frac{2}{3}} d^{\frac{4}{3}}}{R^{\frac{2}{3}}} \tag{2}$$

Here, the update rule of the t iteration with the affinity and diversity term is defined as $\theta(t+1) = \theta(t) - \eta g(t) - q(t, \mu_t, \mu_a)$, and the extra term satisfies $q(t, \mu_t, \mu_a) \leq c$. The hyper-parameters μ_t and μ_a represent the co-efficient of tuning affinity and diversity respectively.

Besides, $d := \|f^{(0,0)} - f^\star\|$ refers to the distance between initialization $f^{(0,0)}$ and the global optimum $f^\star$, σ bounds variance of stochastic gradient by $\mathbb{E}[\|g_i(f^{(r,k)}) - \nabla \mathcal{L}_i(f^{(r,k)})\|^2 | f^{(r,k)}] \leq \sigma^2$, and ζ bounds variance of local and global gradient by $\max_i \sup_f \left\|\nabla \mathcal{L}_i(f^{(r,k)}) - \nabla \mathcal{L}(f^{(r,k)})\right\| \leq \zeta$.

How to reduce communication rounds under data heterogeneity? Increasing local fine-tuning steps seems to be a straightforward technique to reduce communication costs. Nevertheless, this approach cannot reduce the an error term in the convergence rate (see the 4th term of the RHS of Eq. 2), which remains unaltered by increasing local steps. Moreover, increasing local update steps in the presence of Non-IID client data exacerbates the inconsistency in local objectives, further magnifying this error term. Here, we provide a more detailed explanation, specifically identifying the conditions under which increasing iteration steps can effectively reduce communication rounds.

Proposition 3.2. *Under the data heterogeneity setting, when the total number of gradient computations across all clients ($K = M\tau R$) is fixed and the local steps τ satisfies*

$$\tau \leq \frac{\sigma}{\zeta + c} \sqrt{\frac{\sigma}{d\beta} \frac{K^{\frac{1}{2}}}{M^2}}, \tag{3}$$

the error upper bound Eq.2 will be dominated by the second term $\mathcal{O}(1/\sqrt{K})$.

We provide the proof for Proposition 3.2 in Appendix B.2. Accordingly, increasing the bound in Eq. 2 and meeting the aforementioned condition for local steps allows us to reduce communication rounds. From the above in-equation, we can observe that although increasing the number of local training steps can reduce communication rounds, there is a limit to the number of steps that can be added. This limit is primarily determined by the error term introduced by local updates.

Why connecting low-loss valley in local training with pre-trained initialization can achieve extreme communication rounds reduction? Our analysis indicates that for substantial communication efficiency in federated learning, it is not enough to just increase local training steps. The focus should be on minimizing the error term from local updates, particularly the last term in Formula 2. This term, influenced by gradient dissimilarity (ζ), distance to optimal weights (d), and our proposed regularization update bound c, remains significant even as training steps increase.

Prior research suggests [35] that pre-training initialization reduces ζ by aligning client updates, and overparameterization ensures that the optimal parameters are typically close to the initialization [21, 5, 30], decreasing d. It is important to note that the regularization related term c is influenced by a combination of model diversity and affinity, and can be reduced by adjusting the parameters μ_t and μ_a. In the absence of a common pre-trained initialization, ensuring model affinity within the model pool is often challenging (*i.e.*, models from different clients tend to diverge significantly from the initialization values), resulting in a larger value of c, which in turn affects the effectiveness of our method. Therefore, our approach is more suitable when combined with pre-trained models. Consequently, a combination of pre-training and our proposed connectivity preserving local training can effectively lower error terms from local updates, increasing the limit of local training steps and thus reducing communication rounds. More experimental support see our Appendix.

3.3 Our Solution: *LSS* Algorithm

In this part, we first present the shortcomings of the previous model soups method applied in FL. Secondly, we propose our three targeted improvements, *i.e.* ***random model interpolation*** (Sec. 3.3.1), ***diversity term*** (Sec. 3.3.2), and ***affinity regularization term*** (Sec. 3.3.3). Finally, we present the complete algorithm process and detailed implementation in local client training.

Algorithm 1 *LSS* (Local Training) Pseudo-code

Require: f_p pre-trained model ($R = 1$) or global aggregated model ($R > 1$); $\mathcal{L}$ loss function; $\mathcal{D}$ dataset; $dist$ distance function; τ iteration steps; η learning rate; λ_a affinity coefficient; λ_d diversity coefficient; n number of averaged models.

LSS Local Training :
$\mathcal{M} \leftarrow \{f_p\}$
for $p_i = 1$ to N **do**
  $f_{p_i} \leftarrow Averaging(\mathcal{M})$
  $\mathcal{M} \leftarrow \mathcal{M} \cup \{f_{p_i}\}$ {sequential training with newly added model}
  for $t = 1$ to τ **do**
    $f_s \leftarrow RandomInterpolation(\mathcal{M})$ {connectivity preserving}
    $\mathcal{L}_{reg}(f_{p_i}) = \mathcal{L}(f_s, \mathcal{D}) + \lambda_a \cdot dist(f_{p_i}, f_p) - \lambda_d \cdot dist(f_{p_i}, \mathcal{M})$
    $f_{p_i} \leftarrow f_{p_i} - \eta \nabla_{f_{p_i}} \mathcal{L}_{reg}(f_{p_i})$
  end for
end for
Inference:
$f \leftarrow Averaging(\mathcal{M})$

Limitation of previous model soups methods. Previous model soups methods [50] can induce a trained model located in a connected low-loss valley, but their training efficiency is exceedingly low, due to two aspects: *Time-Consuming model selection phase:* Firstly, these methods involve a time-consuming model selection phase, which consists of all candidate models available for weight interpolation [3, 50]. This phase aims to choose models that reside within the same low-loss valleys. During this selection process, significant computational resources are consumed to identify suitable models for interpolation, adding to the overall training time and complexity. *Extensive and redundant model training:* Secondly, model soups entail an extensive number of model training iterations, lacking prior guidance and relying on brute-force, random, and often redundant training [28, 50]. Many of the trained models end up unused, further exacerbating the computational inefficiency.

3.3.1 Random interpolation conserving connected low-loss region.

To address the *time-consuming model selection* issue of the previous soups-based method, we propose a sequential random model interpolation method during training. This innovative approach streamlines the training process by eliminating the need for subsequent model selection steps within

the **model pool** (*i.e.*, local models to be interpolated), which traditionally consumes a considerable amount of computational resources and time. Let $\mathcal{M} = \{f_{p_1}, f_{p_2}, \ldots, f_{p_N}\}$ be a pool of N models, where f_{p_i} represents the weights of the i-th model. We define the interpolated model f_{interp} as a weighted combination of the models in $\mathcal{M}$. The interpolation coefficients $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_N)$ are sampled using a uniform distribution and normalization strategy. The interpolated model f_{interp} is then computed as: $f_{\text{interp}} = \sum_{i=1}^{N} \alpha_i f_{p_i}$. Here, α_i represents the weight assigned to the i-th model in the pool. The uniform distribution ensures that the coefficients α_i are non-negative and sum to 1, providing a simple and effective way to combine the model weights from the pool $\mathcal{M}$. Forward and backward propagation are performed using the interpolated model, updating the weights of the currently active model (*i.e.*, the newly added model) (corresponding to Algorithm 1 Line 7), while previously added model weights remain fixed.

3.3.2 Diversity term.

The diversity term is proposed to address the *redundant model training* issue of the previous soups-based methods by encouraging low-loss region expansion. In particular, the diversity indicator assesses the variability among models within the model pool by summing the distances between pairs of models. Greater distances between models indicate a higher degree of diversity, which correlates with enhanced model quality. This diversity metric is integrated into the FL local training process as a regularization term to facilitate the extensive enlargement of low-loss regions, consequently maximizing the effectiveness of trained models. The diversity term (in Algorithm 1 Line 8) measures the distance between the current training model and other models that will be averaged, and we hope that this distance to be large. The diversity loss can be defined as

$$\ell_{\text{diversity}} = \textit{dist}(f, \mathcal{M}) = \frac{1}{N} \sum_{n=1}^{N} \textit{dist}(f, f_n). \tag{4}$$

Here, f_n belongs to local interpolated model pool $\mathcal{M}$ and N is the number of local candidate models. The candidate models (*i.e.*, model soups ingredients) are models to be interpolated in local training, and the model pool is the set of local candidate models (see Algorithm 1 Line 5). We use the ℓ_2 norm to measure the distance between model weights.

3.3.3 Affinity term.

The affinity term is proposed to control the expansion of low-loss regions and prevent local candidate model training divergence. The affinity indicator assesses the level of alignment between each candidate model within the model pool and the initial global model by calculating the cumulative distances between each candidate model and the initialization model. Smaller distances between models signify a stronger affinity, indicating higher model quality. To ensure the controlled expansion of low-loss regions and reduce the probability of overlapping connected regions, this affinity metric is integrated into the training process as a regularization term. The affinity term (in Algorithm 1 Line 8) measures the distance between the candidate model and the initial model weights, with the aim of minimizing this dissimilarity (maximize this loss term) to ensure that the distance remains relatively small. The affinity loss can be defined as

$$\ell_{\text{affinity}} = \textit{dist}(f, f_p). \tag{5}$$

Here, f_p is a pre-trained model in the first communication round ($R = 1$). Moreover, it encourages each local model to lie in a close zone in the parameter space, which is beneficial for subsequent server aggregation, especially under data heterogeneity. We use l_2 distance for the $dist(,)$ metric for both Eq. 4 and Eq. 5.

