FedHPro: Federated Hyper-Prototype Learning via Gradient Matching

Huan Wang [1 2] Jun Shen [1] Haoran Li [3] Zhenyu Yang [4] Jun Yan [1] Ousman Manjang [5] Yanlong Zhai [5] Di Wu [6]
Guansong Pang [2]

FedHPro: Federated Hyper-Prototype Learning via Gradient Matching

Huan Wang

- University of Wollongong & Singapore Management University
- supervised by Prof. Guansong Pang (SMU) & Prof. Jun Shen (UOW)

Paper: https://arxiv.org/abs/2605.13475

Code: https://github.com/mala-lab/FedHPro

ICML International Conference On Machine Learning

Outline

- 1. Background & Motivation
- 2. Methodology
- 3. Experiments
- 4. Conclusion

Background – Prototype-based FL

- Considering practicality and decent generalization, prototype-based FL methods have emerged as a novel FL paradigm, that *transfers global prototypes among clients* to tackle data heterogeneity.
- Existing prototype-based FL methods directly collect the local prototypes from biased data distributions of clients, which unavoidably introduces semantic inconsistency.
- This is evident in the feature distribution (Fig.1):
  - Vanilla (FedProto[1]) shows clear separation between different categories in simple domains but exhibits ambiguous clustering in hard domains.
- Their updates are still primarily performed at the prototype-level by client-specific representations, which may inherit semantic biases and thus fail to provide a consistently aligned global signal.

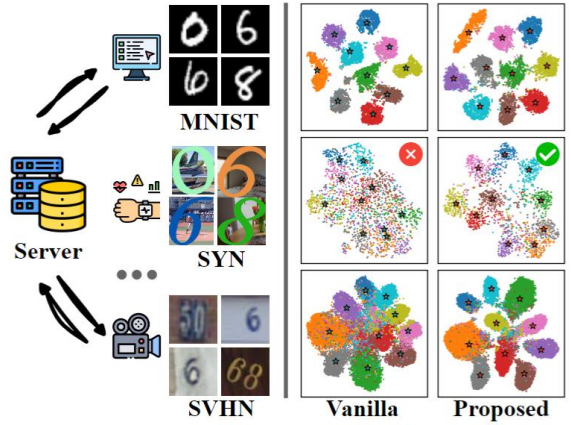
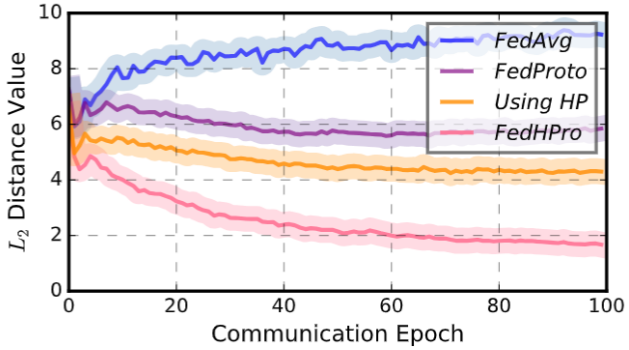


Figure 1. **Illustration of heterogeneous FL with domain skew.** The **Vanilla** column visualizes the feature distribution of a standard prototype-based FedProto (Tan et al., 2022a), showing failures in hard domains like SYN. In contrast, the **Proposed** column shows our approach achieves a larger inter-class distance and a smaller intra-class distance in such domains. Due to its simple aggregation, the prototypes in Vanilla (★) fail to reliably align with class centers, whereas our Proposed (★) yields better separability.

[1] Yue Tan et.al, 'FedProto: Federated Prototype Learning across Heterogeneous Clients', AAAI 2022.

Background – Motivation

- An intuitive solution is to preserve sufficient local data with diverse data distributions to learn a uniform representation space, thus producing semantically consistent global prototypes.
  - However, this is fundamentally incompatible with the privacy-preserving FL paradigm.
- This dilemma inspires our **core idea**: if the FL model could learn a representation space using all clients' samples, the obtained prototypes would exhibit better semantic consistency. In other words, **we expect to optimize global prototypes over all real samples, instead of adopting skewed clients' representations**.
- We achieve this by simulating the optimization trajectories of real samples via **gradient matching**.



