Appendix

A. Notations

The notations used in this paper are listed in Tab. 7.

B. Details of Method

B.1. Algorithm

Algorithm 1 outlines the comprehensive training procedure for our suggested FedWCA, while Algorithm 2 details the local adaptation process.

B.2. Soft Neighborhood Density

We review Soft Neighborhood Density (SND) [41] used to assess the overall model efficacy in our FedWCA (see Eq. (3)). SND is originally proposed as a method for evaluating models in an unsupervised manner by analyzing how densely data points cluster together using the overall models' outputs. It defines 'soft neighborhoods' of a data point by the distribution of its similarity to other points, and measures density as the entropy of this distribution.

Let $h = g \circ f$ and $D = \{x_i\}_{i=1}^{N}$ be an evaluated model and an unlabeled dataset, respectively. SND first computes the similarity between samples. Let $Q_{i,j} = \cos(h(x_i), h(x_j))$ be the (i,j) element of the similarity matrix, where $\cos(\cdot,\cdot)$ is the cosine similarity. The diagonal elements of Q are ignored to compute the distance to neighbors for each sample. Q is then converted into a probabilistic distribution P using the scaling temperature parameter T and the softmax function:

$$P_{i,j} = \frac{\exp(Q_{i,j}/T)}{\sum_{j'} \exp(Q_{i,j'}/T)}. \tag{5}$$

The temperature is set to 0.05 in the original paper and we use it. We finally obtain the SND value of h by computing the entropy for each row of P (*i.e.*, each sample) and taking the average of all samples:

$$S(h) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{N} P_{i,j}\log P_{i,j}. \tag{6}$$

See the original paper for detailed explanation.

B.3. Cluster Weights

We provide the final specific cluster weights $\boldsymbol{v}_k = (v_{k,1}, \ldots, v_{k,C})$ in the initial model $f_k^{\text{init}} = \sum_c v_{k,c} f_c$. According to Sec. 4.2, the initial model for client k is computed as follows:

$$\begin{aligned}
&f_k^{\text{init}} \qquad\qquad (7)\\
&= \beta_{k,0} f_{c_k} + \beta_{k,1}\sum_c \alpha_{k,c}\tilde{f}_c\\
&= \beta_{k,0} f_{c_k} + \beta_{k,1}\sum_c \alpha_{k,c}\{B_{c,0} f_c + B_{c,1}\sum_{c'} A_{c'\to c} f_{c'}\}\\
&= \sum_c \{\mathbb{1}(c = c_k)\beta_{k,0} + \beta_{k,1}\alpha_{k,c}B_{c,0}\\
&\qquad\qquad + \beta_{k,1}\sum_{c'} B_{c',1}\alpha_{k,c'}A_{c\to c'}\} f_c,
\end{aligned}$$

where $\mathbb{1}$ is the indicator function. The third equation can be obtained by replacing the subscripts c and c'. We thus obtain the cluster weights $\boldsymbol{v}_k$ as:

$$\begin{aligned}
v_{k,c} = {} & \mathbb{1}(c = c_k)\beta_{k,0} + \beta_{k,1}\alpha_{k,c}B_{c,0}\\
& + \beta_{k,1}\sum_{c'} B_{c',1}\alpha_{k,c'}A_{c\to c'}.
\end{aligned} \tag{8}$$

B.4. Loss Function of SHOT

We review the loss function of SHOT [29], adopted as a loss function of our method because of its simplicity. The loss function comprises the cross-entropy loss for the pseudo-labeled dataset and an Information Maximization (IM) loss [13] for the unlabeled dataset. The IM loss promotes confident model outputs and counteracts a bias towards any single class by discouraging trivial score distributions. The total loss function $\mathcal{L}(h)$ for the model h weights these two loss functions with a balancing parameter λ:

$$\begin{aligned}
&\mathcal{L}(h) = \mathcal{L}_{\text{IM}}(h; D) + \lambda\mathcal{L}_{\text{CE}}(h; \hat{D});\\
&\mathcal{L}_{\text{CE}}(h; \hat{D}) = -\frac{1}{N}\sum_{(x,\hat{y})\in\hat{D}} \log\,(h(x))_{\hat{y}},\\
&\mathcal{L}_{\text{IM}}(h; D) = \sum_{m=1}^{M} \hat{o}_m \log \hat{o}_m\\
&\qquad\qquad - \frac{1}{N}\sum_{m=1}^{M}\sum_{x\in D}(h(x))_m \log\,(h(x))_m,
\end{aligned} \tag{9}$$