3.3.4 Overall pipeline.

We outline *LSS* as follows: We begin with the initialization of the client's local model with the pretrained global model. Then we will refine the local model using affinity and diversity loss. This step is performed for a few local update steps. Finally, after updating local model, we aggregate them in the server following the common averaging operation in FedAvg [33]. The flow of *LSS* for local updating (Step 2 described in Sec 3.1) can be found in Algorithm 1.

In conclusion, our method aims to minimize the distance between the local fine-tuned model and the pre-trained initialized global model while maximizing the distance between the model soups

Table 1: Label shift test accuracy after $R = 1$ and $R = 3$ communication rounds. We primarily compared two categories of methods: conventional FL methods and state-of-the-art local weight averaging-based fine-tuning methods that enhance domain generalization.

	FMNIST		**CIFAR-10**	
Method	Accuracy ($R = 1$) ↑	Accuracy ($R = 3$) ↑	Accuracy ($R = 1$) ↑	Accuracy ($R = 3$) ↑
FedAvg [33]	35.54(1.71)	90.04(0.32)	58.34(0.86)	66.74(0.76)
FedProx [27]	33.48(1.52)	89.28(0.36)	56.74(0.92)	63.21(0.83)
MOON [26]	36.01(1.66)	91.28(0.30)	58.96(1.24)	67.04(1.12)
FedBN [29]	34.20(1.73)	89.87(0.47)	57.04(0.75)	64.51(0.67)
FedFomo [56]	33.94(1.65)	88.41(0.69)	55.01(0.89)	62.69(0.75)
FedRep [7]	36.20(1.52)	91.07(0.23)	57.73(0.82)	66.23(0.73)
FedBABU [36]	36.18(1.43)	91.31(0.26)	60.14(1.06)	67.16(0.87)
SWA [20]	55.82(1.02)	91.03(0.19)	59.07(1.28)	67.45(1.15)
SWAD [1]	58.66(0.87)	91.22(0.16)	60.54(1.15)	67.65(0.97)
Soups [50]	60.11(0.64)	91.56(0.24)	61.00(1.04)	67.63(0.94)
DiWA [41]	63.21(0.54)	91.88(0.13)	61.32(1.26)	68.05(1.10)
LSS (4x Mem.)	**72.66**(0.73)	**92.45**(0.21)	**65.96**(1.50)	**75.16**(1.07)

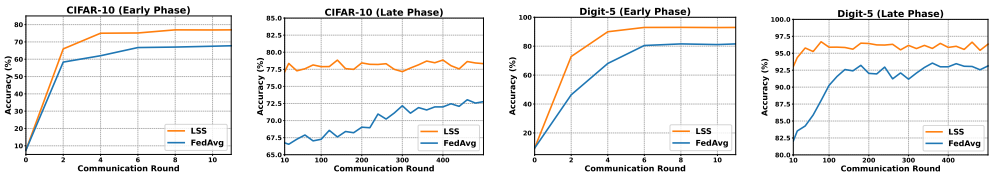
Table 2: Feature shift test accuracy after $R = 1$ and $R = 3$ communication rounds. *LSS* consistently outperforms other methods on both datasets across under feature shift settings.

	Digit-5		**DomainNet**	
Method	Accuracy ($R = 1$) ↑	Accuracy ($R = 3$) ↑	Accuracy ($R = 1$) ↑	Accuracy ($R = 3$) ↑
FedAvg [33]	46.36(2.08)	80.48(0.81)	18.76(3.52)	29.43(2.01)
FedProx [27]	44.01(1.92)	77.83(0.68)	17.27(3.22)	27.18(2.29)
MOON [26]	50.11(1.72)	83.02(0.64)	19.61(3.54)	31.27(2.34)
FedBN [29]	46.02(1.93)	81.42(0.71)	18.16(3.09)	28.65(1.89)
FedFomo [56]	41.87(2.13)	76.21(0.98)	15.10(3.82)	25.69(2.38)
FedRep [7]	47.43(1.73)	82.02(0.63)	18.89(2.60)	30.42(1.84)
FedBABU [36]	48.02(1.81)	83.20(0.79)	19.44(2.43)	32.06(1.88)
SWA [20]	54.13(0.72)	85.33(0.62)	22.07(2.55)	35.90(1.61)
SWAD [1]	57.02(0.71)	86.84(0.64)	21.98(2.61)	36.73(1.57)
Soups [50]	59.71(0.82)	87.07(0.58)	22.75(2.85)	38.02(1.40)
DiWA [41]	61.54(0.83)	88.83(0.69)	24.88(2.54)	38.32(1.50)
LSS (4x Mem.)	**72.86**(1.64)	**92.97**(0.65)	**27.86**(2.85)	**41.35**(1.46)

ingredients (*i.e.*, the models to be averaged). Our fine-tuned models find large low-loss regions on their respective local datasets while ensuring parameters close to the pre-trained initialization. It is intuitive that the parameters of our fine-tuned models can be more easily aligned with those of models fine-tuned on similar datasets, thereby improving communication efficiency.

4 Experiment

4.1 Experimental Setup

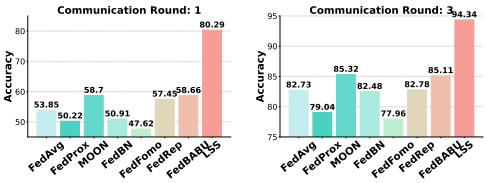


(a) (b) (c) (d)

Figure 3: Convergence comparison of our proposed *LSS* with FedAvg. *LSS* achieves high accuracy much earlier (around 6 to 8 rounds) than FedAvg, which takes hundreds of communication rounds.

Dataset. Our experimental section considers two scenarios of Non-IID settings, namely label shift and feature shift. The label shift scenario investigates datasets such as FMNIST [51] and CIFAR10 [25], while feature shift involves Digit5 and DomainNet. Further information on the specific datasets can be found in the appendix. In the label shift scenario, we partitioned the dataset into five clients and the data for each client are sampled following Dirichlet distributions with coefficient $\alpha = 1.0$, yielding imbalanced label distributions. In the feature shift scenario, we utilized five clients for Digit5 [12, 29] and five clients for DomainNet [38]. Additional results on an extended number of clients are presented in the appendix.

Model. In terms of models, we used the ImageNet pre-trained ResNet50 [15] as the base model for the DomainNet dataset, while for other datasets, we used the pre-trained ResNet-18 trained on ImageNet [8]. We also present the experimental results based on the vision transformer (ViT) model [10] with parameter-efficient fine-tuning.

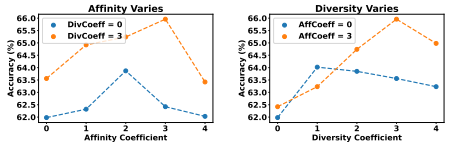


(a) (b)

Figure 4: Evaluation on ViT fine-tuned with LoRA (Digit5 dataset).

Baselines. We compare *LSS* against the vanilla FL method - FedAvg [33] and several advanced FL algorithms designed for Non-IID settings, including FedProx [27], MOON [26], FedBN [29], FedFomo [56], FedRep [7] and FedBABU [36]. Additionally, we make comparisons with top-performing weight/model-averaging-based domain generalization methods including SWA [20], SWAD [1], Soups [50] and DiWA [41] by adapting them to FL. In particular, the specific approach is to modify the local client training in the FedAvg framework to a corresponding fine-tuning approach. For more details, please refer to the appendix.

Evaluation and implementation details. Unless otherwise specified, the model performance in the experiments below refers to the global model performance after aggregation on the server side. Our training optimizer uses the Adam optimizer with a learning rate of $5\mathrm{e}{-4}$ and a training batch size of 64. For commonly used FL methods, due to the significant increase in local update steps that leads to worse convergence, we set their local update steps to 8. For SWA, SWAD, and our method, we take more local update steps, with each model being averaged trained 8 steps, and the default number of models to be averaged is 4. For the Model Soups method and DiWA, we train 32 models with 8 steps. Additional details of experiment implementations are included in the Appendix.



(a) (b)

Figure 5: Ablation on the affinity & diversity term.

4.2 Performance Comparison

Results on label shift. To demonstrate the effectiveness of *LSS* on label shift scenario, we conduct comparison experiments on FMNIST and CIFAR-10 datasets. We consider fine-tuning with an extremely limited number of communication rounds (*i.e.*, $R = 1$ and $R = 3$). Table 1 reports the test accuracy with the format of mean (std) for all compared algorithms. All experiments are repeated 3 runs with different random seeds. In Table 1, *LSS* achieves the best accuracy on all settings of both datasets, which validates that *LSS* is efficient and effective in fine-tuning FL for label shift Non-IID. Notably, with just one round of communication, *LSS* can double the accuracy of the best Non-IID FL baseline method. Surprisingly, the simple extension of model-averaging-based domain generalization methods onto FedAvg [33] (the 2nd big row in Table 1) perform very well, especially when the number communication round is small. The superior performance using local weight averaging-based fine-tuning is likely because it significantly reduces the gradient variance of local and global variance (see 3.2). We further provide results on different levels of label shift in the supplementary material.

