Supplement for "Federated Self-supervised Learning for Heterogeneous Clients"

In this supplementary material, we provide proofs for the key results in the paper in Appendix A along with the intuitive explanations of the results. Then, we present a setting to illustrate the framework's utility in the text classification setting in Appendix B, and provide additional experiments in Appendix C.

A PROOFS

We first provide proof of Lemma 4.5 in Appendix A.1. Then, the proof of Lemma 4.6 is given in Appendix A.2.

A.1 PROOF OF LEMMA 4.5

Let $\mathcal{L}_E$ denote the loss function after E local epochs. Then, from the Lipschitz smoothness assumption in Assumption 4.1, we obtain that

$$\mathcal{L}_1 \leq \mathcal{L}_0 - \eta[\nabla \mathcal{L}_0^T g_0] + \frac{L_1 \eta^2}{2}[||g_0||^2].$$

Therefore, we have

$$\begin{aligned}
\mathbb{E}[\mathcal{L}_1] &\leq \mathbb{E}[\mathcal{L}_0] - \eta \mathbb{E}[\nabla \mathcal{L}_0^T g_0] + \frac{L_1 \eta^2}{2}\mathbb{E}[||g_0||^2] \\
&= \mathbb{E}[\mathcal{L}_0] - \eta ||\nabla \mathcal{L}_0||^2 + \frac{L_1 \eta^2}{2}\mathbb{E}[||g_0||^2] \\
&= \mathbb{E}[\mathcal{L}_0] - \eta ||\nabla \mathcal{L}_0||^2 + \frac{L_1 \eta^2}{2}(Var(g_0) + \mathbb{E}[||g_0||]^2) \\
&\leq \mathbb{E}[\mathcal{L}_0] - (\eta - \frac{L_1 \eta^2}{2})||\nabla \mathcal{L}_0||^2 + \frac{L_1 \eta^2}{2}\sigma^2.
\end{aligned}$$

By summing the above bounds over E number of local epochs, we find that

$$\begin{aligned}
\sum_{i=1}^{E} \mathbb{E}[\mathcal{L}_i] &\leq \sum_{i=0}^{E-1} \mathbb{E}[\mathcal{L}_i] - (\eta - \frac{L_1 \eta^2}{2}) \sum_{i=0}^{E-1} ||\nabla \mathcal{L}_i||^2 + \frac{L_1 E \eta^2}{2}\sigma^2 \\
\mathbb{E}[\mathcal{L}_E] &\leq \mathcal{L}_0 - (\eta - \frac{L_1 \eta^2}{2}) \sum_{i=0}^{E-1} ||\nabla \mathcal{L}_i||^2 + \frac{L_1 E \eta^2}{2}\sigma^2.
\end{aligned}$$

Therefore, if $\eta < \frac{2(\sum_{i=0}^{E-1} ||\nabla \mathcal{L}_i||^2)}{L_1(\sum_{i=0}^{E-1} ||\nabla \mathcal{L}_i||^2 + E\sigma^2)}$, we obtain $\mathcal{L}_E \leq \mathcal{L}_0$ and the local loss reduces after E local epochs. As a consequence, we obtain the conclusion of the lemma.

A.2 PROOF OF LEMMA 4.6

For any arbitrary client after the global representation update step, if the loss function gets modified to $\mathcal{L}_{E'}$ from $\mathcal{L}_E$, we have

$$\begin{aligned}
\mathcal{L}_{E'} &= \mathcal{L}_E + \mathcal{L}_{E'} - \mathcal{L}_E, \\
&= \mathcal{L}_E + \mu\Big(\mathrm{d}(.;t) - \mathrm{d}(.;t-1)\Big).
\end{aligned}$$

As for client i, $\mathrm{d}(.;t) = \text{Linear-CKA}(K_i(t), \bar{K}(t))$, we have,

$$\mathcal{L}_{E'} = \mathcal{L}_E + \mu\Big(\text{Linear-CKA}(K_i(t), \bar{K}(t)) - \text{Linear-CKA}(K_i(t), \bar{K}(t-1))\Big).$$