where N is the dataset size and $\mathcal{L}_{\text{CE}}(h; \hat{D})$ represents the cross-entropy loss calculated over the pseudo-labeled dataset $\hat{D}$, which encourages the model h to align with the pseudo-labels. The IM loss $\mathcal{L}_{\text{IM}}(h; D)$, on the other hand, comprises two terms: (1) the negative entropy of the mean output score $\hat{o} = \sum_{x\in D} h(x)/N$ and (2) the entropy of the model's output scores for each data point in the unlabeled dataset D.

Table 7. **Notation in the paper.**

Symbol/Notation	Definition
Problem	
$\mathcal{X}$	feature space
$\mathcal{Y}$	label space
M	number of classes indexed by m
K	number of clients indexd by k
R	number of communication rounds indexed by r
E	number of local epochs for each client
$D_k = \{x_i\}_{i=1}^{N_k}$	unlabeled dataset of client k with N_k samples
$\mathcal{P}_k(\mathcal{X})$	data distribution of client k
$f : \mathcal{X} \to \mathbb{R}^q$	feature extractor
$g : \mathbb{R}^q \to \mathbb{R}^M$	classifier
$h = g \circ f$	classification model
$h_S = g_S \circ f_S$	source model
$W = [w_1, \dots, w_M] \in \mathbb{R}^{q \times M}$	classifier weight for g_S
$h_k = g_k \circ f_k$	locally trained model by client k
Method	
$A(k, l)$	adjacency matrix for client clustering
κ_k	nearest neighbor client for client k
C	number of clusters indexed by c
c_k	cluster index assigned to client k
f_c	cluster model for cluster c
$\tilde{f}_c$	soft cluster model for cluster c: $\tilde{f}_c = B_{c,0} f_c + B_{c,1} \sum_{c'} A_{c' \to c} f_{c'}$
$\bar{f}_k$	locally combined model by client k using soft cluster models: $\bar{f}_k = \sum_c \alpha_{k,c} \tilde{f}_c$
f_k^{init}	initial model of client k for each round: $f_k^{\text{init}} = \beta_{k,0} f_{c_k} + \beta_{k,1} \sum_c \alpha_{k,c} \tilde{f}_c$
$m(i)$	class whose classifier weight vector is closest to $f_c(x_i)$: $m(i) = \arg\max_m \cos(f_c(x_i), w_m)$
$\boldsymbol{v}_k = (v_{k,1}, \dots, v_{k,C})$	cluster weights for cluster models: $f_k^{\text{init}} = \sum_c v_{k,c} f_c$
$\boldsymbol{\alpha}_k = (\alpha_{k,1}, \dots, \alpha_{k,C})$	locally calculated cluster weights for soft cluster models
$\boldsymbol{\beta}_k = (\beta_{k,0}, \beta_{k,1})$	pseudo-performance weights for f_{c_k} and $\bar{f}_k$
$A_{c' \to c}$	benefit metric indicating the relative advantage of cluster c for clients in cluster c'
$B_{c,0}, B_{c,1}$	coefficients balancing the emphasis between f_c and $\sum_{c'} A_{c' \to c} f_{c'}$
T_a, T_b	temperature parameters controlling $\boldsymbol{\alpha}_k$ and $\boldsymbol{\beta}_k$, respectively
λ, μ	balancing parameters for the loss function and mixup, respectively
$\hat{D}_k = \{x_i, \hat{y}_i\}_{i=1}^{N_k}$	pseudo-labeled dataset of client k obtained by a prototype-based pseudo-labeling and mixup

