Supplementary Materials for Onboarding Without Forgetting: Hypernetwork Personalization with Data-Free Replay for Personalized Federated Learning

Thinh Nguyen[1, 2] Le Huy Khiem[3] Van-Tuan Tran[4]
Khoa D Doan[1, 2] Nitesh V Chawla[3] Kok-Seng Wong[1, 2*]
[1]VinUni-Illinois Smart Health Center, VinUniversity, Hanoi, Vietnam
[2]College of Engineering & Computer Science, VinUniversity, Hanoi, Vietnam
[3]University of Notre Dame, Indiana, USA
[4]Technische Universität Berlin, Germany
thinh.nth@vinuni.edu.vn, kle3@nd.edu, tranva@tcd.ie
khoa.dd@vinuni.edu.vn, nchawla@nd.edu, wong.ks@vinuni.edu.vn

A. Implementation Details

Model Architectures. Our experiments utilize **LeNet-5** [4] as the backbone for all client-side networks. The final fully connected layer is adjusted to the number of classes in each dataset: 10 for CIFAR-10, 100 for CIFAR-100, and 200 for Tiny-ImageNet. **The global hypernetwork** is a two-layer Multi-Layer Perceptron (MLP) that maps client embeddings of dimension $d = 32$ through a hidden layer of 512 neurons to generate the complete personalized parameter vector of LeNet-5. For generating client embeddings, we employ a lightweight convolutional neural network (CNN) consisting of one convolutional layer (32 filters, 3×3 kernels), followed by BN, a ReLU activation, global average pooling, and a final linear transformation projecting to the 32-dimensional embedding space.

Training Procedure. Federated training is conducted over 200 communication rounds for initial client batches and an additional 100 rounds for each new client batch. Each communication round randomly selects 5% of currently available clients. Selected clients perform local training for one epoch per round using Stochastic Gradient Descent (SGD) with momentum set to 0.90, a batch size of 32, and an initial learning rate of 0.01 following a cosine decay schedule throughout training. Clients transmit their embedding vectors, gradients with respect to hypernetwork parameters ϕ, and batch-specific mask gradients to the server. No raw images, labels, or sample-specific gradients are exchanged. We summarize our method in the Algorithm 1.

DeepInversion-based Data-free Replay. Synthetic images for server-side replay are generated via DeepInversion using BN statistics from the client models. The image synthesis optimization runs for 250 iterations per image, with an initial learning rate of 0.1. The feature alignment losses are weighted by the following coefficients: total variation regularization $\beta_{\text{TV}} = 10^{-5}$, L_2 regularization $\beta_{L_2} = 10^{-4}$, and feature distribution alignment $\beta_{\text{feature}} = 10^{-2}$. To balance computational cost and diversity, the synthetic dataset per class per batch consists of 20 images for CIFAR-10 [1], 5 images for CIFAR-100 [1], and 3 images for Tiny-ImageNet [3]. Generated data are utilized solely for fine-tuning server-side global models.

Hardware and Computational Resources. All reported computational performance measurements, including server-side latency, are obtained by averaging over 200 federated training rounds executed on a single NVIDIA RTX A5000 GPU with 24 GB of VRAM. Experiments are conducted on a computational node equipped with a 32-core AMD CPU running at 3.0 GHz and 128 GB of RAM. For fairness in comparison, no multi-GPU or distributed training optimizations are employed.

B. Sensitivity Analysis

Hyper-parameters tuning. We analyze two key factors in our server-side replay: (i) the weights of the DeepInversion objective ($\beta_{\text{feat}}, \beta_{\text{TV}}, \beta_{L_2}$), and (ii) the per-step synthetic data budget. Unless stated otherwise, results are on **CIFAR-10** with **5 onboarding steps**. We report PA for new clients, RI for existing clients, and Fréchet Inception Distance (FID) [2]. Higher PA/RI are better, lower FID is better.