Results on feature shift. Table 2 evaluates on feature shift scenario using Digits-5 and DomainNet datasets. Similar to the previous experiment setting for Table 1, we repeat all the algorithms with 3 random seeds. Consistent with the observation in Table 2, *LSS* is the top-performing method under all the settings for both datasets. We also observe better performance achieved by adapting model-averaging-based domain generalization methods (the 2nd big row in Table 2) in FL than the existing Non-IID FL methods (the 1st big row in Table 2), which further verifies the effectiveness of model averaging to obtain better global model while improving communication efficiency.

Convergence plots. We also evaluate the strength of *faster convergence* using the proposed *LSS* compared with FedAvg [33] on CIFAR-10 (label shift) and Digtis-5 (feature shift). Fig. 3 depicts the testing accuracies at early and late phases regarding the number of communication rounds to reach convergence. First, by looking at the final testing accuracies on Fig. 3 (b) and (d), *LSS* achieves better performance. Second, Fig. 3 (a) and (c) show that *LSS* almost meets the targeted performance at the

very early stage (*i.e.*around 6 to 8 rounds), whereas FedAvg requests over hundreds of communication rounds.

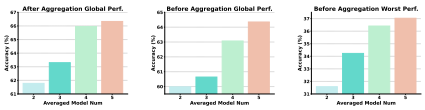
Parameter-Efficient Tuning with ViT. We also deployed the Vision Transformer (ViT) [10] in FL learning. On Digits-5 dataset, we evaluate the ViT model with a resolution of 224 and a patch size of 16, which was pretrained on the ImageNet-21k dataset. Due to the large number of parameters in ViT, we used a parameter-efficient fine-tuning method called LoRA [18] to train it for all the methods. For more details about our ViT architecture and LoRA training, please refer to the appendix. It can be observed in Fig. 4 that our method is applicable to pre-trained ViT models, demonstrating that our approach can be combined with parameter-efficient fine-tuning methods to further enhance the communication efficiency of FL.

4.3 Ablation Studies

We conducted ablation experiments on the main components (*i.e.*, affinity, diversity term and averaged model quantity) of our proposed method and evaluated their performance on the CIFAR dataset, with the performance metric being the global model performance at communication round $R = 1$.

Investigation on regularization losses. In order to examine the importance of affinity loss and diversity loss, as well as the influence of their corresponding coefficients, we adjust one coefficient within a range of 0 to 4 while maintaining the other at a constant value. By comparing the performance with and without loss term, we observe that adding affinity and diversity terms can enhance the model's performance. Additionally, we observe that the two terms complement each other, and selecting appropriate coefficients can achieve significant performance improvement (*e.g.*, adjusting the affinity coefficient to 3 as shown in Fig. 5 (a) and diversity coefficient to 3 as shown in Fig. 5 (b)).

Investigation on the number of averaged models. To investigate the impact of the averaged model quantity on enhancing communication efficiency and reducing gradient variance between local and global, we experiment with varied model quantities and evaluate their influence on global model performance, averaged local model performance[2], and worst out-of-distribution (OOD) generalization performance on the other clients. Fig. 6 shows that increasing the number of averaged models can improve the model's OOD generalization ability and enhance the performance of the aggregated model. This similar upward trend confirms the validity of our analysis linking OOD generalization and local-global variance. We provide a more detailed analysis on connecting our proposed *LSS* and OOD generalization in appendix C. Additionally, we can observe that increasing the number of models in our method can improve both pre-aggregation and post-aggregation model performance.



(a) (b) (c)

Figure 6: Ablation studies on the impact of the number of averaged models on communication efficiency and performance variance. We evaluated the influence of varied model quantities on global and averaged local model performance, as well as generalization on the worst client.

5 Conclusion

Limitations and Broader Impact. Our method reduces communication rounds but trades off training memory and performance. Future work should explore more memory-efficient deployments. While focused on vision tasks, extending to language and multimodal scenarios is promising. Balancing performance and communication in healthcare FL is promising, but excessive reduction can impair critical medical decisions. Careful trade-off consideration is essential for reliable FL applications in sensitive areas.

Conclusion. We propose an efficient method, Local Superior Soups (*LSS*), to reduce communication rounds in FL with pre-trained initialization, addressing the challenge of data heterogeneity. By employing sequential model interpolation, connectivity preservation, and two regularization terms (diversity and affinity), the method allows for an increase in local training steps and a reduction in communication rounds while avoiding client drift. This approach, tailored for pre-trained model adaptation in FL, offers training and inference efficiency, making it suitable for practical deployment

[2]the average performance of local models of individual clients before aggregation on the overall client dataset

in edge computing scenarios. As the first step towards understanding and developing model soups-based methods in pre-trained models in FL, this study conducts experiments on benchmark datasets. Our method attain superior performance with a only few rounds of communication and surpasses the performance of standard FL methods significantly across four datasets and under two distribution shift scenarios.

References

[1] Junbum Cha, Sanghyuk Chun, Kyungjae Lee, Han-Cheol Cho, Seunghyun Park, Yunsung Lee, and Sungrae Park. SWAD: domain generalization by seeking flat minima. In *NeurIPS*, pages 22405–22418, 2021.

[2] Hong-You Chen, Cheng-Hao Tu, Ziwei Li, Han-Wei Shen, and Wei-Lun Chao. On the importance and applicability of pre-training for federated learning, 2022.

[3] Minghui Chen, Meirui Jiang, Qi Dou, Zehua Wang, and Xiaoxiao Li. Fedsoup: Improving generalization and personalization in federated learning via selective model interpolation. In *MICCAI (2)*, volume 14221 of *Lecture Notes in Computer Science*, pages 318–328. Springer, 2023.

[4] Gary Cheng, Karan N. Chadha, and John C. Duchi. Fine-tuning is fine in federated learning. *CoRR*, abs/2108.07313, 2021.

[5] Lénaïc Chizat, Edouard Oyallon, and Francis R. Bach. On lazy training in differentiable programming. In *NeurIPS*, pages 2933–2943, 2019.

[6] Leshem Choshen, Elad Venezian, Shachar Don-Yehiya, Noam Slonim, and Yoav Katz. Where to start? analyzing the potential value of intermediate models. *CoRR*, abs/2211.00107, 2022.

[7] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pages 2089–2099. PMLR, 2021.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. IEEE Computer Society, 2009.

[9] Li Deng. The MNIST database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Process. Mag.*, 29(6):141–142, 2012.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*. OpenReview.net, 2021.

[11] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M. Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pages 3259–3269. PMLR, 2020.

[12] Yaroslav Ganin and Victor S. Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 1180–1189. JMLR.org, 2015.

[13] Neel Guha, Ameet Talwalkar, and Virginia Smith. One-shot federated learning. *CoRR*, abs/1902.11175, 2019.

[14] Kaiming He, Ross B. Girshick, and Piotr Dollár. Rethinking imagenet pre-training. In *ICCV*, pages 4917–4926. IEEE, 2019.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778. IEEE Computer Society, 2016.

[16] Parikshit Hegde, Gustavo de Veciana, and Aryan Mokhtari. Network adaptive federated learning: Congestion and lossy compression. In *INFOCOM*, pages 1–10. IEEE, 2023.

[17] Clare Elizabeth Heinbaugh, Emilio Luz-Ricca, and Huajie Shao. Data-free one-shot federated learning under very high statistical heterogeneity. In *The Eleventh International Conference on Learning Representations*, 2023.

[18] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *ICLR*. OpenReview.net, 2022.

[19] Pavel Izmailov, Wesley J. Maddox, Polina Kirichenko, Timur Garipov, Dmitry P. Vetrov, and Andrew Gordon Wilson. Subspace inference for bayesian deep learning. In *UAI*, volume 115 of *Proceedings of Machine Learning Research*, pages 1169–1179. AUAI Press, 2019.

[20] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry P. Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. In *UAI*, pages 876–885. AUAI Press, 2018.

[21] Arthur Jacot, Clément Hongler, and Franck Gabriel. Neural tangent kernel: Convergence and generalization in neural networks. In *NeurIPS*, pages 8580–8589, 2018.

[22] Jean Kaddour, Linqing Liu, Ricardo Silva, and Matt J. Kusner. Questions for flat-minima optimization of modern neural networks. *CoRR*, abs/2202.00661, 2022.

[23] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *CoRR*, abs/2001.08361, 2020.

[24] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank J. Reddi, Sebastian U. Stich, and Ananda Theertha Suresh. SCAFFOLD: stochastic controlled averaging for federated learning. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pages 5132–5143. PMLR, 2020.

[25] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *corr*, 2009.

[26] Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. In *CVPR*, pages 10713–10722. Computer Vision Foundation / IEEE, 2021.

[27] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In *MLSys*. mlsys.org, 2020.

[28] Weishi Li, Yong Peng, Miao Zhang, Liang Ding, Han Hu, and Li Shen. Deep model fusion: A survey. *CoRR*, abs/2309.15698, 2023.

[29] Xiaoxiao Li, Meirui Jiang, Xiaofei Zhang, Michael Kamp, and Qi Dou. Fedbn: Federated learning on non-iid features via local batch normalization. In *ICLR*. OpenReview.net, 2021.