Since this holds for all clients, we drop the client index i going forward and obtain that

$$\begin{aligned}
\mathcal{L}_{E'} &= \mathcal{L}_E + \mu\Big(\text{trace}(K(t)\bar{K}(t)) - \text{trace}(K(t)\bar{K}(t-1))\Big) \\
&= \mathcal{L}_E + \mu\Big(\text{trace}(K(t)(\bar{K}(t) - \bar{K}(t-1)))\Big) \\
&= \mathcal{L}_E + \mu\Big(\text{trace}(K(t)(\sum_{k=1}^{N} w_k K_k(t) - \sum_{k=1}^{N} w_k K_k(t-1))\Big) \\
&= \mathcal{L}_E + \mu\Big(\sum_{i=1}^{L}\sum_{j=1}^{L}(K(t)_{i,j}(\sum_{k=1}^{N} w_k K_k(t)_{i,j} - \sum_{k=1}^{N} w_k K_k(t-1)_{i,j})\Big) \\
&= \mathcal{L}_E + \mu\Big(\sum_{k=1}^{N} w_k(\sum_{i=1}^{L}\sum_{j=1}^{L} K(t)_{i,j}(K_k(t)_{i,j} - K_k(t-1)_{i,j}))\Big) \\
&= \mathcal{L}_E + \mu\Big(\sum_{k=1}^{N} w_k(\sum_{i=1}^{L}\sum_{j=1}^{L} K(t)_{i,j}(\Phi_k(x_i;\mathcal{W}_k^t).\Phi_k(x_j;\mathcal{W}_k^t) - \Phi_k(x_i;\mathcal{W}_k^{t-1}).\Phi_k(x_j;\mathcal{W}_k^{t-1})))\Big).
\end{aligned}$$

Given the above equations, we find that

$$\begin{aligned}
\mathcal{L}_{E'} = \mathcal{L}_E + \mu\sum_{k=1}^{N} w_k \sum_{i=1}^{L}\sum_{j=1}^{L} K(t)_{i,j}\Big(&\Phi_k(x_i;\mathcal{W}_k^t)(\Phi_k(x_j;\mathcal{W}_k^t) - \Phi_k(x_j;\mathcal{W}_k^{t-1})) \\
&+ \Phi_k(x_j;\mathcal{W}_k^{t-1})(\Phi_k(x_i;\mathcal{W}_k^t) - \Phi_k(x_i;\mathcal{W}_k^{t-1})).\Big)
\end{aligned}$$