C. Detailed Experimental Settings

C.1. Datasets

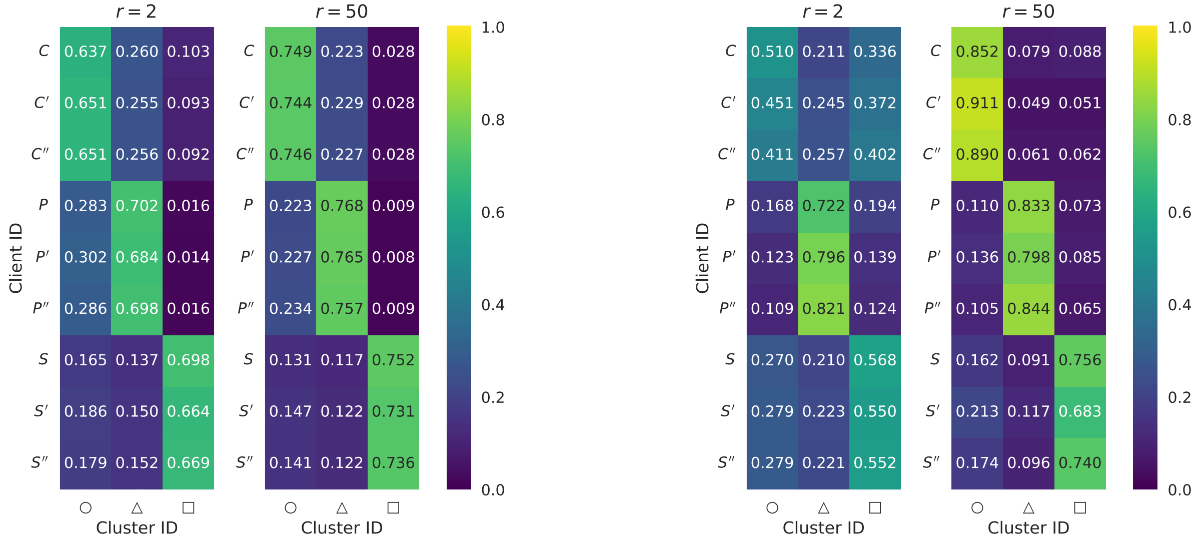
Tab. 8 summarizes the details of datasets used in our experiments. For Digit-Five, we used the entire dataset for USPS and 34,000 randomly selected samples for the other domains. For each dataset, one domain was selected as the source, and the remaining domains served as target domains. Data samples from each target domain were distributed to 8 clients for Digit-Five and 3 clients for PACS and Office-Home per domain, all in an i.i.d. manner. Each client's data were divided into three subsets: 20% for testing, 64% for training, and 16% for validation.

C.2. Implementation Details

For the implementation, we set the local epoch to $E = 5$ and total communication rounds to $R = 100$ for Digit-Five and PACS, and $R = 50$ for Office-Home, ensuring convergence of each method's learning. In the case of FedPCL+PL, we limited R to 20, due to overfitting observed in this method on the PACS and Office-Home datasets. The stochastic gradient descent (SGD) was used with the best learning rate of 10^{-3} for Digit-Five and 10^{-4} for PACS and Office-Home in all methods, which are selected from $\{10^{-i} | i \in \{1, 2, 3, 4, 5\}\}$. For source model training, we set the learning rate to 0.001 in all datasets. Additionally, we adopted a weight decay of 0.001 and a momentum of 0.9, in line with standard SFDA studies [29]. In LADD's implementation, we randomly selected 100 samples from each client's target data for computing style features, focusing on the central 5% region of the frequency spectrum. Other hyperparameters for LADD were also appropriately tuned. The temperature parameters of FedPCL were set to 70 as a result of tuning. For our methods, the temperature parameters were set to $T_b = 0.05$ for all datasets (aligned with SND [41]) and $T_a = 0.01$ for Digit-Five and Office-Home,

Table 8. **Dataset information** including number of samples, classes, and clients for three datasets.

Datasets	Number of samples		Classes	Clients
	Per domain	Total		
Digit-Five	MNIST: 34,000, MNIST-M: 34,000 SVHN: 34,000, SYNTH: 34,000, USPS: 9298	145,298	10	32
PACS	Art Painting: 2,048, Cartoon: 2,344 Photo: 1,670, Sketch: 3,929	9,991	7	9
Office-Home	Art: 2,424, Clipart: 4,365 Product: 4,437, Real-World: 4,353	15,579	65	9



(a) FedWCA-L

(b) FedWCA

Figure 5. **Visualization of cluster weights** for FedWCA-L and FedWCA when the communication round r is 2 and 50. Art Painting in PACS dataset is used for the source dataset. Clients C, C', C'' belong to Cartoon, P, P', P'' to Photo, and S, S', S'' to Sketch. See the first row of Tab. 4 for the cluster IDs (denoted by ◯, △, and □) assigned to each client. The number represents the weight for the corresponding cluster.

