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	coefficient for local training regularization term
τ	local training steps

B CONVERGENCE ANALYSIS

B.1 FORMAL RESTATEMENT OF CONVERGENCE THEOREM

Standard FL (McMahan et al., 2017) 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 in Wang et al. (2021a) and ours) 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*

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

Following Wang et al. (2021a), we have the main theorem on converngence rate as follows. For the complete proof, please refer to Wang et al. (2021a). The main theorem on convergence rate as follows.

Theorem B.1 (**Theorem 1**, Convergence Rate for Convex Local Functions Wang et al. (2021a))**.** *Under the aforementioned assumptions, we have*

$$\mathbb{E}\left[\frac{1}{\tau R}\sum_{r=0}^{R-1}\sum_{k=1}^{\tau}\mathcal{L}(\overline{f}^{(r,k)}) - \mathcal{L}(f^\star)\right] \leq \frac{d^2}{2\eta\tau R} + \frac{\eta\sigma^2}{M} + 4\tau\eta^2\beta\sigma^2 + 18\tau^2\eta^2\beta\zeta^2. \tag{8}$$

Further, when the client learning rate is chosen as

$$\eta = \min\left\{\frac{1}{4\beta}, \frac{M^{\frac{1}{2}}d}{\tau^{\frac{1}{2}}R^{\frac{1}{2}}\sigma}, \frac{d^{\frac{2}{3}}}{\tau^{\frac{2}{3}}R^{\frac{1}{3}}\beta^{\frac{1}{3}}\sigma^{\frac{2}{3}}}, \frac{d^{\frac{2}{3}}}{\tau R^{\frac{1}{3}}\beta^{\frac{1}{3}}\zeta^{\frac{2}{3}}}\right\}, \tag{9}$$

we have

$$\mathbb{E}\left[\frac{1}{\tau R}\sum_{r=0}^{R-1}\sum_{k=1}^{\tau}\mathcal{L}(\overline{f}^{(r,k)}) - \mathcal{L}(f^\star)\right] \leq \frac{2\beta d^2}{\tau R} + \frac{2\sigma d}{\sqrt{M\tau R}} + \underbrace{\frac{5\beta^{\frac{1}{3}}\sigma^{\frac{2}{3}}d^{\frac{4}{3}}}{\tau^{\frac{1}{3}}R^{\frac{2}{3}}} + \frac{19\beta^{\frac{1}{3}}\zeta^{\frac{2}{3}}d^{\frac{4}{3}}}{R^{\frac{2}{3}}}}_{\text{errors from local updates \& Non}-\text{IID data}}. \tag{10}$$

Here, $d := \|f^{(0,0)} - f^\star\|$ refers to the distance between initialization and the global optimum $f^\star$.

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

$$\tau \leq \frac{\sigma}{\zeta}\sqrt{\frac{\sigma}{d\beta}\frac{K^{\frac{1}{2}}}{M^2}}, \tag{11}$$

the error upper bound Eq.equation 11 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 Wang et al. (2021a), when the total number of gradient evaluations/computations across all clients ($K = M\tau R$) is fixed and the local steps τ satisfies:

$$\tau \leq \min\left\{\frac{\sigma}{d\beta}\frac{K^{\frac{1}{2}}}{M^2}, \frac{\sigma}{\zeta}\sqrt{\frac{\sigma}{d\beta}\frac{K^{\frac{1}{2}}}{M^2}}\right\}. \tag{12}$$

When the upper bound of local steps (Eq.(12)) becomes larger, there will be more communication savings. Therefore, the quantity in Eq.(12) 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 $\zeta \geq \sigma$, and:

$$1 \leq \frac{\sigma}{\zeta}\sqrt{\frac{\sigma}{d\beta}\frac{K^{\frac{1}{2}}}{M^2}} \leq \sqrt{\frac{\sigma}{d\beta}\frac{K^{\frac{1}{2}}}{M^2}} \leq \frac{\sigma}{d\beta}\frac{K^{\frac{1}{2}}}{M^2} \tag{13}$$

Therefore, we have Lemma 1:

$$\tau \leq \frac{\sigma}{\zeta}\sqrt{\frac{\sigma}{d\beta}\frac{K^{\frac{1}{2}}}{M^2}}, \tag{14}$$