We see that when increasing β_{feat} tightens alignment between synthetic activations and BatchNorm statistics from client training, shrinking the synthetic–real gap. As β_{feat}

*Corresponding author: wong.ks@vinuni.edu.vn

Algorithm 1: pFedDSH: Personalized Federated Data-free Sub-Hypernetwork for PCO-FL

Input : global hypernetwork $H(\cdot;\phi)$, total batches T, global epochs R, local epochs E, learning rates η, α, masking regularizer λ.
Output: Personalized global hypernetwork $H(\cdot;\phi^{(T)})$, masks $\{m_t\}_{t=1}^{T}$.

for $t \leftarrow 1$ **to** T **do**
  for $r \leftarrow 1$ **to** R **do**
    // **Step 1: Embedding Transmission**
    foreach *client* $c \in \mathcal{B}_t$ **do**
      Client initializes local embedding e_c and sends (e_c, t) to server.
    // **Step 2: Model Generation**
    Server initializes or retrieves mask m_t.
    Server generates personalized masked model for each client $\theta_c^{(t,0)} \leftarrow H(e_c;\phi) \odot m_t$ and sends it to corresponding clients $c \in \mathcal{B}_t$.
    // **Step 3: Local Client Training**
    foreach *client* $c \in \mathcal{B}_t$ **do**
      Train locally for E epochs: $\theta_c^{(t)} \leftarrow \theta_c^{(t)} - \eta \nabla_{\theta_c^{(t)}} \mathcal{L}(\theta_c^{(t)}, \mathcal{D}_c)$.
      Send updated gradients $\nabla_\phi^{(c)}$ and mask gradients $\nabla_{m_t}^{(c)}$ to the server.
    // **Step 2: Server Aggregation**
    Update global hypernetwork and mask: $\phi^{(t)} \leftarrow \phi^{(t-1)} - \alpha \sum_{c\in\mathcal{B}_t} \nabla_\phi^{(c)}$, $m_t \leftarrow m_t - \alpha \sum_{c\in\mathcal{B}_t} \nabla_{m_t}^{(c)}$.
  // **Step 3: Replay via DeepInversion**
  if $t > 1$ **then**
    Generate synthetic dataset $\mathcal{S}_t$.
    Fine-tune previous batches' subnetworks with synthetic data.

Table 1. Sensitivity to replay loss weights on CIFAR-10 (5 onboarding steps). Means ± std over 5 seeds. PA/RI ↑ higher is better, FID ↓ lower is better. Replay time is the average server cost per batch on an RTX A5000.

β_{feat}	β_{TV}	β_{L_2}	PA (%) ↑	RI (%) ↑	FID ↓ / Time (s)
0.25	0.20	1×10^{-6}	7.95 ± 0.37	-0.35 ± 0.29	65.10 / 6.20
0.50	0.20	1×10^{-5}	9.85 ± 0.41	0.62 ± 0.33	53.40 / 7.10
1.00	**0.20**	1×10^{-5}	$\mathbf{11.32 \pm 0.28}$	$\mathbf{2.08 \pm 0.25}$	**47.20 / 8.00**
2.00	0.20	1×10^{-5}	11.45 ± 0.35	1.96 ± 0.31	44.80 / 9.70
1.00	0.50	1×10^{-5}	10.05 ± 0.42	1.42 ± 0.39	50.70 / 8.50
1.00	0.20	5×10^{-5}	10.65 ± 0.33	1.74 ± 0.27	48.30 / 8.10

rises, FID drops substantially and RI turns positive, indicating more faithful replay that better routes knowledge to earlier clients. In contrast, β_{TV} suppresses high-frequency artifacts. A moderate value stabilizes inversion without erasing class-discriminative detail. Pushing it to 0.50 oversmooths images, slightly hurting PA/RI and increasing solve time. Finally, β_{L_2} prevents degenerate, high-energy solutions. Too small can yield noisy textures, too large overregularizes and mildly reduces PA/RI even if FID stays low, because the replay set becomes less informative.

Table 2. Sensitivity to synthetic replay budget per step (DeepInversion). Means ± std over 5 seeds on CIFAR-10 (5 steps). Server time is the average per-batch replay time on an RTX A5000.

Images/step	Iter./image	PA (%) ↑	RI (%) ↑	FID ↓	Server time (s)
32	20	9.82 ± 0.41	0.64 ± 0.27	59.40	1.70
64	20	10.55 ± 0.35	1.05 ± 0.33	54.10	3.20
128	20	11.05 ± 0.29	1.54 ± 0.37	49.60	5.10
256	**20**	$\mathbf{11.42 \pm 0.32}$	$\mathbf{2.12 \pm 0.28}$	**47.30**	**8.00**
512	20	11.53 ± 0.44	2.25 ± 0.39	46.80	14.20

Synthetic budget. More images per step reduce FID and improve PA/RI, but server time also increases. We therefore use **256 images/step, 20 iters/image** as a good point for the main experiments.

C. Theoretical Analysis

Let's $\mathcal{B}_t$ be the t-th onboarding batch, $H(e_c;\phi)$ the global hypernetwork, $m_c \in \{0,1\}^{|\theta|}$ the binary mask for client c, and $\hat{\theta}_c = m_c \odot H(e_c;\phi)$ the personalized parameters served to that client. We first present formal assumptions and then the main convergence theorem with a detailed proof.

