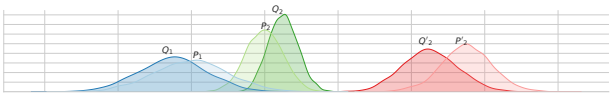

Figure 7: An example of different distribution P_1, P_2, P_2', Q_1, Q_2, Q_2'.

Proof of Theorem 1

Proof. First, we prove that If $P_1 = P_2$, $p_1 = p_2$. We can describe each personalized model as an approximation Q, the objective of the first client is

$$argmin D_{KL}\left(P_1(\mathcal{X}) \| Q_{M_{p_1}}(\mathcal{X})\right) \tag{10}$$

And the objective of the second client is

$$argmin D_{KL}\left(P_2(\mathcal{X}) \| Q_{M_{p_2}}(\mathcal{X})\right) \tag{11}$$

Without the generalization, we assume local training follows the correct process and reach the optimization point after training. Because $P_1 = P_2$, we have

$$D_{KL}\left(Q_{M_{p_1}}(\mathcal{X}) \| Q_{M_{p_2}}(\mathcal{X})\right) = 0 \tag{12}$$

For simplicity, we denote $Q_{M_{p_1}}$ as Q_1 and $Q_{M_{p_2}}$ as Q_2.

$$-D\left(Q_1 \| Q_2\right) = \sum_{x \in \mathcal{X}} Q_1(x) \ln \frac{Q_2(x)}{Q_1(x)} \tag{13}$$

$$\leq \sum_{x \in \mathcal{X}} Q_1(x) \left(\frac{Q_2(x)}{Q_1(x)} - 1\right) \tag{14}$$

$$= \left(\sum Q_2(x) - \sum Q_1(x)\right) \tag{15}$$

$$= (1 - 1) = 0 \tag{16}$$

Hence, inequality must hold. So, $Q_1 = Q_2$, which means $M_{p_1}^T \mathcal{A} = M_{p_2}^T \mathcal{A}$. Since they are sampled from p_1 and p_2, we have $\mathbb{E}[p_1] = \mathbb{E}[p_2]$.

Second, we prove that when $D_{KL}\left(P_1 \| P_2\right)$ gets larger, $\langle \mathbb{E}[p_1], \mathbb{E}[p_2] \rangle$ gets smaller.

Figure 7 shows an example of the possible distribution. In most cases, the difference between the estimation of a trained neural network and the real data distribution is smaller than the difference between different data distributions. We assume P_1 is fixed and $D_{KL}\left(P_1 \| P_2'\right) > D_{KL}\left(P_1 \| P_2\right)$. We notate $D_{KL}\left(P_1(\mathcal{X}) \| Q_{M_{p_1}}(\mathcal{X})\right)$ as D_1 and $D_{KL}\left(P_2(\mathcal{X}) \| Q_{M_{p_2}}(\mathcal{X})\right)$ as D_2. Normally, changing the non-i.i.d parameter can cause a certain distribution drift of each local dataset and on the other hand, the training neural network can lead to a relatively small error. As a result, we assume $D_{KL}\left(P_2' \| P_2\right) >> D_2, D_2', D_1$.

$$D_{KL}\left(P_1 \| P_2'\right) - D_{KL}\left(P_1 \| P_2\right) \tag{17}$$

$$= \sum_{x \in \mathcal{X}} P_1(x) \ln \frac{P_2(x)}{P_2'(x)} \tag{18}$$

$$= \sum_{x \in \mathcal{X}} P_1(x) \ln \frac{Q_2(x) + q_2(x)}{Q_2'(x) + q_2'(x)} \tag{19}$$

where we use q_2' and q_2 to describe the small distribution divergence after the local network in optimized. Since the scale of $\frac{Q_2(x)+q_2(x)}{Q_2'(x)+q_2'(x)}$ is much larger than the scale of $\frac{q_2(x)}{q_2'(x)}$, we can have

