A FedPop Details

For completeness, we present the pseudo code of FedPop with SHA as population constructor in Algorithm 1. We set the number of active clients at each communication round to 10, random seeds to $\{1, 2, 3, 4, 5\}$. We execute 3 times elimination and set the number of initial HP-configurations to 27 for SHA following (Khodak et al., 2021). The power-law decay used in the computation of the weighted sum for the validation scores list $\{s_i\}$ (Section 3.4) is described as following:

$$s = \frac{\sum_{r=1}^{R} 0.9^r \cdot s_i^r}{\sum_{r=1}^{R} 0.9^r}, \tag{1}$$

where r is index of score in the score list, R is the length of the validation score list.

Algorithm 1: FedPop with SHA as population constructor.

```
Input: Number of active clients per round $K$, number of HP-configurations $N_c$, maximum
  communication budget for each HP-configuration $R_c$, perturbation interval for FedPop-G $T_g$, model
  weight $\boldsymbol{w}$, $N_c$ server HP-vectors $\boldsymbol{\alpha} = \{\boldsymbol{\alpha}_1, ..., \boldsymbol{\alpha}_{N_c}\}$, $N_c$ client HP-vectors $\boldsymbol{\beta} = \{\boldsymbol{\beta}_1^0, ..., \boldsymbol{\beta}_{N_c}^0\}$,
  elimination rate $\eta_{sha} \in \mathcal{N}$, elimination rounds $\{R_0 = 0, R_1, ..., R_E\}$
Copy the model weights $\boldsymbol{w}_i \leftarrow \boldsymbol{w}$ for all $N_c$ tuning processes.
for elim. step $t \leftarrow 1$ to $E$ do
    for comm. round $r \leftarrow R_{t-1}$ to $R_t$ do
        for $i \leftarrow 1$ to $N_c$ do
            // in parallel
            if $len(\boldsymbol{\beta}_i) == 1$ then
                Randomly sample $\{\boldsymbol{\beta}_i^k\}_{k=1}^K$ inside $\Delta$-ball of $\boldsymbol{\beta}_i^0$.
            for Client $k \leftarrow 1$ to $K$ do
                // in parallel
                $\boldsymbol{w}_i^k \leftarrow \texttt{Loc}(\boldsymbol{\beta}_i^k, \boldsymbol{w}_i, T^k)$
                $s_i^k \leftarrow \texttt{Val}(\boldsymbol{w}_i^k, V^k)$
            $\boldsymbol{\beta}_i \leftarrow \texttt{FedPop-L}\ (\boldsymbol{\beta}_i, \{s_i^k\}_{k=1}^K, K)$
            $\boldsymbol{w}_i \leftarrow \texttt{Agg}(\boldsymbol{\alpha}_i, \boldsymbol{w}_i, \{\boldsymbol{w}_i^k\}_{k=1}^K)$
            $s_i \leftarrow \frac{1}{K} \sum_{k=1}^K s_i^k$
        if $r\%T_g = 0$ then
            $\{\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i, \boldsymbol{w}_i\}_{i=1}^{N_c} \leftarrow \texttt{FedPop-G}\ (\{\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i, \boldsymbol{w}_i, s_i\}_{i=1}^{N_c}, N_c)$
    $\{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{w}\} \leftarrow \{\{\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i, \boldsymbol{w}_i\} : s_i \leq \frac{1}{\eta_{sha}}$-quantile$(\{s_i\}_{i=1}^{N_c})\}$
    $N_c \leftarrow \frac{N_c}{\eta_{sha}}$
return $\boldsymbol{w}$
```

A.1 The number of tried α and β

In this section, we provide the computation of the numbers of tried $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ shown in Table 1 of the main paper. Specifically, we set $N_c = 5$ for RS and $N_c = 27$ for SHA, where each tuning process is assigned with one HP-configuration, i.e., one $\boldsymbol{\alpha}$ and one $\boldsymbol{\beta}$. For FedEx wrapped with RS, we follow the settings provided in the original paper and assign each tuning process one HP-configuration and 27 additional $\boldsymbol{\beta}$, which leads to in total $(27 \times 5 =)135$ tried $\boldsymbol{\beta}$. For FedPop wrapped with RS, we provide the computation of the numbers in the following:

$$\begin{aligned} \#\text{ of tried } \boldsymbol{\alpha} &= N_c + \frac{1}{\rho} N_c \cdot \frac{R_c}{T_g}, \\ \#\text{ of tried } \boldsymbol{\beta} &= N_c \cdot K + T_g \frac{K}{\rho}. \end{aligned} \tag{2}$$

Following the experimental settings described in the main paper, we observe that FedPop experiments more HP-vectors compared with other methods.