Taking norm on both the sides, we get

$$\begin{aligned}
||\mathcal{L}_{E'}|| &= \Big|\Big|\mathcal{L}_E + \mu\sum_{k=1}^{N} w_k \sum_{i=1}^{L}\sum_{j=1}^{L} K(t)_{i,j}\Big(\Phi_k(x_i;\mathcal{W}_k^t)(\Phi_k(x_j;\mathcal{W}_k^t) - \Phi_k(x_j;\mathcal{W}_k^{t-1})) \\
&\quad + \Phi_k(x_j;\mathcal{W}_k^{t-1})(\Phi_k(x_i;\mathcal{W}_k^t) - \Phi_k(x_i;\mathcal{W}_k^{t-1}))\Big)\Big|\Big| \\
&\leq ||\mathcal{L}_E|| + \mu\Big|\Big|\sum_{k=1}^{N} w_k \sum_{i=1}^{L}\sum_{j=1}^{L} K(t)_{i,j}\Big(\Phi_k(x_i;\mathcal{W}_k^t)(\Phi_k(x_j;\mathcal{W}_k^t) - \Phi_k(x_j;\mathcal{W}_k^{t-1})) \\
&\quad + \Phi_k(x_j;\mathcal{W}_k^{t-1})(\Phi_k(x_i;\mathcal{W}_k^t) - \Phi_k(x_i;\mathcal{W}_k^{t-1}))\Big)\Big|\Big| \\
&\leq \mathcal{L}_E + \mu\sum_{k=1}^{N} w_k \sum_{i=1}^{L}\sum_{j=1}^{L} ||K(t)_{i,j}||\Big(||\Phi_k(x_i;\mathcal{W}_k^t)(\Phi_k(x_j;\mathcal{W}_k^t) - \Phi_k(x_j;\mathcal{W}_k^{t-1})) \\
&\quad + \Phi_k(x_j;\mathcal{W}_k^{t-1})(\Phi_k(x_i;\mathcal{W}_k^t) - \Phi_k(x_i;\mathcal{W}_k^{t-1}))||\Big) \\
&\leq \mathcal{L}_E + \mu\sum_{k=1}^{N} w_k \sum_{i=1}^{L}\sum_{j=1}^{L} ||K(t)_{i,j}||\Big(||\Phi_k(x_i;\mathcal{W}_k^t)||.||(\Phi_k(x_j;\mathcal{W}_k^t) - \Phi_k(x_j;\mathcal{W}_k^{t-1}))|| \\
&\quad + ||\Phi_k(x_j;\mathcal{W}_k^{t-1})||.||(\Phi_k(x_i;\mathcal{W}_k^t) - \Phi_k(x_i;\mathcal{W}_k^{t-1}))||\Big) \\
&\leq \mathcal{L}_E + \mu\sum_{k=1}^{N} w_k \sum_{i=1}^{L}\sum_{j=1}^{L} ||K(t)_{i,j}||\Big(||\Phi_k(x_i;\mathcal{W}_k^t)||.||\mathrm{L}_2\eta g_{t,k}|| + ||\Phi_k(x_j;\mathcal{W}_k^{t-1})||.||\mathrm{L}_2\eta g_{t,k}||\Big).
\end{aligned}$$

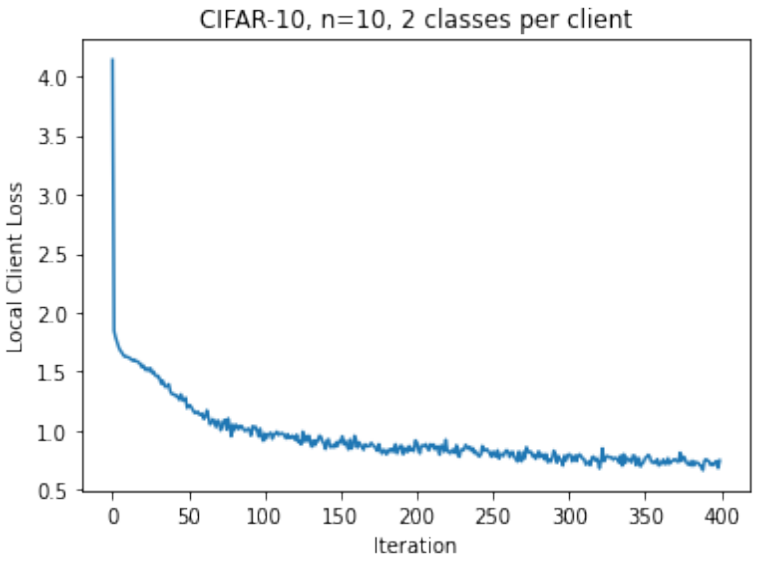

Figure 2: Local loss at the end of each training round for a randomly selected client.