$$D_{KL}\left(P_1 \| P_2'\right) - D_{KL}\left(P_1 \| P_2\right) \tag{20}$$

$$\approx \sum_{x \in \mathcal{X}} P_1(x) \ln \frac{Q_2(x)}{Q_2'(x)} \tag{21}$$

$$\approx \sum_{x \in \mathcal{X}} Q_1(x) \ln \frac{Q_2(x)}{Q_2'(x)} \tag{22}$$

$$= D_{KL}\left(Q_1 \| Q_2'\right) \tag{23}$$

$$- D_{KL}\left(Q_1 \| Q_2\right) > 0 \tag{24}$$

As a result, we prove that $D_{KL}\left(Q_1 \| Q_2'\right) > D_{KL}\left(Q_1 \| Q_2\right)$. We can then fix P_2' and assume a P_1' which leads to a bigger divergence. For the same reason, we will have the conclusion that if $D_{KL}\left(P_1' \| P_2'\right) > D_{KL}\left(P_1 \| P_2\right)$, then $D_{KL}\left(Q_1' \| Q_2'\right) > D_{KL}\left(Q_1 \| Q_2\right)$. As a result, $\langle \mathbb{E}[p_1'], \mathbb{E}[p_2'] \rangle < \langle \mathbb{E}[p_1], \mathbb{E}[p_2] \rangle$. □

Derivation of Equation (5)

$$\begin{aligned}
\nabla J_{\alpha_i}(\alpha_i) &= \sum_{d \in [D]} \nabla_{\alpha_i} p^d r(\mathcal{A}(e = o^d)) \\
&= \sum_{d \in [D]} p^d \nabla_{\alpha_i} \log(p^d) r(\mathcal{A}(e = o^d)) \\
&= \mathbb{E}_{v_i \sim \alpha_i}[r(w_i) \nabla_{\alpha_i} \log(p_i(v_i))] \\
&\approx r_i \nabla_{\alpha_i} \log(p_i(v_i))
\end{aligned} \tag{25}$$

Derivation of Equation (6)

As shown in Equation (6), we directly give out the analytical solution of $\nabla_{\alpha_i} \log(p_i(g_i))$. So, first we omit i here, which means the client index. And for each dimension, the α_i^ds are independent architecture parameters of other dimensions of, so we can calculate the gradients of architecture parameters of each dimension separately. As a result, in the following derivation, $\nabla_\alpha \log(p(g))$ means $\nabla_{\alpha_i^d} \log(p_i^d(g_i^d))$ with any d. For each g, we assume n-th position is 1 in this one-hot vector.

$$\begin{aligned}
\nabla_\alpha \log(p(g)) &= \nabla_\alpha \log(p(g_n = 1)) \\
&= \nabla_\alpha \log(p_n) \\
&= \nabla_\alpha (\alpha_n - \log(e^{\alpha_1} + \cdots + e^{\alpha_{k_d}})) \\
&= \nabla_\alpha \alpha_n - p \\
&= \delta - p
\end{aligned} \tag{26}$$

Convergence Analysis

We prove convergence in two steps. First, we show the upper bound of the convergence rate of updating the supernet. And then we show that the architecture process of updating $\boldsymbol{\alpha}$ will converge. And these two steps can provide a convergence guarantee for PerFedRLNAS.

With the first two steps completed, we can also use them to prove the convergence of federated neural architecture search

Algorithm 2: Federated training of supernet

1: **for** each round $r \leftarrow 1$ to R **do**
2: Sample N clients: $\mathcal{N} \subseteq \{1 \cdots K\}$.
3: Local model $w_i \leftarrow M_i^T \mathcal{A}$ and send.
4: **for** on client $i \in \mathcal{N}$ parallel **do**
5: **for** $t = 1 \cdots T$ **do**
6: Compute minibatch gradients $g_i(w_i)$.
7: $w_i \leftarrow w_i - \eta_l g_i(w_i)$.
8: **end for**
9: Communicate $\Delta w_i \leftarrow w_i - M_i^T A$.
10: **end for**
11: $\Delta \mathcal{A} = \sum_{i \in \mathcal{N}} (\frac{M_i}{\sum_{i \in \mathcal{N}} M_i})^T \Delta w_i$.
12: **end for**

algorithms in previous work, which have never been proved before.

First, we make some assumptions which are realistic in actual deep learning processes.