Table 9. **Hyperparameters for our FedWCA** including the learning rate (lr), balancing parameter for cross-entropy loss λ, balancing parameter for mixup μ, and temperature parameters T_a, T_b. Only for SYNTH of Digit-Five and Clipart of Office-Home source domains, T_a is set to 0.001 and 0.1, respectively.

Datasets	Hyperparameters				
	lr	λ	μ	T_a	T_b
Digit-Five	0.001	0.1	0.55	0.01	0.05
PACS	0.0001	0.3	0.55	0.1	0.05
Office-Home	0.0001	0.3	0.55	0.01	0.05

except for SYNTH ($T_a = 0.001$) and Clipart ($T_a = 0.1$) source domains, while $T_a = 0.1$ for PACS. The balancing parameter λ was assigned the same value as in the original SHOT paper [29]. In particular, Tab. 9 summarizes the hyperparameters for FedWCA.

C.3. Hyperparameter Tuning

For LADD, we searched the regularization parameters and starting rounds from $\{10^{-i} | i \in \{-2, -1, 0, 1, 2, 3, 4, 5\}\}$ and $\{5, 10, 30, 50, 80, 100\}$, respectively. For FedPCL, the best temperature parameter was selected from $\{7 \times 10^{-i} | i \in \{-2, -1, 0, 1, 2, 3\}\}$ as par the original paper. For our FedWCA, we searched the temperature parameters T_a and T_b from $\{0.001, 0.005, 0.01, 0.05, 0.1\}$.

D. Limitations and Discussions

Complexity of FedWCA. We mention that the derivation of cluster weights in our method, while not computationally intensive, involves a complex procedure. Although Sec. 5.2 demonstrates that simpler weight calculations do not provide sufficient performance, identifying simpler yet effective methods for calculating cluster weights that match

Table 10. **Results for FedWCA modification reducing costs.** The numbers are the mean values ± standard deviations of the averaged accuracy (%) across all clients and all target domains. In the latter $U-1$ rounds of every U round, the initial model of the FedWCA can be computed server-side as opposed to client-side, thereby cutting communication, storage, and computational costs. This revised FedWCA boasts an equivalent performance to its original counterpart when $U=5$.

Datasets		**Methods**	
		Revised FedWCA $U=5$	FedWCA
Digit-Five	MN	73.37 ± 3.65	72.74 ± 3.57
	SV	89.88 ± 2.24	90.13 ± 2.08
	MN-M	83.16 ± 0.56	82.93 ± 0.79
	US	57.33 ± 3.00	58.56 ± 4.26
	SY	85.70 ± 1.05	84.06 ± 1.38
	Avg.	77.89 ± 11.93	77.69 ± 11.47
PACS	Ar	80.89 ± 2.54	80.63 ± 2.44
	Ca	83.23 ± 0.91	83.18 ± 0.86
	Ph	65.32 ± 1.06	65.50 ± 1.24
	Sk	84.28 ± 6.40	84.22 ± 6.60
	Avg.	78.43 ± 8.45	78.38 ± 8.38
Office-Home	Ar	66.13 ± 0.67	66.06 ± 0.50
	Cl	68.33 ± 0.61	68.32 ± 0.42
	Pr	61.30 ± 0.60	61.46 ± 0.88
	Re	67.63 ± 0.29	68.06 ± 0.46
	Avg.	65.85 ± 2.86	65.97 ± 2.83

FedWCA's performance remains a task for future research. However, our method can be slightly modified to reduce computational costs, as discussed below.

Concerns about distributing all cluster models. Our method requires each client to receive all soft cluster models, which may raise some concerns: (1) privacy concerns and (2) additional communication and storage costs.

(1) Distributing cluster models poses minimal privacy risks due to several factors. Initially, we note that the server distributes all ”soft cluster models” to each client, along with the original cluster model of the respective client. Soft cluster models, produced by merging all cluster models, roughly incorporate all clients' local models and thus, privacy risks remain similar to typical FL methods such as FedAvg. Further, distributing the single original cluster model to clients is a standard practice in current clustered FL studies and carries minimal privacy risk. This is due to our algorithm's design, where clients are only privy to their specific cluster IDs with no insight into other cluster details such as cluster size or client membership, even if it is composed of only two clients.