METHODS	MNIST	USPS	SVHN	SYN	AVG ↑	△
FedAvg	96.68	90.43	76.35	51.85	78.82	–
FedProto	97.40	91.89	79.22	53.15	80.41	+1.59
Using HP	97.48	92.24	80.90	56.39	**81.75**	**+2.93**
FedHPro	98.52	93.13	84.95	62.59	**84.80**	**+5.98**

Left: The L_2 distance of centralized prototypes calculated by centralized training using all clients' samples to global prototypes from FedAvg, FedProto, and our method.
Right: The corresponding accuracy, '*Using HP*' means replacing global prototypes with hyper-prototypes.

Methodology – Federated Hyper-Prototype Learning

Methodology – Optimizing Hyper-prototypes

- We introduce **hyper-prototypes**, which 1) leverage a set of learnable class-wise prototypes to adequately capture semantically meaningful knowledge; and 2) optimize them through gradient matching to simulate optimization trajectories of real samples from clients (i.e., *approximating the prototypes that are calculated directly using all clients' data samples*).

average gradients of class **c** on client **k**

$$\mathbf{g}_k^c = \frac{1}{n_k^c} \sum_{(x_i, y_i) \in \mathcal{D}_k^c} \nabla_{z_i} \mathcal{L}_k(x_i, y_i) \in \mathbb{R}^d,$$

$$\nabla_{z_i} \mathcal{L}_k(x_i, y_i) = \left(\frac{\partial h_k}{\partial z_i}\right)^\top \nabla_{h_k} \mathcal{L}_{CE}(h_k(z_i), y_i),$$

averaging over all participating clients

$$\mathbf{g}^c = \frac{1}{|\mathcal{A}^c|} \sum_{k \in \mathcal{A}^c} \mathbf{g}_k^c \in \mathbb{R}^d,$$

$$\mathcal{G} = [\mathbf{g}^1, \ldots, \mathbf{g}^c, \ldots, \mathbf{g}^{\mathbb{C}}] \in \mathbb{R}^{\mathbb{C} \times d},$$

matching

We initialize hyper-prototypes: $\{\mathbf{s}_i^c\}|_{i=1}^{|\mathcal{I}|} \in \mathbb{R}^{|\mathcal{I}| \times d}$
Then, we calculate gradients of hyper-prototypes via a virtual loss function and label:

$$\mathbf{g}_{\text{HP}}^c = \frac{1}{|\mathcal{I}|} \sum_{i=1}^{|\mathcal{I}|} \nabla_{\mathbf{s}_i^c} \mathcal{L}_{vir}(h^*(\mathbf{s}_i^c), y_{vir}) \in \mathbb{R}^d,$$

$$\mathcal{G}_{\text{HP}} = [\mathbf{g}_{\text{HP}}^1, \ldots, \mathbf{g}_{\text{HP}}^c, \ldots, \mathbf{g}_{\text{HP}}^{\mathbb{C}}] \in \mathbb{R}^{\mathbb{C} \times d},$$

We set $\mathcal{L}_{vir}$ as the cross-entropy loss & y_{vir} as the corresponding label c

$$\mathcal{L}_{GM}(\mathbf{g}^c, \mathbf{g}_{\text{HP}}^c) = 1 - \frac{\mathbf{g}^c \cdot \mathbf{g}_{\text{HP}}^c}{\|\mathbf{g}^c\|_2 \times \|\mathbf{g}_{\text{HP}}^c\|_2},$$

Methodology – Hyper-prototypes vs. Prototypes

- After optimizing the hyper-prototypes on M rounds, we can obtain the updated hyper-prototypes as:
  - $\mathcal{S}_M = [\{\mathbf{s}_i^1\}|_{i=1}^{|\mathcal{I}|}, \ldots, \{\mathbf{s}_i^c\}|_{i=1}^{|\mathcal{I}|}, \ldots, \{\mathbf{s}_i^{\mathbb{C}}\}|_{i=1}^{|\mathcal{I}|}]$,
  - Note that the optimized hyper-prototypes serves as the initialization for the next round of training.
- The hyper-prototypes achieve larger inter-class separation and tighter intra-class compactness, even across multiple domains.
- Optimizing $\mathcal{L}_{GM}$ successfully makes the gradients of hyper-prototypes close to the gradients of real samples, indicating a good approximation to the centralized prototypes.