Assumption 2. (Lower bound condition) Functions f_i are bounded below, $\min_{w_i} f_i(w_i) > -\infty$

Assumption 3. (Smooth condition) For every $i \in [K]$, f_i is continuously differentiable and L-smooth on variable w and u, and the gradient is bounded by a non-negative constant H

$$\|\nabla f_i(w)\| \leq H, \tag{27}$$

$$\|\nabla f_i(w) - \nabla f_i(u)\| \leq L\|w - u\|, \ \forall w, u \in \mathbb{R}^d \tag{28}$$

Assumption 4. (Learning rate condition) On each local step of clients, each step of selected optimizer can be approximated with a specific gradient $g_i(w_i)$ with a given learning rate η_l.

Assumption 5. (Gradient bound condition) For every $i \in [K]$, $g_i(w)$ is bounded

$$\mathbb{E}[\|g_i(w) - \nabla f_i(w)\|^2] \leq \sigma^2, \forall w \in \mathbb{R}^d \tag{29}$$

Convergence of supernet

For each w_i, we can view it as $M_i^T \mathcal{A}$, where vector M_i is a binary mask over parameters. We rewrite the process of updating the supernet in Algorithm 2. The only difference between supernet update and FedAvg, not considering the process of how we generate masks, is shown in line 3 and line 11.

Theorem 6. *For any L-smooth functions $\{f_i\}$ which satisfy Assumption 2, Assumption 3, Assumption 4, Assumption 5, according to Algorithm 2 with particular architecture parameters, value $\|\nabla f(\mathcal{A})\|^2$ has expected error smaller than ϵ in for some value of η_l, with the following bound on rounds R,*

$$R = \mathcal{O}\left(\frac{L(H^2 + \sigma^2)F}{T\epsilon^2} + \frac{L\sqrt{H^2 + \sigma^2}F}{\sqrt{T}\epsilon^{\frac{3}{2}}} + \frac{F}{\eta_l T \epsilon}\right) \tag{30}$$

where $F = f(\mathcal{A}_0) - f(\mathcal{A}^*)$.

In Algorithm 2, as each parameter of will be multiplied with a factor between 0 (not been selected) and 1 (only selected once), we can give an upper bound of the convergence rate. In Theorem 6, in FedAvg, M_i will be exactly the identity vector and as a result each parameter will be multiplied with $\frac{1}{N}$. This is the same as the time complexity for FedAvg proved in SCAFFOLD's (Karimireddy et al. 2020).

Convergence of architecture search

Since we have proved no matter what search policy is, the supernet will always converge. Now, we show that our search policy will also converge. Without loss of generality, we assume $\mathcal{A}$ is well trained. We have

Theorem 7. *On each client, if we have a search space U with D search dimensions and choices over each dimension can be represented as an one-hot vector, there must $\exists M_i^* \in U$ for each client such that $M_i^{*T} \mathcal{A}$ minimizes f over local dataset of client i.*

So the problem becomes how to find an α_i for each client such that we can generate M_i^* from α_i with a given sample function. During the architecture search process of Algorithm 1 (line 16), we have that the policy gradient is proportional to

$$\nabla J(\alpha) \propto \sum_{r \in [R]} \mu(\mathcal{A}) q_\pi(\mathcal{A}^r, a^r) \nabla \pi(a^r | \mathcal{A}^r, \alpha) \tag{31}$$

where $\mu(\mathcal{A})$ is the distribution of the supernet. According to the policy gradient theorem, the convergence rate of architecture search is proportional to $\mathcal{O}(\frac{1}{\epsilon})$.

Generalization to new clients in PerFedRLNAS only needs the architecture search step since the whole supernet $\mathcal{A}$ is already well-trained. When a new client joins the system, the client can jump the local training and uploading phase in Algorithm 1 and still obtain its customized model.

Convergence of general FedNAS

Now, we can leverage Theorem 6 and Theorem 7 to prove the convergence rate of previous federated NAS work. We summarize the bound on rounds R to make the gradient of objective function Equation (1) has expected error smaller than ϵ in Table 4.

Technicalities in Proof of Convergence

First, we refer some lemmas proved in the SCAFFOLD (Karimireddy et al. 2020).