[30] Xinyan Li and Arindam Banerjee. Experiments with rich regime training for deep learning. *CoRR*, abs/2102.13522, 2021.

[31] Bing Luo, Xiang Li, Shiqiang Wang, Jianwei Huang, and Leandros Tassiulas. Cost-effective federated learning in mobile edge networks. *IEEE J. Sel. Areas Commun.*, 39(12):3606–3621, 2021.

[32] Wesley J. Maddox, Pavel Izmailov, Timur Garipov, Dmitry P. Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. In *NeurIPS*, pages 13132–13143, 2019.

[33] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR, 2017.

[34] Vaishnavh Nagarajan and J. Zico Kolter. Uniform convergence may be unable to explain generalization in deep learning. In *NeurIPS*, pages 11611–11622, 2019.

[35] John Nguyen, Jianyu Wang, Kshitiz Malik, Maziar Sanjabi, and Michael Rabbat. Where to begin? on the impact of pre-training and initialization in federated learning. *CoRR*, abs/2210.08090, 2022.

[36] Jaehoon Oh, Sangmook Kim, and Se-Young Yun. Fedbabu: Towards enhanced representation for federated image classification. In *ICLR*. OpenReview.net, 2022.

[37] Younghyun Park, Dong-Jun Han, Do-Yeon Kim, Jun Seo, and Jaekyun Moon. Few-round learning for federated learning. In *NeurIPS*, pages 28612–28622, 2021.

[38] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *ICCV*, pages 1406–1415. IEEE, 2019.

[39] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, 2021.

[40] Alexandre Ramé, Kartik Ahuja, Jianyu Zhang, Matthieu Cord, Léon Bottou, and David Lopez-Paz. Recycling diverse models for out-of-distribution generalization. *CoRR*, abs/2212.10445, 2022.

[41] Alexandre Ramé, Matthieu Kirchmeyer, Thibaud Rahier, Alain Rakotomamonjy, Patrick Gallinari, and Matthieu Cord. Diverse weight averaging for out-of-distribution generalization. In *NeurIPS*, 2022.

[42] Shivalika Singh, Freddie Vargus, Daniel D'souza, Börje F. Karlsson, Abinaya Mahendiran, Wei-Yin Ko, Herumb Shandilya, Jay Patel, Deividas Mataciunas, Laura O'Mahony, Mike Zhang, Ramith Hettiarachchi, Joseph Wilson, Marina Machado, Luisa Souza Moura, Dominik Krzeminski, Hakimeh Fadaei, Irem Ergün, Ifeoma Okoh, Aisha Alaagib, Oshan Mudannayake, Zaid Alyafeai, Minh Chien Vu, Sebastian Ruder, Surya Guthikonda, Emad A. Alghamdi, Sebastian Gehrmann, Niklas Muennighoff, Max Bartolo, Julia Kreutzer, Ahmet Üstün, Marzieh Fadaee, and Sara Hooker. Aya dataset: An open-access collection for multilingual instruction tuning. *CoRR*, abs/2402.06619, 2024.

[43] Yan Sun, Li Shen, Tiansheng Huang, Liang Ding, and Dacheng Tao. Fedspeed: Larger local interval, less communication round, and higher generalization accuracy. In *ICLR*. OpenReview.net, 2023.

[44] Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. Towards personalized federated learning. *CoRR*, abs/2103.00710, 2021.

[45] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971, 2023.

[46] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288, 2023.

[47] Jianyu Wang, Zachary Charles, Zheng Xu, Gauri Joshi, H Brendan McMahan, Maruan Al-Shedivat, Galen Andrew, Salman Avestimehr, Katharine Daly, Deepesh Data, et al. A field guide to federated optimization. *arXiv preprint arXiv:2107.06917*, 2021.

[48] Jianyu Wang, Zachary Charles, Zheng Xu, Gauri Joshi, H. Brendan McMahan, Blaise Agüera y Arcas, Maruan Al-Shedivat, Galen Andrew, Salman Avestimehr, Katharine Daly, Deepesh Data, Suhas N. Diggavi, Hubert Eichner, Advait Gadhikar, Zachary Garrett, Antonious M. Girgis, Filip Hanzely, Andrew Hard, Chaoyang He, Samuel Horváth, Zhouyuan Huo, Alex Ingerman, Martin Jaggi, Tara Javidi, Peter Kairouz, Satyen Kale, Sai Praneeth Karimireddy, Jakub Konečný, Sanmi Koyejo, Tian Li, Luyang Liu, Mehryar Mohri, Hang Qi, Sashank J. Reddi, Peter Richtárik, Karan Singhal, Virginia Smith, Mahdi Soltanolkotabi, Weikang Song, Ananda Theertha Suresh, Sebastian U. Stich, Ameet Talwalkar, Hongyi Wang, Blake E. Woodworth, Shanshan Wu, Felix X. Yu, Honglin Yuan, Manzil Zaheer, Mi Zhang, Tong Zhang, Chunxiang Zheng, Chen Zhu, and Wennan Zhu. A field guide to federated optimization. *CoRR*, abs/2107.06917, 2021.

[49] Mitchell Wortsman, Maxwell Horton, Carlos Guestrin, Ali Farhadi, and Mohammad Rastegari. Learning neural network subspaces. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pages 11217–11227. PMLR, 2021.

[50] Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *ICML*, volume 162 of *Proceedings of Machine Learning Research*, pages 23965–23998. PMLR, 2022.

[51] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *CoRR*, abs/1708.07747, 2017.

[52] Zhewei Yao, Amir Gholami, Kurt Keutzer, and Michael W. Mahoney. Pyhessian: Neural networks through the lens of the hessian. In *IEEE BigData*, pages 581–590. IEEE, 2020.

[53] Rui Ye, Rui Ge, Xinyu Zhu, Jingyi Chai, Yaxin Du, Yang Liu, Yanfeng Wang, and Siheng Chen. Fedllm-bench: Realistic benchmarks for federated learning of large language models. *CoRR*, abs/2406.04845, 2024.

[54] Jie Zhang, Chen Chen, Bo Li, Lingjuan Lyu, Shuang Wu, Shouhong Ding, Chunhua Shen, and Chao Wu. DENSE: data-free one-shot federated learning. In *NeurIPS*, 2022.

[55] Lin Zhang, Li Shen, Liang Ding, Dacheng Tao, and Ling-Yu Duan. Fine-tuning global model via data-free knowledge distillation for non-iid federated learning. In *CVPR*, pages 10164–10173. IEEE, 2022.

[56] Michael Zhang, Karan Sapra, Sanja Fidler, Serena Yeung, and Jose M. Alvarez. Personalized federated learning with first order model optimization. In *ICLR*. OpenReview.net, 2021.

[57] Zhuo Zhang, Yuanhang Yang, Yong Dai, Qifan Wang, Yue Yu, Lizhen Qu, and Zenglin Xu. Fedpetuning: When federated learning meets the parameter-efficient tuning methods of pre-trained language models. In *ACL (Findings)*, pages 9963–9977. Association for Computational Linguistics, 2023.

[58] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. In *NeurIPS*, 2023.

[59] Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. In *NeurIPS*, pages 14747–14756, 2019.

Roadmap of Appendix The appendix is organized as follows. We list the notations table in Section A. We provide the theoretical proof of the convergence analysis in Section B. We present the theoretical intuition of our proposed two loss term C. Next, we provide more detailed related work in Sec. D We present more experiment details and results in Sec. E.

A Notation Table

Table 3: Important notations used in the paper.

Notations	Description
f	model parameters
l	learning procedure
m	feature dimension
n	number of samples
x	a sample
y	a label
$\mathcal{D}$	set of training domain
$\mathcal{L}$	loss function
M	number of clients
N	number of averaged models
R	total communication rounds
X	input space of data
Y	label space
λ	coeff. for local training reg. term
τ	local training steps

B Convergence Analysis

B.1 Formal Restatement of Convergence Theorem

Standard FL [33] employs a server to coordinate the following iterative distributed training:

Step 1 In each global round of training $r \in [R]$, the server broadcasts its current global model weight $f_g^{(r-1)}$ to all the clients;

Step 2 The selected client c copies the current server model weight $f_c^{r,0} \leftarrow f_g$, performs τ local step updates, then sends $f_c^{r,L} - f_g^{(r-1)}$ back to the server;

Step 3 The server aggregates the updates from all clients $\{f_c^{r,\tau} - f_g^{(r-1)}\}_{c=1}^{C}$ to form the new server model using the weighted averaging in Eq 1:

Note that the initialization $f^{(0,0)}$,with the subscription indicating model at 0-th communication round and 0-th local step, is a pre-trained model (*e.g.* using public datasets) in our problem. This work focus on improving *Step 2* to explore a larger low-loss region in local clients.

Formally, we present the convergence results (Theorem 1) and specify the following formal assumptions: 1) *Convexity and Smoothness Assumption* on β-smooth loss function, 2) *Bounded Variance of Stochastic Gradient Assumption* and 3) *Bounded Variance of Local and Global Gradient Assumption*).

Assumption B.1. (Convexity and Smoothness). $\mathcal{L}_i$ is convex and β-smooth for all $i \in [M]$, i.e.,

$$\|\nabla \mathcal{L}_i(w) - \nabla \mathcal{L}_i(v)\| \leq \beta \|w - v\|,$$

for all w, v in its domain and $i \in [M]$.