Taking expectation on both sides of the above bounds, we have

$$\begin{aligned}
\mathbb{E}[\mathcal{L}_{E'}] &\leq \mathbb{E}[\mathcal{L}_E] + \mu \sum_{k=1}^{N} w_k \sum_{i=1}^{L}\sum_{j=1}^{L} ||K(t)_{i,j}|| \Big(\mathbb{E}[||\Phi_k(x_i; \mathcal{W}_k^t)||.||\mathrm{L}_2 \eta g_{t,k}||] + \mathbb{E}[||\Phi_k(x_j; \mathcal{W}_k^{t-1})||.||\mathrm{L}_2 \eta g_{t,k}||] \Big) \\
&\leq \mathbb{E}[\mathcal{L}_E] + \mu \sum_{k=1}^{N} w_k \sum_{i=1}^{L}\sum_{j=1}^{L} ||K(t)_{i,j}|| \Big(R\mathrm{L}_2 \eta P + R\mathrm{L}_2 \eta P \Big) \\
&= \mathbb{E}[\mathcal{L}_E] + 2\mu\eta\mathrm{L}_2 PR \sum_{i=1}^{L}\sum_{j=1}^{L} ||K(t)_{i,j}|| \\
&\leq \mathbb{E}[\mathcal{L}_E] + 2\mu\eta\mathrm{L}_2 PR \sum_{i=1}^{L}\sum_{j=1}^{L} R^2 \\
&\leq \mathbb{E}[\mathcal{L}_E] + 2\mu\eta\mathrm{L}_2 PR^3 L^2.
\end{aligned}$$

Thus, we have,

$$\mathbb{E}[\mathcal{L}_{E'}] \leq \mathbb{E}[\mathcal{L}_E] + 2\mu\eta\mathrm{L}_2 PR^3 L^2,$$

which concludes the proof.

A.3 Convergence Demonstration in Experiments

We monitor the local loss at the end of each training round and we see that the decreasing loss theorem holds in practice as well. The Figure 2 shows a plot of the local loss for one randomly selected client with the number of global round for the proposed method on CIFAR-10 dataset.

B Applications in Other Domains

While the experiments with image recognition tasks demonstrated the potential of the approach in popular federated self-supervised settings, note that the Hetero-SSFL framework is indeed extensible to other domains and applications as well since the key idea of the framework (i.e., enhance local self-supervised learning on heterogeneous clients with limited data by obtaining peer supervision from other clients) is broadly applicable. We now briefly illustrate how the proposed method can be applied for the text classification task.

Consider each client to be learning a transformer for language modeling using the self-supervised Masked Language Modeling(MLM) objective. This objective tries to reconstruct the sentence by predicting the masked words in the sentence. Thus, the local self-supervised loss function for client k, $\ell(\mathcal{W}_k)$, corresponds to the MLM loss. The peer supervision is achieved as usual, by server obtaining the embeddings on the RAD from each client, aggregating those embeddings and sending it to the clients for the next round of training. The local loss function at each client is similar to Equation (4) but with the new $\ell(\mathcal{W}_k)$. After pre-training using self-supervision, we evaluate the performance of training by following a linear evaluation protocol using a sentiment classification task on Sentiment140 dataset from LEAF (Caldas et al., 2019).

Experimental Setup For local self-supervised learning we use ALBERT transformer (Lan et al., 2020). We partition the Sentiment140 data in 5 non-iid partitions for each client and fine-tune the transformer for each client on locally available data for 200 rounds, using AdamW optimizer and a range of μ values between .01 and 100 and report the best performance. After pre-training, the parameters of the transformer are frozen and a linear classifier is trained on the sentence embeddings obtained from the transformer. The test accuracy of the linear classifier is reported as the result.

For baselines, we consider the client's local model training on local data, FedAvg version of ALBERT transformer and a divergence aware update method as shown in the baselines FEDEMA and FedU. Since the other self-supervised federated methods like FedEMA and FedU consider only the image classification task, we are unable to compare against them.

The Table 4 below shows the test accuracy of Hetero-SSFL in comparison with the above mentioned baselines.

Table 4: Test accuracy for different methods for text classification on Sentiment140 dataset.