Assumption 1 (Smoothness)**.** *The global objective function is defined as*

$$\mathcal{F}(\phi) = \mathbb{E}_c \left[\mathcal{L}_c(H(e_c;\phi) \odot m_c)\right]$$

is L-smooth, meaning that there exists $L > 0$ such that for all ϕ, ϕ':

$$\|\nabla\mathcal{F}(\phi) - \nabla\mathcal{F}(\phi')\| \le L\|\phi - \phi'\|.$$

Assumption 2 (Unbiased Gradient)**.** *For each communication round r, the stochastic gradient obtained from participating clients is unbiased:*

$$\mathbb{E}_{c\sim\mathcal{B}_r}\left[\nabla_\phi \mathcal{L}_c(H(e_c;\phi^{(r)}) \odot m_c)\right] = \nabla_\phi \mathcal{F}(\phi^{(r)}).$$

Assumption 3 (Bounded Gradient Variance)**.** *There exists a finite constant $\sigma^2 > 0$ such that for every communication round r:*

$$\mathbb{E}_{c\sim\mathcal{B}_r}\left[\|\nabla_\phi \mathcal{L}_c(H(e_c;\phi^{(r)}) \odot m_c) - \nabla_\phi \mathcal{F}(\phi^{(r)})\|^2\right] \le \sigma^2.$$

Assumption 4 (Uniform Client Sampling)**.** *At each communication round r, exactly K clients are uniformly sampled without replacement.*

Assumption 5 (Mask Immutability)**.** *Each client's mask is updated only within its onboarding batch. Specifically, if client c joins in batch t_0, there exists a round $r_{freeze}(c)$ after which the mask remains fixed:*

$$m_c^{(r)} = m_c^{(r_{freeze}(c))}, \quad \forall r > r_{freeze}(c).$$

Assumption 6 (Diminishing Step-Sizes)**.** *The sequence of server learning rates $\{\eta^{(r)}\}$ satisfies Robbins-Monro conditions:*

$$\sum_{r=0}^{\infty} \eta^{(r)} = \infty, \quad \textit{and} \quad \sum_{r=0}^{\infty} \left(\eta^{(r)}\right)^2 < \infty.$$

Now we present the main theorem, as follows:

Theorem 1 (Convergence to Stationary Point)**.** *Under Assumptions 1-6, the sequence of hypernetwork parameters $\{\phi^{(r)}\}$ generated by pFedDSH converges to a stationary point of the global objective $\mathcal{F}$. Formally, we have:*

$$\lim_{r\to\infty} \mathbb{E}\left[\|\nabla \mathcal{F}(\phi^{(r)})\|^2\right] = 0.$$

Proof. Consider the iterative update rule at each communication round r:

$$\phi^{(r+1)} = \phi^{(r)} - \eta^{(r)} g^{(r)},$$

where

$$g^{(r)} = \frac{1}{K} \sum_{c\in\mathcal{B}_r} \nabla_\phi \mathcal{L}_c(H(e_c;\phi^{(r)}) \odot m_c).$$

By Assumption 1 (smoothness), we have:

$$\mathcal{F}(\phi^{(r+1)}) \le \mathcal{F}(\phi^{(r)}) + \langle \nabla \mathcal{F}(\phi^{(r)}), \phi^{(r+1)} - \phi^{(r)} \rangle + \frac{L}{2}\|\phi^{(r+1)} - \phi^{(r)}\|^2.$$

Substitute the update rule, and taking conditional expectation, by Assumptions 2 and 3, we have:

$$\mathbb{E}\left[\mathcal{F}(\phi^{(r+1)})|\phi^{(r)}\right] \le \mathcal{F}(\phi^{(r)}) - \eta^{(r)}\|\nabla \mathcal{F}(\phi^{(r)})\|^2 + \frac{L}{2}(\eta^{(r)})^2(\sigma^2 + \|\nabla \mathcal{F}(\phi^{(r)})\|^2).$$

Taking expectations and summing from $r = 0$ to T:

$$\sum_{r=0}^{T} \eta^{(r)} \mathbb{E}\|\nabla \mathcal{F}(\phi^{(r)})\|^2 \le \mathcal{F}(\phi^{(0)}) - \mathbb{E}[\mathcal{F}(\phi^{(T+1)})] + \frac{L\sigma^2}{2} \sum_{r=0}^{T} (\eta^{(r)})^2.$$