A.2 Annealing process of ϵ and p_{re}

In this section, we describe the cosine annealing process for the values of perturbation intensity ϵ and resampling probability p_{re} described in the Section 3.4 of the main paper. For ϵ, we apply

$$\epsilon = \begin{cases} \frac{\epsilon_0}{2} \cdot (1 + cos(\pi \frac{r}{r_0})), & r < r_0 \\ 0, & r \geq r_0 \end{cases} \tag{3}$$

where ϵ_0 is set to 0.1 for all experiments. For ϵ used in `FedPop-L`, we set $r_0 = 0.2T_g$. Specifically, we stop the local search of β after the first $0.2T_g$ communication rounds of a newly initialized (perturbed) HP-configuration to save local computation costs at each client. We observe that this early-stopping of `FedPop-L` leads to comparable results as executing `FedPop-L` for all rounds. Therefore, we apply this strategy to save local computational costs without performance decrease. For ϵ used in `FedPop-G`, we set $r_0 = R_c$.

For p_{re}, we apply

$$p_{re} = \frac{p_{re}^0}{2} \cdot (1 + cos(\pi \frac{r}{R_c})) \tag{4}$$

where p_{re}^0 is set to 0.1 for all experiments.

A.3 Local search space for β_i^k

In this section, we describe the process of the selection criterion of local search space for $\boldsymbol{\beta}_i^k$ in the i-th HP-configuration. Following previous work (Khodak et al., 2021), we sample $\boldsymbol{\beta}_i^k$ inside a Δ-ball centered by $\boldsymbol{\beta}_i^0$. Specifically, for hyperparameters sampled from discrete distribution, e.g., $epoch\_num$, we define the search space as its neighboring discrete values, i.e., $\{x_{j-1}, x_j, x_{j+1}\}$, where j is the index of the current value. For hyperparameters sampled from continuous distribution, e.g., $learning\_rate$, we define the search space as $[x_j - 0.2(b-a), x_j + 0.2(b-a)]$, where a and b are the upper- and lower-bound of the original distribution.

B Experimental Details

B.1 Visualization of Benchmark Datasets

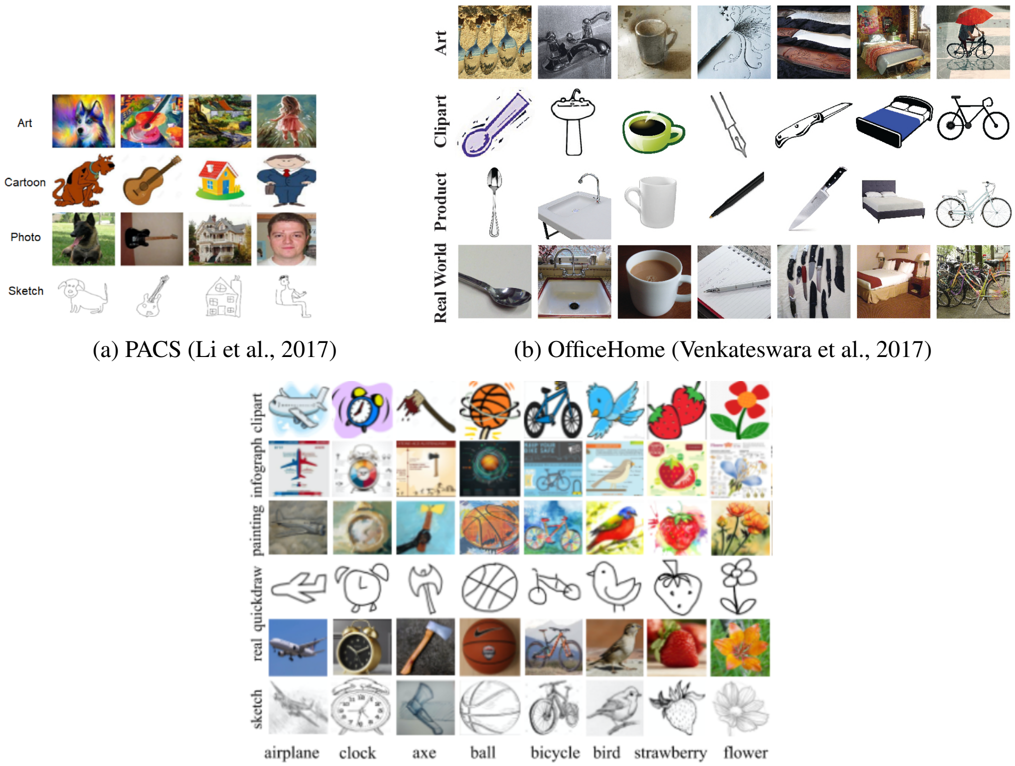
In this section, we show example images in different domains from the adopted benchmark datasets, i.e., PACS (Figure 1a), OfficeHome (Figure 1b), and DomainNet (Figure 1c). We can see that there exists strong appearance variation and distribution shifts across different domains, e.g., in PACS and DomainNet there exists both photo-like realistic pictures (*Photo*) and highly abstract human sketches (*Sketch*). Therefore, by assigning data from one of the domains to each client, we are able to simulate the experimental setting with features distribution shift in FL.