	Method			
	FedAvg	Local	Divergence-Aware	Hetero-SSFL
Test Accuracy	61.53 ± 7.2	61.7 ± 2.2	58.19 ± 9.2	$\mathbf{62.79 \pm 1.7}$

C Additional Experiments

Table 5: Top-1 test accuracy for different methods under semi-supervised learning for CIFAR100.

Method	1% labeled data	10% labeled data
FedU	20.5 ± 0.5	46.3 ± 0.57
FedEMA	23.11 ± 0.65	48.6 ± 0.47
Hetero-SSFL	$\mathbf{37.4 \pm 0.95}$	$\mathbf{57.4 \pm 0.43}$

Table 6: Top-1 test accuracy for different methods under semi-supervised learning for CIFAR10.

Method	1% labeled data	10% labeled data
FedU	72.9 ± 2.25	86.8 ± 0.5
FedEMA	75.8 ± 1.4	87.4 ± 0.8
Hetero-SSFL	$\mathbf{79.3 \pm 1.85}$	$\mathbf{88.7 \pm 0.25}$

Table 7: Average effect of collaboration on different clients. Column 2 depicts the test accuracy of clients with local models and column 3 the accuracy under Hetero-SSFL.

Client Model Architecture	Local SSL	Hetero-SSFL
Resnet-18	71.9	**85.4**
Resnet-34	77.8	**93.7**

We also test our framework on the semi-supervised learning task, as described in FedU (Zhuang et al., 2021) and FedEMA (Zhuang et al., 2022) papers. Under this protocol, we let the federated SSFL

methods train the local models for representation learning. For the evaluation, we assume that only a small fraction(10% or 1%) of labeled data is available. Then, after the federated representation learning of local models on clients, we fine-tune the local models along with a linear classification layer on top for 100 epochs. We evaluate our method and the baselines, FedU and FedEMA, in this setting on CIFAR10 and CIFAR100 datasets and observe that our method outperforms the baselines. For fine-tuning the models, we use SGD with Nesterov momentum as the optimizer with momentum = 0.9, learning rate = 0.05 and batch size = 512 and 256 for 10% labeled data and 1% labeled data respectively. The results reported in Table 5 and Table 6 are for federated setting with non-IID clients with 5 users, 20 classes per user for CIFAR100 and 2 classes per user for CIFAR10.

For additional insights, we show the average change in test accuracy of clients with different architectures when performing standalone training versus collaborative learning in the heterogeneous setting in Table 7.

D Details about RAD

The Representation Alignment Dataset (RAD) is important for training in Hetero-SSFL. Functionally, the RAD is used to guide the simultaneous training on the peer clients by serving as a common dataset for representation alignment. Also, since the models are not transmitted and only the RAD representations are transmitted from clients to the server, there is no privacy loss due to involvement of RAD in the training process. The only assumption about this dataset is to be of the same domain as the training data, i.e., for image classification tasks it could be images sourced from the web and for the text-classification tasks it could be publicly available text like from Wikipedia, Reddit etc. Because the only requirement for RAD is to come from the same domain, it does not pose practical constraints. Several applications can make use of publicly available datasets to use for training in Hetero-SSFL, for example, learning self-supervised models for image datasets could use MNIST or CIFAR datasets for training, similarly acoustic applications can use audio datasets provided by Huggingface, and likewise.

In the experiments described in Table 1 and Table 2, we subset a part of the dataset (5000 data points) and set it aside to be used as RAD before beginning the training process. Since the RAD is considered generic with no bearing on clients, it is created before the clients are intialised and data is partitioned into clients. We further analysed the effect of using simple MNIST dataset as an RAD and saw similar results. The change in performance with respect to the size of RAD is given below in the Figure 3. When the RAD size is very low, the local models performance is closer to the performance in standalone training. As the size of the RAD grows, the performance gets better but because the gain is small after 3000 points, we keep $L = 3000$ for all the experiment simulations. Also, in cases when $\mu = 0$, the performance of the Hetero-SSFL is same as standalone training ensuring that the performance gain is not due to additional data in the form of RAD.