Assumption B.2. (Bounded variance of stochastic gradient). Each client can achieve an unbiased stochastic gradient with σ^2-*uniformly bounded variance* for all $k \in [0, \tau)$, namely

$$\mathbb{E}[g_i(f_i^{(r,k)})|f_i^{(r,k)}] = \nabla \mathcal{L}_i(f_i^{(r,k)}), \quad \mathbb{E}[\|g_i(f_i^{(r,k)}) - \nabla \mathcal{L}_i(f_i^{(r,k)})\|^2 | f_i^{(r,k)}] \leq \sigma^2. \tag{6}$$

Assumption B.3. (Bounded variance of local and global gradient). The difference of local gradient $\nabla\mathcal{L}_i(f)$ and the global gradient $\nabla\mathcal{L}(f)$ is bounded in ℓ_2 norm, namely

$$\max_i \sup_f \left\| \nabla\mathcal{L}_i(f_i^{(r,k)}) - \nabla\mathcal{L}(f_i^{(r,k)}) \right\| \leq \zeta. \tag{7}$$

Assumption B.4. (Bounded regularization update). The update introduced by the affinity and diversity regularization term $q(t, \mu_t, \mu_a)$ in the update rule $\theta(t+1) = \theta(t) - \eta g(t) - q(t, \mu_t, \mu_a)$, is bounded by a constant c, namely

$$q(t, \mu_t, \mu_a) \leq c. \tag{8}$$

We have the main theorem on convergence rate, which similar to [47] except for the introduced regularization terms.

Theorem 1: Convergence Rate for Convex Local Functions with Affinity and Diversity Constraint Under Convexity and Smoothness Assumption on β-smooth loss function, Bounded Variance of Stochastic Gradient and Bounded Variance of Local and Global Gradient assumptions, when the client learning rate is chosen properly as,

$$\eta = \min\{\frac{1}{4\beta}, \frac{M^{\frac{1}{2}}d}{\tau^{\frac{1}{2}}R^{\frac{1}{2}},\sigma}, \frac{d^{\frac{2}{3}}}{\tau^{\frac{2}{3}}R^{\frac{1}{3}}\beta^{\frac{1}{3}}\sigma^{\frac{2}{3}}}, \frac{d^{\frac{2}{3}}}{\tau R^{\frac{1}{3}}\beta^{\frac{1}{3}}(\zeta+c)^{\frac{2}{3}}}\} \tag{9}$$

we define $\epsilon = \mathbb{E}\left[\frac{1}{\tau R}\sum_{r=0}^{R-1}\sum_{k=1}^{\tau}\beta(\overline{f}^{(r,k)}) - \beta(f^\star)\right]$, and have

$$\epsilon \leq \frac{2\beta R^2}{\tau R} + \frac{2\sigma d}{\sqrt{M\tau R}} + \frac{5\beta^{\frac{1}{3}}\sigma^{\frac{2}{3}}d^{\frac{4}{3}}}{\tau^{\frac{1}{3}}R^{\frac{2}{3}}} + \frac{15\beta^{\frac{1}{3}}(\zeta+c)^{\frac{2}{3}}d^{\frac{4}{3}}}{R^{\frac{2}{3}}} \tag{10}$$

Here, the update rule of the t iteration with the affinity and diversity term is defined as $\theta(t+1) = \theta(t) - \eta g(t) - q(t, \mu_t, \mu_a)$, and the extra term satisfies $q(t, \mu_t, \mu_a) \leq c$. The hyper-parameters μ_t and μ_a represent the co-efficient of tuning affinity and diversity respectively.

Besides, $d := \|f^{(0,0)} - f^\star\|$ refers to the distance between initialization $f^{(0,0)}$ and the global optimum $f^\star$, σ bounds variance of stochastic gradient by $\mathbb{E}[\|g_i(f^{(r,k)}) - \nabla\mathcal{L}_i(f^{(r,k)})\|^2 | f^{(r,k)}] \leq \sigma^2$, and ζ bounds variance of local and global gradient by $\max_i \sup_f \left\|\nabla\mathcal{L}_i(f^{(r,k)}) - \nabla\mathcal{L}(f^{(r,k)})\right\| \leq \zeta$.

The regularization update bound is reasonable since $q(\mu_t, \mu_a) = \mu_t * (\theta - \theta_m) - \mu_a * (\theta - \theta_m)$. Here, θ_m is the averaged parameter of all the parameter in the model pool for interpolation. As the inherent trade-off effect of diversity and affinity term, $q(\mu_t, \mu_a)$ will not diverge too much in practice. And the the bounded value c can be effectively controlled by tuning hyper-parameter μ_a and μ_t

The distinguishing factor in our convergence rate, as compared to that in [47], stems from the unique inter-client update bound facilitated by our proposed regularization term. We have the inter-client (*e.g.*, for client index 1 and 2) update bound as

$$\|\nabla\mathcal{L}_1(f) - \nabla\mathcal{L}_2(f) - q_1(\mu_t, \mu_a) + q_2(\mu_t, \mu_a)\| \leq 4(\zeta^2 + c^2 + 2\zeta c) \tag{11}$$

Proof. To find a tight bound for $\|\nabla\mathcal{L}_1(f) - \nabla\mathcal{L}_2(f) - q_1(\mu_t, \mu_a) + q_2(\mu_t, \mu_a)\|$, we will use the given inequalities: $\left\|\nabla\mathcal{L}_i(f^{(r,k)}) - \nabla\mathcal{L}(f^{(r,k)})\right\| \leq \zeta$, *i.e.*,

$$\|\nabla\mathcal{L}_1(f) - \nabla\mathcal{L}(f))\| \leq \zeta,\ \|\nabla\mathcal{L}_2(f) - \nabla\mathcal{L}(f))\| \leq \zeta. \tag{12}$$

By breaking down the expression with inserting global gradient $\nabla\mathcal{L}$ and then apply the triangle inequality of absolute value, and our given inequalities, we have

$$\|\nabla\mathcal{L}_1(f) - \nabla\mathcal{L}_2(f)) - q_1(\mu_t, \mu_a) + q_2(\mu_t, \mu_a)\| \leq (2\zeta + 2c)^2. \tag{13}$$

Combined our derived inter-client update bound with the Equation 28 in Appendix D.2 in [47], we can easily obtain a different client update drift bound ϵ_c as follows:

$$\epsilon_c \leq 4\tau\eta^2\sigma^2 + 14\tau^2\eta^2(\zeta + c)^2 \tag{14}$$

Leveraging the above in-equation and choosing the learning rate η properly, we can get our Theorem 1.

B.2 Proof of Lemma 1

Lemma 1. Under the data heterogeneity setting, when the total number of gradient computations across all clients ($K = M\tau R$) is fixed and the local steps τ satisfies

$$\tau \leq \frac{\sigma}{\zeta + c}\sqrt{\frac{\sigma}{d\beta}\frac{K^{\frac{1}{2}}}{M^2}}, \tag{15}$$

the error upper bound Eq.equation 15 will be dominated by the second term $\mathcal{O}(1/\sqrt{K})$.

Taking local steps can save total communication rounds compared to synchronous SGD. To be more specific, as suggested in [47], when the total number of gradient evaluations/computations across all clients ($K = M\tau R$) is fixed and the local steps τ satisfies:

$$\tau \leq \min\left\{\frac{\sigma}{d\beta}\frac{K^{\frac{1}{2}}}{M^2}, \frac{\sigma}{\zeta + c}\sqrt{\frac{\sigma}{d\beta}\frac{K^{\frac{1}{2}}}{M^2}}\right\}. \tag{16}$$

When the upper bound of local steps (Eq.(16)) becomes larger, there will be more communication savings. Therefore, the quantity in Eq.(16) represents the largest savings in communication rounds. Next, we show the error upper bound under the data heterogeneity setting.

Proof. Under high data heterogeneity, we have $\zeta + c \geq \sigma$, and:

$$1 \leq \frac{\sigma}{\zeta + c}\sqrt{\frac{\sigma}{d\beta}\frac{K^{\frac{1}{2}}}{M^2}} \leq \sqrt{\frac{\sigma}{d\beta}\frac{K^{\frac{1}{2}}}{M^2}} \leq \frac{\sigma}{d\beta}\frac{K^{\frac{1}{2}}}{M^2} \tag{17}$$

Therefore, we have Lemma 1:

$$\tau \leq \frac{\sigma}{\zeta + c}\sqrt{\frac{\sigma}{d\beta}\frac{K^{\frac{1}{2}}}{M^2}}, \tag{18}$$

□

This Lemma 1 indicates that when client data are Non-IID, the side effects of the error term in the Theorem 1 will be further exacerbated, therefore, increasing the local iteration steps effectively reduces the communication rounds.

C Theoretical Intuitions.

C.1 Decomposition of Generalization Bound

Connecting ζ with out-of-distribution error. ensemble is a category of the promising method that ensembles trained models to improve generalizability as demonstrated in centralized settings via reducing model discrepancy [20]. To reduce the variance ζ of local and global gradients that is resulted by data heterogeneity, we aim to adapt ensemble to FL. Intuitively, local client training that can reduce the error on the worst domain (client) in FL will reduce the variance ζ.