Lemma 8. *(Separating mean and variance) Let $\{x1, \ldots, x_\gamma\}$ be γ random variable in $\mathbb{R}^d$ which are not necessarily independent. First we suppose their mean is $\mathbb{E}[x_i] = \mu_i$ and variance is bounded as $E[\|x_i - \mu_i\|^2] \leq \sigma^2$. The following holds*

$$\mathbb{E}[\|\sum_{i \in \gamma} x_i\|^2] \leq \|\sum_{i \in \gamma} \mu_i\|^2 + \gamma^2 \sigma^2 \tag{32}$$

Now suppose their conditional means is $\mathbb{E}[x_i | x_{i-1}, \ldots, x_1] = \mu_i$ and variance is bounded as before, and then we can show the tighter bound

$$\mathbb{E}[\|\sum_{i \in \gamma} x_i\|^2] \leq 2\|\sum_{i \in \gamma} \mu_i\|^2 + 2\gamma\sigma^2 \tag{33}$$

Table 4: Summarization of the bound on rounds R for different federated NAS methods to achieve expected error smaller than ϵ.

Method	Convergence rate
FedNAS	$\mathcal{O}\left(\frac{L(H^2+\sigma^2)F}{NT\epsilon^2}+\frac{L\sqrt{H^2+\sigma^2}F}{\sqrt{T}\epsilon^{\frac{3}{2}}}+\frac{F}{\eta_{\min}T\epsilon}\right)$
Direct-FedNAS	$\mathcal{O}\left(\frac{L(H^2+\sigma^2)F}{NT\epsilon^2}+\frac{L\sqrt{H^2+\sigma^2}F}{\sqrt{T}\epsilon^{\frac{3}{2}}}+\frac{F}{\eta_l T\epsilon}\right)$
SPIDER (f_i strong convex)	$\mathcal{O}\left(\frac{L(H^2+\sigma^2)F}{KT\epsilon^2}+\frac{L\sqrt{H^2+\sigma^2}F}{\sqrt{T}\epsilon^{\frac{3}{2}}}+\frac{F}{\eta_l T\epsilon}\right)$
FedorAS	$\mathcal{O}\left(\frac{B_\phi L(H^2+\sigma^2)F_1}{\phi_{\min}KT_1\epsilon_1^2}+\frac{L(H^2+\sigma^2)F_2}{NT_2\epsilon_2^2}+\frac{L\sqrt{H^2+\sigma^2}F_1}{\sqrt{T_1}\epsilon_1^{\frac{3}{2}}}+\frac{L\sqrt{H^2+\sigma^2}F_2}{\sqrt{T_2}\epsilon_2^{\frac{3}{2}}}+\frac{F_1}{\eta_{l_1}T_1\epsilon_1}+\frac{F_2}{\eta_{l_2}T_2\epsilon_2}+KC^2\right)$
FedRLNAS	$\mathcal{O}\Big(L(H^2+\sigma^2)(\frac{F_1}{T_1\epsilon_1^2}+\frac{F_2}{T_2\epsilon_2^2}+\frac{F_3}{KT_3\epsilon_3^2})+ L\sqrt{H^2+\sigma^2}(\frac{F_1}{\sqrt{T_1}\epsilon_1^{\frac{3}{2}}}+\frac{F_1}{\sqrt{T_2}\epsilon_2^{\frac{3}{2}}}+\frac{F_3}{\sqrt{T_3}\epsilon_3^{\frac{3}{2}}})+\frac{F_1}{\eta_{l_1}T_1\epsilon_1}+\frac{F_2}{\eta_{l_2}T_2\epsilon_2}+\frac{F_3}{\eta_{l_3}T_3\epsilon_3}\Big)$

Lemma 9. *(sub-linear convergence rate). For every non-negative sequence $\{d_{r-1}\}_{r\geq 1}$ and any parameters $\eta_{\max}\geq 0$,$c\geq 0$, $R\geq 0$, there exists a constant step-size $\eta\leq\eta_{\max}$ such that*

$$\Psi_R=\frac{1}{R+1}\sum_{r\in[R+1]}\left(\frac{d_{r-1}}{\eta}-\frac{d_r}{\eta}+c_1\eta+c_2\eta^2\right) \quad (34)$$

$$\leq\frac{d_0}{\eta_{\max}(R+1)}+\frac{2\sqrt{c_1d_0}}{\sqrt{R+1}}+2(\frac{d_0}{R+1})^{\frac{3}{2}}c_2^{\frac{1}{3}} \quad (35)$$