B.2 ImangeNet-1k Experimental Setup

In this section, we provide more details for our analysis on ImageNet-1K (Deng et al., 2009) dataset. We first split the original training set into training and validation set with a ratio of 9:1 for our experiment and use the original validation set as the testing set since the original test set is unlabelled. Afterwards, we split the training and validation set using the Dirichlet distribution with coefficient of 1.0. Here, we split the data into 100 subsets and assigning each subset to one client, leading to 100 clients joining the FL. We set the active clients per communication round as 10 and use the same hyperparameter search space as other datasets. For the centralized training, we adopt the hyperparameters used in the PyTorch repository https://github.com/pytorch/examples/tree/main/imagenet.

B.3 Hyperparameter Search Space

For all optimization, we use stochastic gradient descent (SGD) optimizer. We sample all hyperparameters from Uniform distribution (U), where $U\{...\}$ indicates discrete distribution and $U[a, b]$



(a) PACS (Li et al., 2017)

(b) OfficeHome (Venkateswara et al., 2017)

(c) DomainNet (Peng et al., 2019)

Figure 1: Example images from the selected cross-silo FL benchmark datasets with non-IID features. *Best viewed in color.*

indicates continuous distribution. The HP-distributions used for server HP-vector ($\boldsymbol{\alpha}$) is listed in the following:

$$\begin{aligned} log_{10}\text{lr}: &\quad U[-1,1] \\ \text{momentum}: &\quad U[0,0.9] \\ log_{10}(1-\gamma): &\quad U[-4,-2] \end{aligned}$$

where γ is the multiplicative factor of lr decay. The HP-distributions used for client local HP-vector ($\boldsymbol{\beta}$) is listed in the following:

$$\begin{aligned} log_{10}\text{lr}: &\quad U[-4,0] \\ \text{momentum}: &\quad U[0,1.0] \\ log_{10}(\lambda): &\quad U[-5,-1] \\ \text{epoch}: &\quad U\{1,..,5\} \\ log_{2}(\text{batch}): &\quad U\{3,..,7\} \\ \text{dropout}: &\quad U[0,0.5] \end{aligned}$$

where λ is the weight decay for SGD optimizer. All experiments are executed in the GPU GeForce GTX TITAN X with 12GB memory.

B.4 Model Architecture

In this section, we provide details about the model architecture used for different benchmark datasets.

Table 1: Model architecture for shakespeare.

Layer	Details
1	Embedding(95, 8)
2	LSTM(8, 256)
3	FC(256, 10)

Table 2: Model architecture for CIFAR-10.

Layer	Details

1	Conv2D(3, 32, 3, 1, 1)
	ReLU(), MaxPool2D(2, 2)

2	Conv2D(32, 64, 3, 1, 1)
	ReLU(), MaxPool2D(2, 2)

3	Conv2D(64, 64, 3, 1, 1)
	ReLU(), MaxPool2D(2, 2)

4	Dropout(p)

5	FC(1024, 64)
	ReLU()

6	FC(64, 10)

Table 3: Model architecture for FEMNIST.

Layer	Details

1	Conv2D(3, 32, 3, 1, 1)
	ReLU(), MaxPool2D(2, 2)

2	Conv2D(32, 64, 3, 1, 1)
	ReLU(), MaxPool2D(2, 2)

3	Conv2D(64, 64, 3, 1, 1)
	ReLU(), MaxPool2D(2, 2)

4	Dropout(p)

5	FC(9216, 1024)
	ReLU()

6	Dropout(p)
6	FC(1024, 62)

Following (Khodak et al., 2021), we use a 6-layer CNN with its details listed in Table 2, 3, and 1, for CIFAR-10, FEMNIST, and shakespeare dataset, respectively. For the convolutional layer (Conv2D), we list parameters with the sequence of input and output dimensions, kernel size, stride, and padding. For the max-pooling layer (MaxPool2D), we list kernel and stride. For the dropout layer (Dropout), we list dropout probability (hyperparameter in hyp-vector β). For the fully-connected layer (FC), we list input and output dimensions. For the Batch Normalization layer (BN), we list the channel dimension. For the embedding layer (Embedding), we list the number of embedding and embedding dimension. For the LSTM layer (LSTM), we list the input dimension and hidden dimension.