In the following, we detail how to reduce ζ with OOD error with a bias-variance-covariance-locality (BVCL) decomposition analysis. ensemble can be defined as: $f_{\text{WA}} \triangleq 1/N\sum_{n=1}^{N} f_n$. We have the following decomposition of ensemble's expected test error. *Bias-variance-covariance-locality decomposition.* The expected generalization error on domain T of f_{WA} over the joint distribution ($L_S^N \triangleq \{l_S^{(N)}\}_{N=1}^N$) of N learning procedure on domain S is:

$$\mathbb{E}_{L_S^N}\mathcal{E}_T(f_{\text{WA}}(L_S^N)) = \mathbb{E}_{(x,y)\sim p_T}\left[\text{bias}^2(x,y) + \frac{1}{N}\text{var}(x) + \frac{N-1}{N}\text{cov}(x)\right] + O(\bar{\Delta}^2), \tag{19}$$

Here, cov refers to the covariance of predictions made by two member models. The first component is the same bias as that of each individual member. The variance of ensemble is split into two parts:

the variance of each member divided by the number of members (N) and a covariance term. The last locality term enforces constraints on the weights to ensure the functional ensembling approximation remains valid. In summary, combining N models reduces variance by a factor of N, but introduces the covariance and locality terms which must be controlled to ensure low OOD error.

In the analysis presented in [41], the authors proposed a BVCL decomposition based on the approximation of functional ensembling (i.e., averaged prediction instead of parameter) by WA. The expected generalization error on domain T of f_{WA} over the joint distribution ($L_S^N \triangleq \{l_S^{(N)}\}_{N=1}^N$) of N learning procedure on domain S is:

$$\mathbb{E}_{L_S^N}\mathcal{E}_T(f_{\text{WA}}(L_S^N)) = \mathbb{E}_{(x,y)\sim p_T}\left[\text{bias}^2(x,y) + \frac{1}{N}\text{var}(x) + \frac{N-1}{N}\text{cov}(x)\right] + O(\bar{\Delta}^2), \quad \text{(BVCL)}$$

Definition C.1 (Bias)**.** For $x \in X$ and $y \in Y$, we define the bias of OOD prediction as,

$$\text{bias}(x,y) = y - \mathbb{E}_{l_S}[f(x,l_S)]. \tag{20}$$

Definition C.2 (Variance)**.** For $x \in X$, we define the variance of prediction as

$$\text{var}(x) = \mathbb{E}_{f_S}\left[\left(f(x,l_S) - \mathbb{E}_{l_S}\left[f(x,l_S)\right]\right)^2\right]. \tag{21}$$

Definition C.3 (Covariance)**.** For $x \in X$, we define the covariance of prediction produced by two different learning procedures l_S and l'_S as

$$\text{cov}(x) = \mathbb{E}_{l_S,l'_S}\left[\left(f(x,l_S) - \mathbb{E}_{l_S}\left[f(x,l_S)\right]\right)\left(f(x,l'_S) - \mathbb{E}_{l_S}\left[f(x,l_S)\right]\right)\right]. \tag{22}$$

Definition C.4 (Locality)**.** For any averaged models f_i (for $i \in [N]$), i is the index of an averaged model, N is the total number of averaged models, we define the locality of all averaged models as

$$\bar{\Delta}^2 = \mathbb{E}_{L_S^N}\Delta^2_{L_S^N} \text{ with } \Delta_{L_S^N} = \max_{i=1}^{N}\left\|f_i - f_{\text{WA}}\right\|_2. \tag{23}$$

Following the definitions of the terms in the BCVL generalization bound, we discuss the insights of reducing the bound via the proposed strategy. Our method is based on WAFT, which enjoys the benefit of reducing prediction variance by averaging the predictions of multiple models. The diversity term in our proposed method reduces the covariance term by encouraging functional diversity in the parameter space. The affinity term in our proposed method reduces the locality term to ensure the approximation of weight averaging (WA) to prediction ensembling.

Analysis on variance. One can see that an increase in the number of averaged models can directly lead to a reduction in variance. The straightforward averaging M models, as seen in the vanilla WAFT method, diminishes variance by a factor of M. However, this approach also introduces covariance and locality terms, which necessitate meticulous management on adding new averaged models to guarantee minimal out-of-distribution (OOD) error.

Analysis on covariance. The covariance term represents the predictive covariance between two member models whose weights are averaged. It increases when the predictions of different averaged models are highly correlated. In the worst-case scenario where all predictions are identical, the covariance is equal to the variance, rendering the benefits of weight averaging ineffective [41]. Conversely, when the covariance is lower, the advantages of weight averaging over individual models become more pronounced. Therefore, it is crucial to address covariance by promoting functional diversity among the averaged models. Our proposed method incorporates a diversity term that aims to reduce this covariance.

Analysis on locality. The locality term, which represents the expected squared maximum distance between weights and their average, constrains the weights to be close and ensures the approximation. The affinity term in our proposed method encourages the reduction of this locality term.

Overall, to reduce WA's error in OOD, we need to seek a good trade-off between diversity and locality. Our solution achieves this balance through two optimizable loss terms, the diversity term, and the affinity term. Besides, the direct combination of M models, as in the vanilla WAFT method, reduces variance by a factor of M but introduces covariance and locality terms that need to be carefully managed in order to ensure low OOD error.

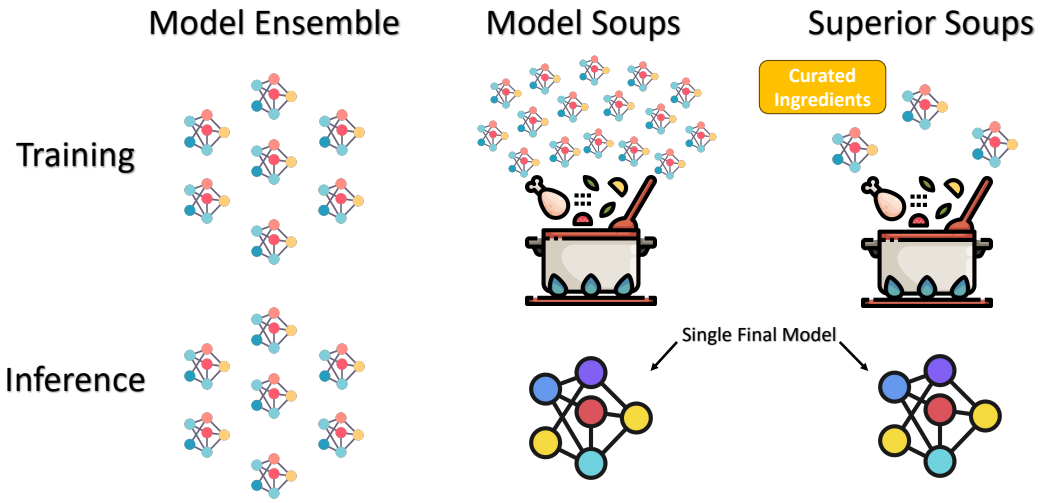


Figure 7: Comparison on model ensemble, model soups, and superior soups.

It is worth noting that, from an implementation perspective, unlike the model soups method (see Fig. 7 middle), which requires retraining a large number of candidate models for model selection and interpolation, our method only selects a few models (typically 3 to 5) for sequential random interpolation training in order to maintain connectivity. This significantly reduces the time cost of local training. Furthermore, unlike model ensembles (see Fig. 7) that require storing multiple model weights and integrating predictions during inference, our method only needs to retain an averaged weight during the final inference stage. This greatly reduces the memory footprint and enhances the inference speed on the client side.

D More Related Work

D.1 Heterogeneous Federated Learning

FL performance downgrading on Non-IID data is a critical challenge. A variety of FL algorithms have been proposed to handle this heterogeneous issue. From an optimization perspective: FedProx [27] adds L_2 norm to the client model and the previous server model to regularize them. This helps to prevent the client models from diverging too far from the server model. Scaffold [24] adds a variance reduction term to mitigate the "clients-drift." MOON [26] uses mode-level contrastive learning to stabilize local training by making the client models more robust to changes in the data distribution. In addition, personalized FL [44] is another approach to achieving high local testing performance on Non-IID data. For aggregation perspective: FedBN [29] uses local batch normalization to alleviate the feature shift before averaging models. For extreme communication efficient: In recent years, there have been some FL methods based on one-shot communication rounds. These methods typically use additional techniques on the server-side, such as using prediction ensembles [13] instead of weight ensembles or generating data [54, 17] from local models for centralized training, to improve the performance of the aggregated model. These methods are orthogonal to our client training-based approach. There are also works on few-round communication rounds in FL based on meta-learning frameworks [37], but the data partition used in the experimental setup may not be suitable for practical FL scenarios.