(2) Our weighted cluster aggregation (WCA) costs $C+1$ times higher than FedAvg (C: number of clusters) due to the requirement of each client receiving and storing C soft cluster models and the original cluster model. However, we can save those costs by altering the computation of the client's initial models in WCA as follows. In the first round of every U rounds, clients compute their initial models as the original method does. In the following $U-1$ rounds, the initial models are computed server-side, utilizing identical cluster weights sent by clients in the first round. This modification allows the server to only distribute two models to each client: the computed initial model and the original cluster model, reducing the communication and storage costs by a factor of $2/(C+1)$ in $(U-1)/U$ of all rounds. This can also lessen the client's computational cost for initial model calculations. As demonstrated in Tab. 10, further tests on Digit-Five, PACS, and Office-Home with $U=5$ showed that the revised FedWCA maintains similar average accuracy as the original. Notably, in some source domains like SYNTH, the revised technique improves accuracy. This enhancement is because the use of identical cluster weights reduces overfitting.

Different enhancement according to dataset. As illustrated in Sec. 5.1, the accuracy enhancement achieved by our method varies with the dataset. Specifically, for datasets exhibiting minor domain gaps, FedAvg may suffice to some extent, as the impact of domain shifts on accuracy is minimal. For instance, in the Office-Home dataset, although our method surpasses FedAvg, the superiority margin is less significant than in Digit-Five and PACS.

Extension to other tasks. While our approach is primarily tailored for classification tasks, its underlying principles could theoretically extend to other complex vision tasks, like object detection. However, direct extrapolation may encounter challenges, such as convergence of feature vectors towards classifier vectors in cluster weight calculations. Adapting and assessing our method in these diverse contexts constitutes an avenue for future research.

E. Supplementary Experimental Results

E.1. Visualization of Cluster Weights

Fig. 5 visualizes the cluster weights for our proposed methods FedWCA and FedWCA-L, a variant of FedWCA wherein cluster weights are computed solely locally (see Sec. 4.2). This shows that clients within the same domain obtain similar weights, indicating the efficacy of our cluster weight calculation based on Eq. (1). FedWCA, in particular, promotes inter-domain collaboration early in learning ($r=2$), notably between Cartoon and Sketch domains. Specifically, Cartoon domain clients (C, C', C'') have significantly higher weights for cluster □ (Sketch) in FedWCA compared to FedWCA-L. This is the result of the Cartoon domain clients considering the benefits for the clients in cluster □. As learning progresses ($r=50$), FedWCA clients increasingly concentrate on their respective clusters, highlighting the method's balance between overall and individual advantages. As a result, this approach leads to notable accuracy improvements for both Cartoon and Sketch clients, as depicted in Fig. 4 (b).

Table 11. **Full results for layer dependence of clustering.** Art Painting of PACS is used as the source domain, and ResNet-18 is employed. Clients C, C', C'' belong to Cartoon, P, P', P'' to Photo, and S, S', S'' to Sketch. The table below reports the cluster IDs (denoted by ○, △, and □) assigned to each client. Ideally, each client group (C, C', C''), (P, P', P''), and (S, S', S'') should be grouped together, as is achieved by the first and second layers.

	Clients								
Layer	C	C'	C''	P	P'	P''	S	S'	S''
1st	○	○	○	△	△	△	□	□	□
2nd	○	○	○	△	△	△	□	□	□
3rd	○	○	○	○	○	○	○	○	○
4th	○	○	○	○	○	○	△	△	△
5th	○	○	○	○	○	○	○	○	○
6th	○	○	○	△	△	△	□	□	□
7th	○	○	○	○	○	○	○	○	○
8th	○	○	○	○	○	○	○	○	○
9th	○	○	○	○	○	○	○	○	○
10th	○	○	○	○	○	○	△	△	△
11th	○	○	○	○	○	○	○	○	○
12th	○	○	○	○	○	○	○	△	△
13th	○	○	○	○	○	○	○	○	○
14th	○	○	○	△	△	△	□	□	□
15th	○	○	○	○	○	○	○	○	○
16th	○	○	○	○	○	○	△	△	△
17th	○	○	○	○	○	○	○	○	○
All	○	○	○	○	○	○	○	○	○

E.2. Results of Client Clustering

Tab. 11 shows the full results for the layer dependence of client clustering in our method with the PACS dataset.