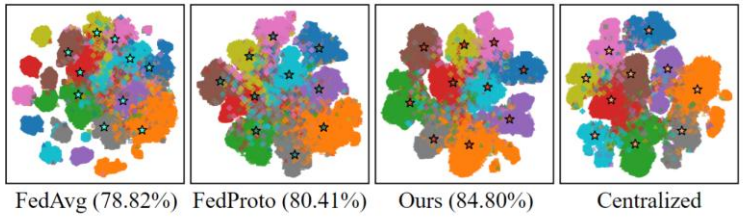


Figure 3. **Visualization for the representation space of different prototypes** on Digits (Peng et al., 2019). Each color (■ ■) indicates one class, and each shape (■ ●) denotes one domain. Centralized is trained on all clients' samples, as an upper-bound reference for prototype quality. The global prototypes of FedAvg and FedProto fail to describe diverse domain information, while our hyper-prototypes promote better semantic consistency across multiple domains. Please zoom in to view details.

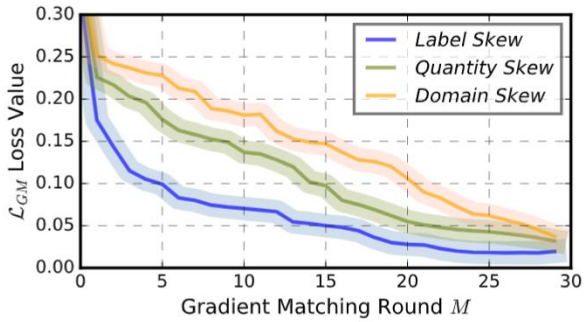
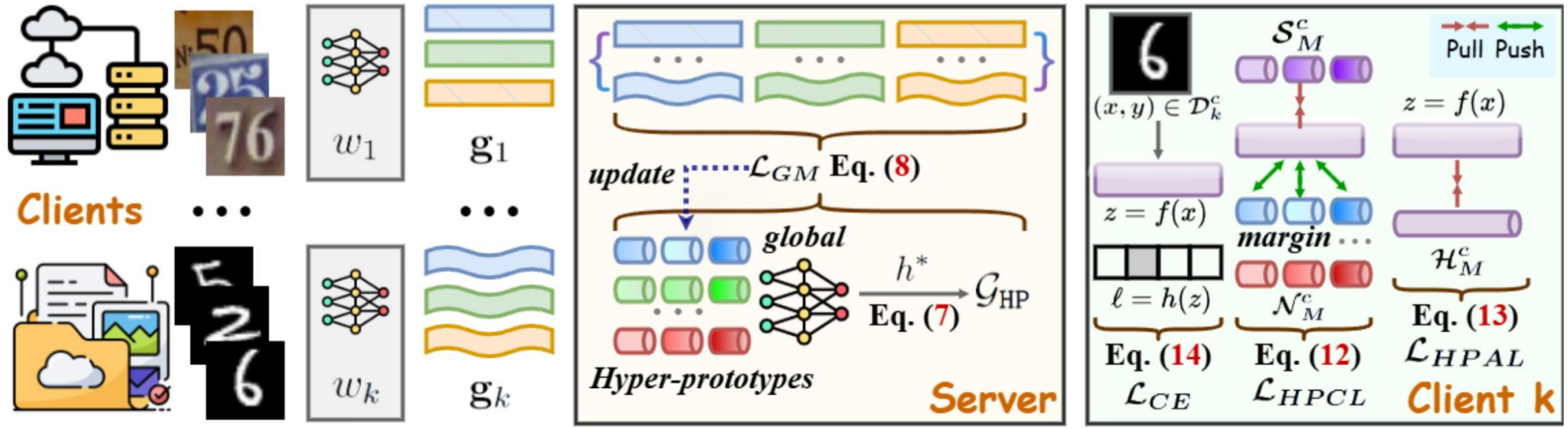


Figure 4. **Loss trend of** $\mathcal{L}_{GM}$ (Equation (8)) under different FL scenarios: CIFAR10 (Krizhevsky et al., 2009) with $\texttt{NID1}_{0.5}$ (label skew), CIFAR10-LT (Krizhevsky et al., 2009) with $\rho = 50$ (quantity skew), Digits (Peng et al., 2019) (domain skew).