Since $\mathcal{F}(\phi)$ is bounded below, and by Assumption 6, we have $\sum_{r=0}^{\infty} (\eta^{(r)})^2 < \infty$. Thus, as $T \to \infty$:

$$\sum_{r=0}^{\infty} \eta^{(r)} \mathbb{E}\|\nabla \mathcal{F}(\phi^{(r)})\|^2 < \infty.$$

By Robbins-Monro lemma since $\sum_r \eta^{(r)} = \infty$, it follows:

$$\lim_{r\to\infty} \mathbb{E}\|\nabla \mathcal{F}(\phi^{(r)})\|^2 = 0.$$

Finally, by Assumption 5, masks remain fixed after the associated batch update. Hence, smoothness and bounded variance conditions continue to hold, validating the previous analysis. Therefore, the sequence converges. □

D. Neuron-Reuse Guarantees

In this section, we will show why batch-specific masking does not exhaust neurons even as client batches grow to hundreds, with one theorem that ties capacity growth directly to data novelty, not to the mere count of batches.

Setup. At onboarding step t for layer ℓ of width C_ℓ, let $m_{\le t}^{(\ell)} \in [0,1]^{C_\ell}$ be the cumulative channel mask and $U_t^{(\ell)} = \{i : m_{\le t}^{(\ell)}(i) = 1\}$ the allocated channels with size $A_t^{(\ell)} = |U_t^{(\ell)}|$. Gradients are gated by

$$\frac{\partial \mathcal{L}}{\partial \theta_{ij}^{(\ell)}} = \frac{\partial \mathcal{L}}{\partial \theta_{ij}^{(\ell)}} \cdot \max\Big\{ g_t^{(\ell)}(i)\big(1 - m_{<t}^{(\ell)}(i)\big), g_t^{(\ell-1)}(j)\big(1 - m_{<t}^{(\ell-1)}(j)\big)\Big\}, \tag{1}$$

so if both endpoints were previously allocated, the multiplicative factor is 0 and that connection is frozen. Let $\mathcal{S}_{t-1}^{(\ell)}$ denote the subspace of layer-ℓ function-gradients reachable by modifying only weights incident to $U_{t-1}^{(\ell)}$ (the "already-allocated subnetwork"). Let $G_t^{(\ell)}$ be the gradient subspace demanded by the new batch B_t at layer ℓ, and define the *novelty dimension*

$$r_t^{(\ell)} := \dim\left(G_t^{(\ell)} \cap \big(\mathcal{S}_{t-1}^{(\ell)}\big)^{\perp}\right).$$

Theorem 2. *Under the gradient gating above, the minimal number of* new *channels that step t can force at layer ℓ equals the novelty dimension $r_t^{(\ell)}$. Consequently,*

$$\Delta A_t^{(\ell)} \le r_t^{(\ell)} \quad \textit{and} \quad A_T^{(\ell)} \le A_0^{(\ell)} + \sum_{t=1}^{T} r_t^{(\ell)},$$

so layer ℓ cannot be exhausted by step T whenever $\sum_{t=1}^{T} r_t^{(\ell)} < C_\ell - A_0^{(\ell)}$.

Proof. Because connections whose endpoints lie in $U_{t-1}^{(\ell)}$ and $U_{t-1}^{(\ell-1)}$ receive zero gradient, the already-allocated subnetwork realizes exactly the directions in $\mathcal{S}_{t-1}^{(\ell)}$. Any component of $G_t^{(\ell)}$ orthogonal to this subspace cannot be effected without activating new channels. Each new channel introduces at most one independent direction for the layer's update, thus at least $r_t^{(\ell)}$ activations are necessary.

Conversely, allocate $r_t^{(\ell)}$ fresh channels and assign their incident weights to span a basis of $G_t^{(\ell)} \cap (\mathcal{S}_{t-1}^{(\ell)})^{\perp}$. Together with the frozen subspace $\mathcal{S}_{t-1}^{(\ell)}$, this reproduces the full step-t gradient. Hence the minimal new allocation equals $r_t^{(\ell)}$, yielding $\Delta A_t^{(\ell)} \leq r_t^{(\ell)}$. Summing over t gives the stated budget inequality; the non-exhaustion condition is the contrapositive of $A_T^{(\ell)} < C_\ell$. □

References

[1] Krizhevsky Alex. Learning multiple layers of features from tiny images. *https://www. cs. toronto. edu/kriz/learning-features-2009-TR. pdf*, 2009. 1

[2] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 1

[3] Yann Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015. 1

[4] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 2002. 1