Additionally, Tab. 12 presents the clustering results for Digit-Five across various source domains using our FINCH method. Fig. 6 shows the specific images possessed by clients in each cluster particularly when using USPS as the source domain. The results from Tab. 12 indicate that, unlike PACS and Office-Home, the clustering for Digit-Five contains some errors, except for the SVHN source domain. To this end, we compare our standard FINCH algorithm, against two alternatives on Digit-Five: (1) LADD clustering based on the shared style features and (2) a hypothetical 'true' clustering based on actual domain information. It must be emphasized that LADD requires a specified number of iterations and a search range for clusters, whereas FINCH necessitates no hyperparameters. Tab. 13 shows the accuracy of FedWCA when applying each clustering algorithm. Notably, FINCH performs comparably to both the true clustering and LADD, with no need for sharing prior information on the clusters or any information other than the model parameters. This highlights our method's strength in enhancing client performance, even when assigned to incorrect clusters, by effectively combining all cluster models using individualized cluster weights.

Table 12. **Results of client clustering in Digit-Five** for each source domain. The clustering algorithm FINCH is applied to Lenet. Each row represents client IDs. Clients 1 to 8, 9 to 16, 17 to 24 and 25 to 32 each belong to the same domain. The table below reports the cluster IDs (denoted by, *e.g.*, ○, △, □, and ♢) assigned to each client. Ideally, each client group (1,2,3,4,5,6,7,8), (9,10,11,12,13,14,15,16), (17,18,19,20,21,22,23,24), and (25,26,27,28,29,30,31,32) should be grouped together, as is achieved by SVHN source domain.

Client ID	**Source domains**				
	MN	SV	MN-M	US	SY
1	○	○	○	○	○
2	○	○	○	○	○
3	○	○	○	○	○
4	○	○	○	○	○
5	○	○	○	○	○
6	○	○	○	○	○
7	△	○	○	○	○
8	△	○	○	○	○
9	□	△	△	△	△
10	□	△	△	△	△
11	□	△	△	△	△
12	□	△	△	△	△
13	♢	△	△	△	△
14	♢	△	△	△	△
15	♡	△	△	△	△
16	♡	△	△	△	△
17	★	□	□	□	□
18	★	□	□	□	□
19	★	□	□	□	□
20	★	□	□	□	□
21	★	□	□	□	♢
22	★	□	♢	♢	♢
23	♣	□	♢	♢	♢
24	♣	□	♢	♢	♢
25	♠	♢	♡	♡	△
26	♠	♢	♡	♡	△
27	♠	♢	♡	♡	△
28	♠	♢	♡	♡	△
29	♠	♢	♡	♡	△
30	♠	♢	♡	♡	△
31	♠	♢	♡	♡	△
32	♠	♢	♡	♡	△

Table 13. **Ablation on clustering algorithm.** Three clustering algorithms are applied to FedWCA. 'True' uses the hypothetical true clusters based on the domain labels. Even without specific cluster information, FedWCA offers similar performance to 'True'.

Clustering	**Digit-Five**					
	MN	SV	MN-M	US	SY	**Avg.**
FINCH	72.74	90.13	83.93	58.56	84.94	77.86
LADD	72.33	89.89	81.99	58.72	85.58	77.70
True	72.93	90.07	83.43	56.95	85.69	77.81

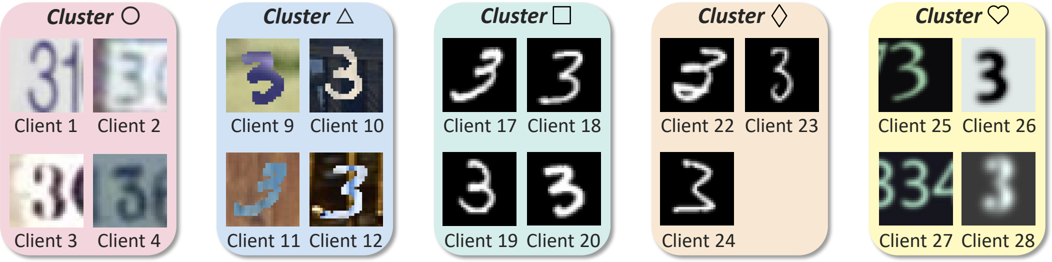


Figure 6. **Images possessed by clients in each cluster.** USPS of Digit-Five serves as the source domain. Our FedWCA clusters clients according to their local models, almost matching their data domains.