Methodology – Overview of FedHPro

- Framework illustration of **Fed**erated **H**yper-**Pro**totype Learning (FedHPro)
  - First, the clients upload the gradients to the server. Based on these gradients from local clients, we leverage a set of learnable units to instantiate the hyper-prototypes.
  - Then, by incorporating the optimized hyper-prototypes, we construct Hyper-Prototype Contrastive Learning (HPCL) with client-specific margin to promote inter-class separability. Besides, we design Hyper-Prototype Alignment Learning (HPAL) to impose intra-class uniformity by smoothly penalizing deviations at the feature-level.



Methodology – Hyper-Prototype Contrastive Learning (HPCL)

- We devise **H**yper-**P**rototype **C**ontrastive **L**earning (HPCL) with client-specific margin to facilitate inter-class separability while preserving semantic meaning.
- We first calculate a margin among local prototypes:
  - $d_k = \frac{1}{(\mathbb{C}-1)^2} \sum_{c_1} \sum_{c_2 \neq c_1} D_{L_2}(\mathbf{p}_k^{c_1}, \mathbf{p}_k^{c_2}),$
  - The client-specific margin adaptively guides each sample's embedding toward a more compact cluster center.
- On this basis, we aim to enforce sample's embedding to be similar to respective hyper-prototypes and dissimilar to the hyper-prototypes of other classes.
  - We define the similarity between sample's embedding and hyper-prototypes as: $s(z_i, \mathcal{S}_M^c) = \frac{1}{|\mathcal{I}|} \sum_{j=1}^{|\mathcal{I}|} \frac{z_i \cdot \mathbf{s}_j^c}{\|z_i\|_2 \times \|\mathbf{s}_j^c\|_2}.$
  - Next, we introduce the following objective term:

$$\mathcal{L}_{HPCL} = \log(1 + \frac{\sum_{\mathcal{S}_M^j \in \mathcal{N}_M^c} \exp((s(z_i, \mathcal{S}_M^j) + d_k)/\tau)}{\exp(s(z_i, \mathcal{S}_M^c)/\tau)}),$$

τ denotes a temperature parameter to control the representation strength

Methodology – Hyper-Prototype Alignment Learning (HPAL)

- Despite the local model pulling sample's embedding toward the corresponding hyper-prototypes, representation inconsistency may still arise due to the intrinsic heterogeneity across clients.
- We design **H**yper-**P**rototype **A**lignment **L**earning (HPAL) to impose intra-class uniformity among clients by smoothly penalizing deviations at the feature-level.

averaged hyper-prototypes

$$\mathcal{H}_M^c = \frac{1}{|\mathcal{I}|}\sum_{i=1}^{|\mathcal{I}|}\mathbf{s}_i^c \in \mathbb{R}^d \quad \Rightarrow \quad \mathcal{L}_{HPAL} = \sum_{q=1}^{d}\begin{cases}\frac{1}{2}(z_{i(q)} - \mathcal{H}_{M(q)}^c)^2; \text{ if } |z_{i(q)} - \mathcal{H}_{M(q)}^c| \leq 1, \\ |z_{i(q)} - \mathcal{H}_{M(q)}^c| - \frac{1}{2}, \text{ otherwise.}\end{cases}$$

q indexes the dimension

⇓

Training Objective

$$\mathcal{L}_{CE} = -\mathbf{1}_{y_i}\log(\delta(h_k(z_i))),$$
$$\mathcal{L} = \mathcal{L}_{CE} + \mathcal{L}_{HPCL} + \mathcal{L}_{HPAL}.$$

Methodology – Pseudo-code Flow

- Server-Side: we optimize the simulated hyper-prototypes via gradient matching;
- Client-Side: we leverage the hyper-prototypes to promote FL local training.

Algorithm 1 FedHPro: Federated Hyper-Prototype Learning

Input: Communication Rounds R, Local Epochs E, Clients K, k-th client's data $\mathcal{D}_k$ and model w_k
Output: The final FL global model w_R^*