E Communication Cost Analysis

In Hetero-SSFL, each global round involves the server sending the common dataset, RAD, to each client and the clients sending back the representations obtained on the RAD. The size of the RAD is given by L, thus the communication cost in each round is $O(L)$. In all our experiments, we set the RAD size between 3000 and 5000. Thus the communication cost in each round of Hetero-SSFL corresponds to transferring these many (3000-5000) parameters twice. This is orders of magnitude less than the conventional FL algorithms that require sharing of the model parameters which run in millions. For federated training of the Resnet18 models, having 11M number of parameters, Hetero-SSFL with $L = 5000$ will require 2000 times less communication cost than conventional methods in each round, making it very practical.

F Ablation Studies

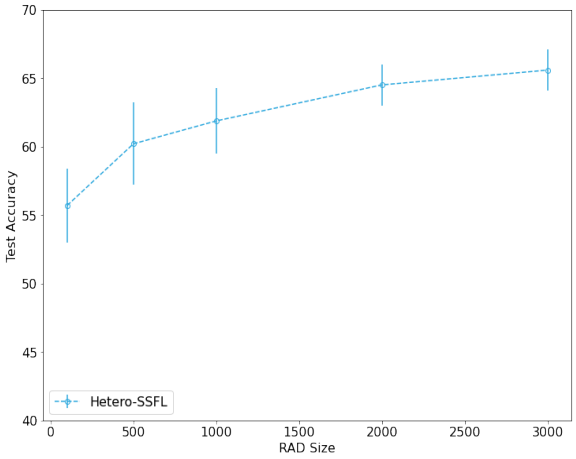

Figure 3: Change in performance with change in RAD size for CIFAR-100 dataset in non-IID setting.

Table 8: Test Accuracy comparison of our method with increasing heterogeneity for CIFAR-100 datasets and non-IID setting.

Setting	Small Models	Resnet Models
20 % small	39.4 ± 1.55	64.2 ± 2.9
40 % small	37.5 ± 1.96	63.7 ± 1.89
Standalone Training	28.8 ± 0.9	55.3 ± 1.3

F.1 Increased Heterogeneity in Clients

To demonstrate the effectiveness of the framework with diverse types of heterogeneous clients, we experiment by adding clients with models of much smaller capacities like CNN and MLP along with Resnet18 and Resnet34 models. We use Hetero-SSFL in this setup on CIFAR-100 dataset to achieve federated training of self-supervised models on CIFAR-100 dataset and use linear evaluation protocol, as described in Section 5.1 to measure the test accuracy. We show the average performance of the small capacity clients, and average performance of the large capacity clients in settings with 20% and 40% small capacity models in Table 8 as compared to the standalone training.

F.2 Effect of μ

The parameter μ regulates the weight of the proximal term (used for peer alignment) against the local self-supervised loss, as shown in Equation 1. Thus, the value of μ plays an important role in the entire training process. $\mu = 0$ corresponds to standalone training where each client is minimizing the local self-supervised loss on its own without any federation. As the μ increases to $0.001, 0.01$, some federation starts to happen and the performance gets better than the standalone training. $\mu = 0.1$ achieves a good balance of the two loss functions for CIFAR datasets, and $\mu = 1$ for Tiny-Imagenet. But as the μ starts increasing beyond 1, the weight of the proximal term becomes higher than the self-supervised loss leading to a drop in performance. This effect is more pronounced when the heterogeneous models are diverse, setting $\mu = 10$ in those settings inhibits learning at the more powerful clients while still improving the performance on the less powerful clients. With this knowledge, we can potentially vary μ across clients depending on their local resources, for example, a client with less number of data points and lower training capacity could benefit more from proximal term by setting a higher μ.