For the classification models on OfficeHome, PACS and DomainNet datasets, we use the widely adopted the backbone ResNet18 (He et al., 2016) and change the output dimension of the last fully-connected layer (FC) to match the number of categories in the dataset.

C Additional Results

C.1 Perturbation Intensity in FedPop

In this section, we analyze the impact of initial perturbation intensity used in FedPop, i.e., ϵ_0. Hereby, we select ϵ_0 from $\{10^{-2}, 10^{-1.5}, 10^{-1}, 10^{-0.5}, 10^{0}\}$. As shown in Figure 2, we observe that using smaller values of ϵ_0 leads to stable performance and smaller accuracy variations, where FedPop always outperforms the baseline SHA.

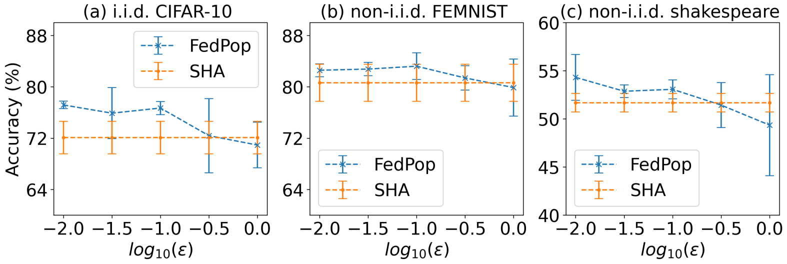


Figure 2: Effects analysis of initial perturbation intensity ϵ_0 in FedPop.

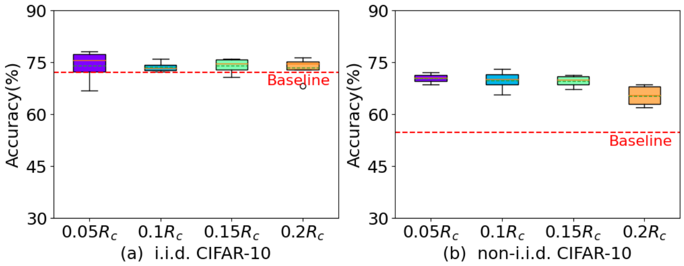
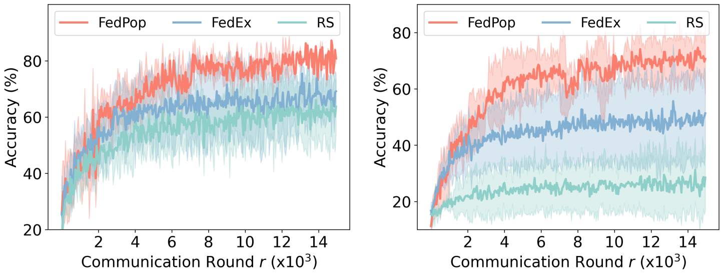


Figure 3: Effects analysis for T_g (evolutionary update frequency for FedPop−G) on CIFAR-10.



(a) *IID* CIFAR-10

(b) *NIID* CIFAR-10

Figure 4: Convergence Analysis on CIFAR-10.

C.2 Perturbation Interval T_g for FedPop−G

In this section, we provide the results of FedPop with different choices of T_g for the perturbation interval of FedPop−G. We conduct the experiments on FedPop and SHA on i.i.d. CIFAR-10 and non-i.i.d. CIFAR-10. We select T_g from $\{5\%, 10\%, 15\%, 20\%\}$ of R_c (total communication budget of each HP-configuration). From the box plot in Figure 3, we observe that applying only limited numbers of FedPop−G already leads to promising results. Most importantly, FedPop, executing FedPop−G with different frequency, always outperforms the baseline method, indicating its promising performance.

C.3 Convergence Analysis

In Figure 4, we display the convergence analysis of FedPop compared with the baseline RS and FedEx on both *IID* and *NIID* CIFAR-10 benchmarks. Hereby, we assume a federated system allowing a larger tuning budget, where we set $(R_t, R_c) = (15000, 1500)$ and report the average local testing results of the active clients after communication round r. We observe that FedPop already outperforms the other methods after 10% of the total budget, indicating its promising convergence rate. Besides, we also observe a reduced performance variation in FedPop, which further substantiates the benefits of evolutionary updates in stabilizing the overall tuning procedure.

References

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Mikhail Khodak, Renbo Tu, Tian Li, Liam Li, Maria-Florina F Balcan, Virginia Smith, and Ameet Talwalkar. Federated hyperparameter tuning: Challenges, baselines, and connections to weight-sharing. *Advances in Neural Information Processing Systems*, 34:19184–19197, 2021.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, pp. 5542–5550, 2017.

Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1406–1415, 2019.

Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5018–5027, 2017.