Table 14. **Performance comparison with extended FL methods (Digit-Five and PACS).** The existing FL methods, FedProx [27], FedAMP [17], and IFCA [12] are extended to FFREEDA by incorporating **PP**: prototype-based pseudo-labeling, **FP**: fixed pseudo-labels in each round, and **IM**: IM loss. FedWCA outperforms them in all source domains.

		Methods (+PP+FP+IM)			
Datasets		FedProx	FedAMP	IFCA	FedWCA (ours)
Digit-Five	MN	56.01 ± 2.46	51.48 ± 0.44	58.68 ± 2.21	**72.74 ± 3.57**
	SV	83.27 ± 1.80	83.00 ± 0.81	83.39 ± 1.60	**90.13 ± 2.08**
	MN-M	81.27 ± 0.29	72.94 ± 2.09	79.82 ± 0.40	**82.93 ± 0.79**
	US	52.78 ± 0.74	53.81 ± 1.54	55.62 ± 2.99	**58.56 ± 4.26**
	SY	82.21 ± 0.54	82.22 ± 0.32	81.71 ± 0.75	**84.94 ± 1.15**
	Avg.	71.11 ± 13.86	68.69 ± 13.73	71.81 ± 12.30	**77.86 ± 11.56**
PACS	Ar	76.50 ± 1.24	73.54 ± 1.30	75.82 ± 3.11	**80.63 ± 2.44**
	Ca	76.16 ± 0.87	77.55 ± 1.21	79.13 ± 2.57	**83.18 ± 0.86**
	Ph	60.88 ± 0.74	63.70 ± 0.79	64.61 ± 1.57	**65.50 ± 1.24**
	Sk	80.29 ± 7.02	75.84 ± 5.87	80.93 ± 6.59	**84.22 ± 6.60**
	Avg.	73.46 ± 8.29	72.66 ± 6.19	75.13 ± 7.46	**78.38 ± 8.38**

E.3. Comparison with Other Federated Learning Methods

In addition to FedAvg and FedPCL, we extended other FL methods using ground-truth labels to FFREEDA by incorporating a prototype-based pseudo-labeling, fixed pseudo-labels per round, and IM loss, and compared them to our method. We adopted three additional FL methods: (1) FedProx [27], (2) FedAMP [17], and (3) IFCA [12]. FedProx modifies FedAvg by adding the proximal term to address data heterogeneity. FedAMP personalizes FL by creating a client-specific cloud model weighted on the similarity of each client's model parameters, with clients subsequently training personalized models based on this cloud model. IFCA, a clustered FL method that requires setting the number of clusters, has each client calculate the loss for all cluster models and train the model with the lowest loss per round.

The performance comparison is shown in Tab. 14. Our FedWCA outperforms all other methods n all source domains for Digit-Five and PACS, highlighting that merely extending existing FL methods to FFREEDA is insufficient and confirming our method's effectiveness. FedProx uses basic averaging for aggregation, FedAMP's parameter similarity weighting is unsuitable for our unlabeled adaptation, and IFCA's cluster model selection is unstable due to the loss calculation based on pseudo-labels.