D.2 Fine-tuning and Model Interpolation

Fine-tuning aims to achieve improved performance on the given task by leveraging the learned knowledge of the pre-trained model. [6] empirically study the impact of fine-tuning from a pre-trained model in FL and unsurprisingly find that starting from a pre-trained model reduces the training time required to reach a target error rate and enables the training of more accurate models than starting from random initialization. [55] propose a knowledge distillation approach for fine-tuning the global model, called FedFTG. In addition, fine-tuning in FL has been widely used in personalized

FL to address Non-IID problems by having each user adapt the global model to personalized local models using their own data. For example, FedBABU [36] splits the model into body and head, then fine-tuning the head part for personalization. [4] propose FTFA and RTFA that start with a pre-trained model and then fine-tunes a small subset of model parameters using the FedAvg [33] algorithm. However, this line of work focuses on optimizing local performance and ignores the generalization of global data. This can lead to a performance drop when we further update the global model from the updated local models. Weight averaging and model recycling are not only efficient ways to aggregate machine learning models but also present promising benefits of improving model generalizability. Inspired by the linear mode connectivity property of neural networks trained with stochastic gradient descent (SGD) [34, 11], Model Soups [50] proposes to combine many independent runs with varied hyper-parameter configurations. Similarly, DiWA [41] utilizes this idea of Model Soups while theoretically analyzing the importance of training different models with diverse hyper-parameters within mild ranges. Soups-based methods [50, 41] rely on aggregating diverse models to improve model generalizability. To induce greater diversity, some methods such as [32] using a high constant learning rate, [49] minimizing cosine similarity between weights, [19] using a tempered posterior and model Ratatouille [40] averages diverse model trained from auxiliary datasets.

E Experiment Details

E.1 Experimental Setup Details

Dataset. We validate the effectiveness of our proposed method with four datasets, FMNIST [51], CIFAR-10 [25], Digit-5 [12, 29], and DomainNet [38]. The Fashion-MNIST (FMNIST) dataset is a dataset of Zalando's article images consisting of a training set of $60,000$ examples and a test set of $10,000$ examples. Each example is a 28×28 grayscale image of a piece of clothing. The dataset is divided into 10 classes: t-shirt/top, trouser, pullover, dress, coat, sandal, shirt, sneaker, bag, and ankle boot. The CIFAR-10 dataset is a popular dataset for machine learning research. It consists of $60,000$ 32×32 color images divided into 10 classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. The dataset is split into $50,000$ training images and $10,000$ test images. The Digit-5 dataset is a collection of five popular digit datasets, MNIST [9] (55000 samples), MNIST-M (55000 samples), Synthetic Digits [12] (25000 samples), SVHN (73257 samples), and USPS (7438 samples). Each digit dataset includes a different style of 0-9 digit images. The DomainNet dataset is a large-scale dataset of images collected from six different domains: clipart, infograph, painting, quickdraw, real, and sketch. The dataset contains $600,000$ images, each labeled with one of 345 object categories. The images in the DomainNet dataset are of high quality and are diverse in terms of their content and style.

Model. We used the pre-trained models from the timm repo [1], which are a collection of state-of-the-art deep learning models for computer vision tasks. For our proposed *LSS*, we use Adam optimizer with a learning rate of $5\mathrm{e}{-4}$, momentum 0.9, and weight decay $5\mathrm{e}{-4}$. The default number of averaged models is 4. Each model updates 8 epoch then aggregates with the others. The default affinity term coefficient is 3 and diversity term coefficient is 3. We set the batch size to 64 by default. For vision transformer (ViT) [10] model, we adopt ViT base model with 224×224 image size and 16×16 input patch size. The ViT is a neural network architecture for image classification that uses a self-attention mechanism to learn the relationships between pixels in an image. ViT has been shown to achieve state-of-the-art results on a variety of image classification benchmarks, including ImageNet and CIFAR-10.

Training Details. We implement all the methods in PyTorch, and we run all the experiments on an NVIDIA Tesla V100 GPU. Unless otherwise specified, the model performance in the experiments below refers to the global model performance after aggregation on the server side. For commonly used FL methods, due to the significant increase in local update steps that leads to worse convergence, we set their local update steps to 8.

Applying WAFT to FL Local Update. For SWA [20], SWAD [1], and our method *LSS*, we take more local update steps, with each model being averaged trained 8 steps, and the default number of models to be averaged is 4. For the Model Soups [50] method and DiWA [41], we trained 32 models and each model trained 8 steps. The hyper-parameter configuration for model selection

[1] https://github.com/huggingface/pytorch-image-models

includes learning rate ($[1e{-}4, 5e{-}4, 1e{-}5]$), batch size ($[32, 64, 128]$), dropout rate ($[0.0, 0.1, 0.3]$), and weight decay $[5e{-}4, 5e{-}5, 5e{-}6]$. Each run randomly select one of the hyper-parameter options. From each run of WAFT method, we take the weights of the epoch with maximum accuracy on the validation dataset, which follows the training distribution.

E.2 Extended Experiment Results

Arbitrarily increasing local steps cannot reduce communication rounds.

From Table 4, we can see that simply increasing local steps does not always lead to improved model performance. For FedAvg on the CIFAR10 dataset, increasing local steps beyond 8 actually results in a decrease in model performance.

Table 4: FedAvg with different local steps: Label shift test accuracy after $R = 1$ communication rounds (CIFAR-10 with 5 Clients).

Method	Accuracy ($\tau = 1$) ↑	Accuracy ($\tau = 4$) ↑	Accuracy ($\tau = 8$) ↑	Accuracy ($\tau = 12$) ↑	Accuracy ($\tau = 16$) ↑
FedAvg [33]	34.03(2.84)	49.08(1.51)	**58.34**(0.86)	55.76(0.82)	53.21(0.80)

Computational and memory costs comparison.

In Table 5, we provide detailed information on computational overhead and memory usage for various methods. Since the computational overhead and memory usage of FedAvg and other used FL methods are nearly identical, we only present the data for FedAvg here. Similarly, as the computational overhead and memory usage for SWA and SWAD, as well as for Soups and DiWA, are also nearly the same, we only show the data for SWA and Soups methods. It can be observed that our method requires more memory compared to other soups-based methods. However, the overall computational time for a single client's communication round is faster in our approach. This is because other soups-based methods require training a large number of models repeatedly to achieve good model performance. For instance, Soups needs to train 32 models, whereas our method only requires training 4 models. If the number of models trained by Soups is reduced to just 4, it only brings about a 5% improvement compared to FedAvg with a communication round of 1.

Table 5: Computational and memory costs of different types of method (ResNet-18).

Costs	FedAvg [33]	SWA [20]	Soups [50]	***LSS*** ($M = 2$)	***LSS*** ($M = 4$)
MACs (G)	1.82	1.82	1.82	2.73	4.55
Train Time Per Epoch (s)	2.66	2.73	2.66	12.27	20.43
Train Time Per Round (s)	21.28	433.31	683.52	100.98	169.77

***LSS* encourages smoothness (reducing β).** In Table 6, we provide the performance degradation of trained models evaluating under varying levels of random noise. Generally, a smaller performance degradation indicates a more robust model, which to some extent reflects the smoothness of the trained model. We can observe that our method exhibits greater robustness to noise perturbation.

Table 6: Smoothness of the trained model. Evaluated trained model performance drop on a testset with added ℓ_0 norm random noise. CIFAR-10 dataset Dirichlet distribution $\alpha = 1.0$ and $\alpha = 0.1$: Label shift test accuracy after $R = 1$

	CIFAR-10 ($4/255$)		**CIFAR-10** ($8/255$)	
Method	Accuracy ($R = 1$) ↓	Accuracy ($R = 3$) ↓	Accuracy ($R = 1$) ↓	Accuracy ($R = 3$) ↓
FedAvg [33]	1.30	1.17	3.06	2.93
LSS	0.89	0.76	2.37	1.85

***LSS* improves flatness of loss landscape.** The sharpness measure utilized in the Table 7 computes the median of the dominant Hessian eigenvalue across all training set batches through the Power Iteration algorithm [52]. This metric signifies the maximum curvature of the loss landscape, commonly employed in the literature on flat minima [22] to indicate sharpness. As demonstrated in the presented table, it is clear that our proposed method results in flatter minima compared to FedAvg.

Table 7: Loss landscape flatness quantification with Hessian eigenvalue.

	FedAvg ↓	*LSS* ($M = 2$) ↓	*LSS* ($M = 3$) ↓	*LSS* ($M = 4$) ↓
Hessian Eigenvalue	193.18	147.20	136.67	**119.14**

Evaluation with more clients. To assess the effectiveness of our method in larger-scale client scenarios, we conducted an expanded experiment involving 50 clients. From the Table 8, we can observe that our proposed method maintains a significant advantage across different client scales, particularly when the number of communication rounds is small ($R = 1$).

Table 8: Different client numbers (5 Clients and 50 Clients): Label shift test accuracy after $R = 1$ and $R = 3$ communication rounds.

	CIFAR-10 (5 Clients)		**CIFAR-10 (50 Clients)**	
Method	Accuracy ($R = 1$) ↑	Accuracy ($R = 3$) ↑	Accuracy ($R = 1$) ↑	Accuracy ($R = 3$) ↑
FedAvg [33]	58.34(0.86)	66.74(0.76)	49.32(0.93)	68.39(0.61)
LSS	**65.96**(1.50)	**75.16**(1.07)	**56.72**(0.53)	**73.32**(0.46)

Table 9: Different Network Architecture (ResNet-18 and ViT): Label shift test accuracy after $R = 1$ and $R = 3$ communication rounds.