□

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

Why connected low-loss valley + pre-trained initialization can achieve extreme communication rounds reduction? Based on the analysis above, we find that simply increasing the number of local training steps is insufficient for achieving extreme communication efficiency. The key lies in reducing the error term introduced by local updates. Importantly, to achieve a significant reduction in communication rounds, our primary focus should be on decreasing the last term of the RHS of Formula 2. This is because, under the extreme communication round reduction condition (*e.g.*, $R = 1$), the denominators of the first three terms all involve the local training steps τ. As τ approaches infinity, the influence of these error terms can be eliminated, but the last error term remains. This term is mainly affected by three factors: local and global gradient dissimilarity ζ, distance between initialized and optimal weights d, and the Lipschitz constant β related with smoothness. Previous research (Nguyen et al., 2022) has shown that FL training based on pre-training initialization can better align updates from different clients, reducing the ζ term, which represents the difference between local and global gradients. Additionally, due to the characteristics of existing overparameterized models (Chizat et al., 2019; Li & Banerjee, 2021), the optimal solution is typically near the initialized point, leading to a very small d term. As for the smoothness β term, intuitively, if clients are trapped in isolated low-loss valleys, this situation reflects the non-smoothness of the local model function. By encouraging the regularization of training to find connected low-loss regions, we can effectively reduce the potential maximum value of the β term during the training process. Through the above analysis, we conclude that pre-training initialization combined with our regularization training that encourages the search for connected regions can reduce the error terms introduced by local updates, thus increasing the upper limit of local training steps and achieving the goal of reducing 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 (Izmailov et al., 2018). 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:

$$\mathbb{E}_{L_S^N} \mathcal{E}_T(f_{\text{WA}}(L_S^N)) = \mathbb{E}_{(x,y)\sim p_T} \Big[\text{bias}^2(x,y) + \frac{1}{N}\text{var}(x) + \frac{N-1}{N}\text{cov}(x) \Big] + O(\bar{\Delta}^2), \tag{15}$$

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 Ramé et al. (2022b), 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:

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

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

$$\operatorname{bias}(x, y) = y - \mathbb{E}_{l_S}[f(x, l_S)]. \tag{16}$$

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

$$\operatorname{var}(x) = \mathbb{E}_{f_S}\left[\left(f(x, l_S) - \mathbb{E}_{l_S}\left[f(x, l_S)\right]\right)^2\right]. \tag{17}$$

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*

$$\operatorname{cov}(x) = \mathbb{E}_{l_S, l'_S}\left[\left(f(x, l_S) - \mathbb{E}_{l_S}\left[f(x, l_S)\right]\right)\left(f(x, l'_S) - \mathbb{E}_{l_S}\left[f(x, l_S)\right]\right)\right]. \tag{18}$$

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*

$$\bar{\Delta}^2 = \mathbb{E}_{L_S^N} \Delta^2_{L_S^N} \text{ with } \Delta_{L_S^N} = \max_{i=1}^{N} \left\| f_i - f_{\mathrm{WA}} \right\|_2. \tag{19}$$

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 Ramé et al. (2022b). 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.

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.