/* **Server-Side Executing** */
for each communication round $r = 1, 2, ..., R$ **do**
  $\mathcal{A}_r \Leftarrow \{1, ..., K\}$ // randomly select clients
  for each client $k \in \mathcal{A}_r$ **in parallel do**
    $w_k, \mathbf{g}_k \leftarrow$ **LocalUpdating**$(w_r^*, \mathcal{S}_M)$
  end
  $w_{r+1}^* \leftarrow \sum_{k=1}^{|\mathcal{A}_r|} \frac{n_k}{N} w_k$ // global aggregation
  /* optimizing hyper-prototypes (Section 2.2) */
  $\mathcal{G} = [\mathbf{g}^1, \ldots, \mathbf{g}^c, \ldots, \mathbf{g}^{\mathbb{C}}] \leftarrow$ Equation (6)
  $\mathcal{G}_{\text{HP}} = [\mathbf{g}^1_{\text{HP}}, \ldots, \mathbf{g}^c_{\text{HP}}, \ldots, \mathbf{g}^{\mathbb{C}}_{\text{HP}}] \leftarrow$ Equation (7)
  $\mathcal{L}_{GM}(\mathcal{G}, \mathcal{G}_{\text{HP}}) \leftarrow$ Equation (8) // gradient matching
  $\mathcal{S}_M = [\mathcal{S}_M^1, \ldots, \mathcal{S}_M^c, \ldots, \mathcal{S}_M^{\mathbb{C}}] \leftarrow$ Equation (9)
end

/* **Client-Side Updating** */
LocalUpdating$(w_r^*, \mathcal{S}_M)$:
$w_k \leftarrow w_r^*$ // distribute global parameters
for each local epoch $e = 1, 2, ..., E$ **do**
  for each batch $b \in$ private data $\mathcal{D}_k$ **do**
    $\mathcal{L}_{CE} = \sum_{i \in b} -\mathbf{1}_{y_i} \log(\delta(h_k(z_i))) \leftarrow$ Equation (14)
    /* Hyper-Prototype Contrastive Learning (Section 3.1) */
    $d_k \leftarrow$ Equation (10) // client margin
    $\mathcal{L}_{HPCL}(z_i, \mathcal{S}_M^c, \mathcal{N}_M^c, d_k)_{i \in b} \leftarrow$ Equation (12)
    /* Hyper-Prototype Alignment Learning (Section 3.2) */
    $\mathcal{L}_{HPAL}(z_i, \mathcal{H}_M^c)_{i \in b} \leftarrow$ Equation (13)
    $\mathcal{L} = \mathcal{L}_{CE} + \mathcal{L}_{HPCL} + \mathcal{L}_{HPAL}$
    $w_k \leftarrow w_k - \eta \nabla \mathcal{L}(w_k; b)$ // update model
  end
end
$\mathbf{g}_k = \{\}$ // initialize gradients
for each class $c = 1, 2, ..., \mathbb{C}$ **do**
  $\mathbf{g}_k^c = \frac{1}{n_k^c} \sum_{(x_i, y_i) \in \mathcal{D}_k^c} \nabla_{z_i} \mathcal{L}_k(x_i, y_i) \leftarrow$ Equation (5)
  $\mathbf{g}_k = \mathbf{g}_k \cup \{\mathbf{g}_k^c\}$
end
return $w_k, \mathbf{g}_k$ to the server

Experiments

Experiments – Main Results

- We can observe that FedHPro consistently outperforms eight SOTA baselines, confirming the significance of the hyper-prototypes in capturing consistent semantic representations and their effectiveness when utilized via the HPCL and HPAL modules.

Table 1. **Comparison results** under label and quantity skew. † denotes the results obtained by exchanging both the prototypes and model parameters (please see analysis in Appendix Table A7) during the FL training. Best results are in **red bold**, with second best in **blue bold**.

METHODS	CIFAR10			HAM10000			TINYIMAGENET			CIFAR10-LT ($\text{NID1}_{0.5}$)		
	$\text{NID1}_{0.2}$	$\text{NID1}_{0.5}$	NID2	$\text{NID1}_{0.2}$	$\text{NID1}_{0.5}$	NID2	$\text{NID1}_{0.2}$	$\text{NID1}_{0.5}$	NID2	$\rho = 10$	$\rho = 50$	$\rho = 100$
FedAvg [AISTATS'17]	80.59	85.60	74.48	45.26	48.68	41.54	41.34	43.88	36.20	75.54	69.05	60.79
FedProx [MLSys'20]	80.74	85.54	74.67	45.42	48.55	41.21	41.16	43.50	36.62	75.15	69.23	60.86
MOON [CVPR'21]	82.04	86.95	76.07	47.33	50.24	43.10	43.22	44.36	37.15	76.11	69.70	60.91
FedProto† [AAAI'22]	81.61	86.25	75.32	46.35	49.70	42.45	42.25	44.19	36.86	75.81	69.43	60.37
FedTGP† [AAAI'24]	84.14	87.31	**77.95**	48.29	50.77	44.26	43.92	45.13	38.21	76.27	71.36	61.94
FedGMKD [NeurIPS'24]	83.56	**88.09**	77.29	47.97	50.49	44.20	44.09	45.30	37.94	**76.83**	71.21	62.13
FedRCL [CVPR'24]	83.80	87.76	77.51	**48.60**	51.10	44.36	**44.30**	45.24	**38.76**	76.60	71.55	62.41
FedSA† [AAAI'25]	**84.27**	87.93	77.86	48.45	**51.28**	**44.85**	44.18	**45.56**	38.35	76.75	**72.30**	**62.48**
FedHPro (Ours)	**85.98**	**89.56**	**79.70**	**50.23**	**52.79**	**46.17**	**45.64**	**46.90**	**40.52**	**78.62**	**74.69**	**64.75**