	CIFAR-10 (ResNet-18)		**CIFAR-10 (ViT Base)**	
Method	Accuracy ($R = 1$) ↑	Accuracy ($R = 3$) ↑	Accuracy ($R = 1$) ↑	Accuracy ($R = 3$) ↑
FedAvg [33]	58.34(0.86)	66.74(0.76)	60.35(0.82)	69.38(0.51)
LSS	**65.96**(1.50)	**75.16**(1.07)	**67.48**(0.70)	**76.81**(0.47)

Evaluation with ViT. To validate the effectiveness of our method across different network architectures, we conducted an expanded experiment using the Vision Transformer (ViT) model based on the Transformer architecture. Upon observing the Table 9, it is evident that our method consistently enhances the communication efficiency of federated learning with ViT model architectures.

Evaluation with different Non-IID level. To further comprehensively validate the effectiveness of our method under different levels of data heterogeneity, we conducted experiments on the CIFAR-10 dataset by adjusting the coefficients α of the Dirichlet distribution. We examined the performance of our method in scenarios with greater distribution variations. Based on the Table 10, it is evident that our method maintains a significant advantage in scenarios with larger data heterogeneity.

Evaluation with different Initialized Models. To compare the performance of our method under different types of parameter initialization, we conducted experiments on the CIFAR-10 dataset using both pre-trained and random initialization. Table 11 shows that our method still maintains a significant advantage with random initialization, but it does not achieve the near-optimal performance seen with pre-trained initialization.

Comparison of Convergence Speed Between FedProx and LSS. Fig. 8 shows the accuracy of the testing during the early and late phases in terms of the number of communication rounds required to reach convergence. These results demonstrate that our method outperforms FedProx in both the early and late phases of federated learning.

Evaluation of Large Language Models for Multilingual Instruction Tuning

Setup. We follow the setup of Fed-Aya [53], which involves four iterative steps: server-to-client model downloading, local model training, client-to-server model uploading, and global model aggregation. For instruction tuning, we use the parameter-efficient fine-tuning technique, LoRA [18], applied to the Llama-7b model.

Dataset. We use the Aya dataset [42], a multilingual instruction tuning dataset with annotations from contributors worldwide. Our experiments include 6 high-resource languages (English, Spanish, French, Russian, Portuguese, Chinese) and 2 low-resource languages (standard Arabic, Telugu). The

Table 10: Different Non-IID level (Dirichlet distribution $\alpha = 1.0$ and $\alpha = 0.1$): Label shift test accuracy after $R = 1$ and $R = 3$ communication rounds.

	CIFAR-10 ($\alpha = 1.0$)		**CIFAR-10** ($\alpha = 0.1$)	
Method	Accuracy ($R = 1$) ↑	Accuracy ($R = 3$) ↑	Accuracy ($R = 1$) ↑	Accuracy ($R = 3$) ↑
FedAvg [33]	$58.34(0.86)$	$66.74(0.76)$	$18.30(2.25)$	$45.85(1.24)$
LSS	**65.96**(1.50)	**75.16**(1.07)	**26.70**(1.62)	**50.02**(0.82)

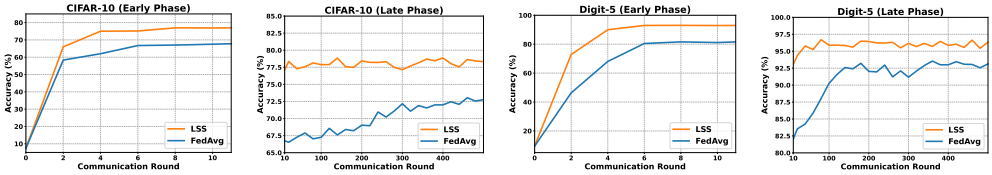
dataset is filtered to include contributors with at least 100 annotations, resulting in 38 clients with a total of 25k data samples.

Model. The model used for our experiments is the Llama2-7b [46], fine-tuned using the LoRA technique. We evaluate the effectiveness of the training methods using an in-domain evaluation metric termed Ref-GPT4 [58], where GPT-4o rates the generated responses against ground-truth responses. The score given by GPT-Ref ranges from 0 to 10. We adopt the same prompt template used in FedLLM-Bench [53]. The implementation of applying our method to LoRA is the same as that used in the ViT experiments (see Fig. 4) described earlier.

Result. Our method, LSS, when applied to large language models for instruction tuning, achieves higher scores than the common FedAvg. This suggests that LSS is a promising approach for improving performance and convergence in federated learning settings for large language models, in addition to its success in image classification. Exploring the use of our method in a diverse set of complex LLM tasks is an interesting direction for future research.

Table 11: Different model initialization (Pre-trained v.s. Random): Label shift test accuracy after $R = 1$ and $R = 3$ communication rounds. Result: It shows that our method still maintains a significant advantage with random initialization, but it does not achieve the near-optimal performance seen with pre-trained initialization.

	CIFAR-10 (Pre-trained)		**CIFAR-10 (Random)**	
Method	Accuracy ($R = 1$) ↑	Accuracy ($R = 3$) ↑	Accuracy ($R = 1$) ↑	Accuracy ($R = 3$) ↑
FedAvg [33]	58.34(0.86)	66.74(0.76)	14.83(2.03)	25.42(0.71)
LSS	**65.96**(1.50)	**75.16**(1.07)	**30.64**(1.73)	**37.86**(1.33)



(a) (b) (c) (d)

Figure 8: Convergence comparison of our proposed *LSS* with FedProx. *LSS* also achieves high accuracy much earlier (around 6 to 8 rounds) than FedProx, which takes hundreds of communication rounds.

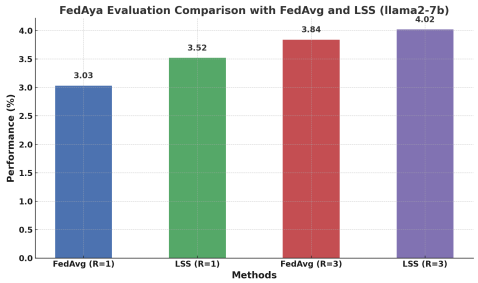


Figure 9: FedAya Evaluation Comparison with FedAvg and LSS. Our method, LSS, when applied to large language models for instruction tuning, achieves higher scores than the common FedAvg. This suggests that LSS is a promising approach for improving performance and convergence in federated learning settings for large language models, in addition to its success in image classification. Exploring the use of our method in a diverse set of complex LLM tasks is an interesting direction for future research.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Yes, the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope. The concepts of federated learning settings are clearly introduced and defined in Sec. 1.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper acknowledges several limitations and areas for future research in Sec. 5. Firstly, it highlights a trade-off between training memory and performance when reducing communication rounds in model merging. This indicates an awareness of the potential drawbacks of their method and suggests that further work is needed to make the method more training-memory efficient. Additionally, the paper notes that it focuses exclusively on vision-related tasks, suggesting that extending the findings to language tasks or multimodal scenarios would be a promising direction for future research.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best

judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Yes, the paper meticulously outlines all relevant assumptions and presents comprehensive, logically sound proofs, ensuring the validity and reliability of each theoretical result. These elements are included in Sec. 3 and Appendix.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper fully discloses all the information needed to reproduce the main experimental results. This information is provided in Sec. 4, with additional detailed information available in the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Yes, the paper specifies all the training and test details necessary to understand the results, including data splits, hyperparameters, their selection process, and the type of optimizer used. This comprehensive detailing ensures that the experimental setup and outcomes are transparent and reproducible in Section 4.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies all the training and test details necessary to understand the results, including data splits, hyperparameters, their selection process, and the type of optimizer used. These details are comprehensively documented in Sec. 4 to ensure clarity and reproducibility.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Yes, the paper reports error bars and other appropriate information about the statistical significance of the experiments in Sec. 4. These details are suitably and correctly defined to ensure the reliability and validity of the reported results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Yes, the paper provides sufficient information on the computer resources needed to reproduce the experiments in Sec. 4. This includes details on the type of compute workers, memory requirements, and execution time, ensuring that others can accurately replicate the experimental setup and results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

Justification: Yes, the research conducted in the paper conforms, in every respect, with the NeurIPS Code of Ethics. The paper adheres to the guidelines on responsible release and publication strategy, ensuring that all necessary safeguards are in place for controlled use of the model (see Sec. 4).

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: es, the paper discusses both potential positive and negative societal impacts of the work performed. The discussion includes an analysis of how the research can benefit society and also addresses possible adverse effects, ensuring a balanced and comprehensive evaluation of the societal implications in Sec. 5.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The paper does not describe safeguards for the responsible release of data or models that have a high risk for misuse, such as pretrained language models, image generators, or scraped datasets. Because we does not focus the release of such assets, we focus model training techniques. (see Sec. 4).

Guidelines:

- The answer NA means that the paper poses no such risks.

- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [NA]

    Justification: The paper does not utilize external assets such as code, data, or models from other creators or original owners. Therefore, there are no specific credits, licenses, or terms of use that need to be mentioned or respected.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not introduce any new assets such as code, data, or models. Therefore, there is no documentation provided alongside new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The paper does not involve crowdsourcing experiments or research with human subjects. Therefore, it does not include instructions given to participants, screenshots, or details about compensation.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The paper does not involve any study participants, crowdsourcing experiments, or research with human subjects. Therefore, it does not describe potential risks incurred by study participants, disclose such risks to subjects, or obtain Institutional Review Board (IRB) approvals.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.