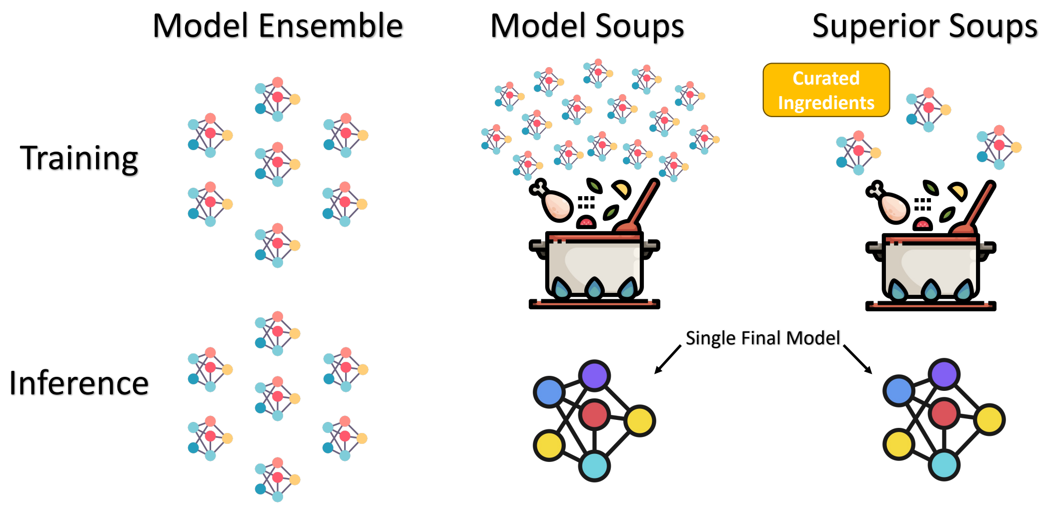

Figure 7: Comparison on model ensemble, model soups, and superior soups.

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 (Li et al., 2020) 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 (Karimireddy et al., 2020) adds a variance reduction term to mitigate the "clients-drift." MOON (Li et al., 2021a) 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 (Tan et al., 2021) is another approach to achieving high local testing performance on Non-IID data. For aggregation perspective: FedBN (Li et al., 2021b) 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 (Guha et al., 2019) instead of weight ensembles or generating data (Zhang et al., 2022a; Heinbaugh et al., 2023) 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 (Park et al., 2021), 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. Choshen et al. (2022) 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. Zhang et al. (2022b) 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 (Oh et al., 2022) splits the model into body and head, then fine-tuning the head part for personalization. Cheng et al. (2021) propose FTFA and RTFA that start with a pre-trained model and then fine-tunes a small subset of model parameters using the FedAvg (McMahan et al., 2017) 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) (Nagarajan & Kolter, 2019; Frankle et al., 2020), Model Soups (Wortsman et al., 2022) proposes to combine many independent runs with varied hyper-parameter configurations. Similarly, DiWA (Ramé et al., 2022b) 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 (Wortsman et al., 2022; Ramé et al., 2022b) rely on aggregating diverse models to improve model generalizability. To induce greater diversity, some methods such as (Maddox et al., 2019) using a high constant learning rate, (Wortsman et al., 2021) minimizing cosine similarity between weights, (Izmailov et al., 2019) using a tempered posterior and model Ratatouille (Ramé et al., 2022a) 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 Xiao et al. (2017), CIFAR-10 Krizhevsky et al. (2009), Digit-5 Ganin & Lempitsky (2015); Li et al. (2021b), and DomainNet Peng et al. (2019). 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 Deng (2012) (55000 samples), MNIST-M (55000 samples), Synthetic Digits Ganin & Lempitsky (2015) (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) Dosovitskiy et al. (2021) 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 Izmailov et al. (2018), SWAD Cha et al. (2021), 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 Wortsman et al. (2022) method and DiWA Ramé et al. (2022b), we trained 32 models and each model trained 8 steps. The hyper-parameter configuration for model selection includes learning rate ($[1\mathrm{e}{-4}, 5\mathrm{e}{-4}, 1\mathrm{e}{-5}]$), batch size ($[32, 64, 128]$), dropout rate ($[0.0, 0.1, 0.3]$), and weight decay $[5\mathrm{e}{-4}, 5\mathrm{e}{-5}, 5\mathrm{e}{-6}]$. Each run randomly select one of the hyper-parameter options. From each run of WAFT method, we take

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

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 2017	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 2017	SWA 2018	Soups 2022	***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 2017	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 (Yao et al., 2020). This metric signifies the maximum curvature of the loss landscape, commonly employed in the literature on flat minima (Kaddour et al., 2022) 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 $\downarrow$	*LSS* $(M = 2)\downarrow$	*LSS* $(M = 3)\downarrow$	*LSS* $(M = 4)\downarrow$
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 $(R = 1)\uparrow$	Accuracy $(R = 3)\uparrow$	Accuracy $(R = 1)\uparrow$	Accuracy $(R = 3)\uparrow$
FedAvg 2017	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)

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.

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 $(R = 1)\uparrow$	Accuracy $(R = 3)\uparrow$	Accuracy $(R = 1)\uparrow$	Accuracy $(R = 3)\uparrow$
FedAvg 2017	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 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.

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 ($R = 1$) $\uparrow$	Accuracy ($R = 3$) $\uparrow$	Accuracy ($R = 1$) $\uparrow$	Accuracy ($R = 3$) $\uparrow$
FedAvg 2017	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)