Experiments – Some Ablations

Table 6. **Comparison for hyper-prototypes and global prototypes** ($\mathbb{P}$ in Equation (4)), $\Rightarrow$ means replace with $\mathbb{P}$ in the loss.

Using Prototypes	Digits		Office-Caltech		CIFAR10-LT	
	AVG ↑	△	AVG ↑	△	ACC ↑	△
Hyper-Prototypes	**84.80**	–	**64.52**	–	**74.69**	–
Using $\mathbb{P} \Rightarrow$ HPCL	83.72	-1.08	63.15	-1.37	73.22	-1.47
Using $\mathbb{P} \Rightarrow$ HPAL	83.29	-1.51	62.42	-2.10	72.89	-1.80
Using $\mathbb{P} \Rightarrow$ Both	81.35	-3.45	59.69	-4.83	72.56	-2.13

- The hyper-prototypes consistently improve the accuracy of these methods in multiple domains. This further confirms that our hyper-prototypes preserve better semantic consistency than the conventional prototypes during FL training.
- Tab.5 shows: 1) HPCL brings significant gains over the baseline, indicating that HPCL is able to promote inter-class separability while preserving semantics; 2) HPAL also yields solid improvements, proving the importance of aligning embeddings across clients via smooth regularization.
- In Tab.6, we use global prototypes to replace hyper-prototype in the loss term. It is evident that using global prototypes largely weakens both HPCL and HPAL, indicating the superiority of the hyper-prototypes over the conventional prototypes from the proposed module perspective.

Table 3. **Enabling existing prototype-based methods**, in which their prototypes are replaced with our hyper-prototypes.

Using HP	Digits				
	MNIST	USPS	SVHN	SYN	AVG ↑
FedProto†	97.48 ↑0.08	92.24 ↑0.35	80.90 ↑1.68	56.39 ↑3.24	81.75 ↑1.34
FedTGP†	97.83 ↑0.10	92.67 ↑0.24	82.49 ↑1.14	57.46 ↑2.27	82.61 ↑0.94
FedGMKD	97.85 ↑0.05	92.71 ↑0.45	82.52 ↑1.39	58.04 ↑1.87	82.79 ↑0.96
FedSA†	97.98 ↑0.20	92.79 ↑0.29	82.93 ↑1.16	59.30 ↑1.18	83.25 ↑0.71

Table 5. **Ablation study for two modules** of FedHPro: HPCL and HPAL, on the Digits (**Top**) and Office-Caltech (**Bottom**).

HPCL	HPAL	MNIST	USPS	SVHN	SYN	AVG ↑
✗	✗	96.68	90.43	76.35	51.85	78.82
✓	✗	98.06	91.74	83.81	60.74	83.58
✗	✓	98.38	91.93	84.16	61.30	83.94
✓	✓	98.52	93.13	84.95	62.59	**84.80**
HPCL	**HPAL**	Caltech	Webcam	Amazon	DSLR	AVG ↑
✗	✗	60.71	48.90	75.79	36.28	55.42
✓	✗	63.27	57.10	79.64	43.67	60.92
✗	✓	64.16	55.35	79.11	46.75	61.34
✓	✓	64.61	62.45	80.69	50.33	**64.52**

Thank You for Your Listening!

Huan Wang

- University of Wollongong & Singapore Management University
- supervised by Prof. Guansong Pang (SMU) & Prof. Jun Shen (UOW)
- If you have any question, or interest in ***FL / Anomaly Detection / Agentic Reasoning Safety***, please feel free to contact me: hw226@uowmail.edu.au