Algorithm 1: FedWCA (Federated learning with weighted cluster aggregation)

```
Input: $K$ clients, $k$-th client's local unlabeled data $D_k = \{x_i\}_{i=1}^{N_k}$, $k$-th client's initial local model
       $h_k^{\text{init}} = g_k^{\text{init}} \circ f_k^{\text{init}}$, source model $h_S = g_S \circ f_S$, local epochs $E$, total communication round $R$, temperature
       parameters $T_a$, $T_b$, balancing parameters $\lambda$, $\mu$
Output: $K$ personalized models $h_k = g_S \circ f_{c_k}$
for $r = 0, \dots R-1$ do
    if $r = 0$ then
        Client:
            Initialize the local model: $h_k^{\text{init}} = g_k^{\text{init}} \circ f_k^{\text{init}} \leftarrow h_S = g_S \circ f_S$
            $f_k$ = ClientLocalAdaptation($r$, $D_k$, $f_k^{\text{init}}$, $g_k^{\text{init}}$, $E$, $\lambda$)
            Send $f_k$ to the server
        Server:
            Cluster $K$ clients into $C$ clusters based on the first layers' parameters of $f_k$ by using FINCH: $k \mapsto c_k$
            Create cluster models $f_c$ by averaging the local models $f_k$ in each cluster $c \in \{1, \dots, C\}$
            Initialize soft cluster models $\tilde{f}_c = f_c$
    else
        Client:
            Calculate $\boldsymbol{\alpha}_k$ for $\tilde{f}_c$ and $\boldsymbol{\beta}_k$ for $f_{c_k}$ and $\bar{f}_c = \sum_c \alpha_{k,c}\tilde{f}_c$ by Eq. (1) and Eq. (3)
            Set an initial model: $f_k^{\text{init}} = \beta_{k,0} f_{c_k} + \beta_{k,1} \sum_c \alpha_{k,c}\tilde{f}_c$
            $f_k$ = ClientLocalAdaptation($r$, $D_k$, $f_k^{\text{init}}$, $f_{c_k}$, $g_k^{\text{init}}$, $E$, $T_a$, $T_b$, $\lambda$, $\mu$)
            Send $f_k$, $\boldsymbol{\alpha}_k$, and $\boldsymbol{\beta}_k$ to the server
        Server:
            Update cluster models $f_c$ by averaging the local models $f_k$ in each cluster $c \in \{1, \dots, C\}$
            Calculate $A_{c \to c'}$ and $B_{c,i}$ by averaging $\alpha_{k,c'}$ and $\beta_{k,i}$ across clients within each cluster $c$
            Update soft cluster models: $\tilde{f}_c = B_{c,0} f_c + B_{c,1} \sum_{c' \in \mathcal{C}} A_{c' \to c} f_{c'}$
    Server: Send $f_{c_k}$ and $\tilde{f}_c$ for every $c$ to the client $k$
```

Algorithm 2: ClientLocalAdaptation

```
Input: Current round $r$, unlabeled data $D = \{x_i\}_{i=1}^N$, initial model $f^{\text{init}}$, cluster model $f_c$, classifier $g$, local epochs
    $E$, temperature parameters $T_a, T_b$, balancing parameters $\lambda, \mu$.
Output: Trained model $f$
if $r = 0$ then
    Compute class-wise prototypes $p_m$ for each class $m$ by $g \circ f^{\text{init}}$ and assign each sample $x \in D$ pseudo-labels $\hat{y}$
      via Eq. (4), and generate a pseudo-labeled dataset $\hat{D} = \{x_i, \hat{y}_i\}_{i=1}^N$
else
    Compute class-wise prototypes $p_m$ and $q_m$ for each class $m$ by $g \circ f^{\text{init}}$ and $g \circ f_c$, and assign each sample
      $x \in D$ pseudo-label $\hat{y}_{\text{init}}$ and $\hat{y}_c$ via Eq. (4), respectively
    Compute normalization factors $p$, $q$:
        $$p = \sum_{m \neq m'} \frac{\cos(p_m, p_{m'})}{M(M-1)}, \quad q = \sum_{m \neq m'} \frac{\cos(q_m, q_{m'})}{M(M-1)}$$
    Assign each sample $x \in D$ pseudo-label $\hat{y} = \hat{y}_{\text{init}}$ if $p_m/p \geq q_m/q$, and $\hat{y} = \hat{y}_c$ otherwise
    Generate pseudo-labeled datasets $\hat{D}^{mat}$ and $\hat{D}^{mis}$:
        $$\hat{D}^{mat} = \{(x, \hat{y}) | x \in D \; s.t. \; \hat{y}_{\text{init}} = \hat{y}_c)\}, \quad \hat{D}^{mis} = \{(x, \hat{y}) | x \in D \; s.t. \; \hat{y}_{\text{init}} \neq \hat{y}_c)\}$$
    Generate a mixed dataset $\hat{D}^{mix}$ and a pseudo-labeled dataset $\hat{D} = \hat{D}^{mat} \cup \hat{D}^{mix}$:
        $$\hat{D}^{mix} = \{((1-\mu)x + \mu x', \hat{y}) | (x, \hat{y}) \in \hat{D}^{mis}, \text{randomly sampled } (x', \hat{y}) \in \hat{D}^{mat} \text{ for each } (x, \hat{y})\}$$
Initialize a model: $f = f^{\text{init}}$
for $e = 0, \ldots, E-1$ do
    Update $f$ with SGD by minimizing the loss function $\mathcal{L}(g \circ f) = \mathcal{L}_{\text{IM}}(g \circ f; D) + \lambda \mathcal{L}_{\text{CE}}(g \circ f; \hat{D})$ defined in
      Eq. (9) while fixing $g$
return $f$
```