Proof of Convergence over SuperNet (Theorem 6)

In this part, we give the proof of Theorem 6

Lemma 10. *(Variance of Server Update). For updates in Algorithm 2, and Assumption 3, Assumption 4, Assumption 5 hold true for any $\bar{\eta}=\eta_l K$:*

$$E\|\mathcal{A}^r-\mathcal{A}^{r-1}\|^2\leq\frac{\bar{\eta}^2(\sigma^2+H^2)}{T} \quad (36)$$

Proof. We can write the update formulation as

$$\mathcal{A}^r=\mathcal{A}^{r-1}+\sum\sum_{i\in\mathcal{N}}(\frac{M_i}{\sum_{i\in\mathcal{N}}M_i})^T(w_{i,T}^r-M_i^T\mathcal{A}^{r-1}) \quad (37)$$

And

$$\Delta\mathcal{A}=-\frac{\bar{\eta}}{T}\sum_{t,i\in\mathcal{N}}(\frac{M_i}{\sum_{j\in\mathcal{N}}M_j})^T g_i(w_i,t-1) \quad (38)$$

And

$$\mathbb{E}[\Delta\mathcal{A}]=-\frac{\bar{\eta}}{T}\mathbb{E}[\frac{M_i}{\sum_{j\in\mathcal{N}}M_j}]^T\mathbb{E}[\sum_{i,t}\nabla f_i(w_{i,t-1})] \quad (39)$$

Now, we can calculate the norm.

$$\mathbb{E}\|\Delta\mathcal{A}\|^2=\bar{\eta}^2\mathbb{E}\|\frac{1}{T}\sum_{t,i\in\mathcal{N}}(\frac{M_i}{\sum_{j\in\mathcal{N}}M_j})^T g_i(w_{i,t-1})\|^2 \quad (40)$$

The key problem now is how we deal with the item $(\frac{M_i}{\sum_{j\in N}M_j})^T$. In federated neural architecture search, it is possible that M_i are not fixed during each communication round. On the other hand, we can give an upper bound over the whole parameter matrix of the supernet. Since M_i is a mask over each parameter of the supernet $\mathcal{A}$, each element in M_i is either 1 or 0. We then observe the denominator of $\mathbb{E}[\frac{M_j}{\sum_{i\in N}M_i}]$, for each parameter, if all personalized client models choose that parameter to compose their model, the corresponding element of the dominator is N. Otherwise, it will only be chosen once and the dominator is exactly 1 for the client i. As a result, the range of each parameter with value x is from 0 to 1.

$$\mathbb{E}\|\Delta\mathcal{A}\|^2\leq\bar{\eta}^2\mathbb{E}\|\frac{1}{T}g_i(w_{i,t-1})\|^2 \quad (41)$$

According to the Lemma 8, we can bound the each local step with bounded variance.

$$\mathbb{E}\|\Delta\mathcal{A}\|^2\leq\bar{\eta}^2\mathbb{E}[\|\frac{1}{T}\sum_{t,i}\nabla f_i(w_{i,t-1})\|]^2+\frac{\bar{\eta}^2\sigma^2}{T} \quad (42)$$

$$\leq\frac{\bar{\eta}^2(\sigma^2+H^2)}{T} \quad (43)$$

$$(44)$$

□

Lemma 11. *(bounding the drift) Suppose Assumption 3, Assumption 4, Assumption 5 hold true and we have $\bar{\eta}=T\eta_l$, we can bound the drift as*

$$\phi_r=\frac{1}{TN}\sum_{i,t}\mathbb{E}\|w_{i,t}^r-M_i^T\mathcal{A}^{r-1}\|^2\leq\frac{\bar{\eta}^2}{T}(H^2+\sigma^2) \quad (45)$$

Proof. Our target is to calculate the bounding of $\phi_r=\frac{1}{TN}\sum_{i,t}\mathbb{E}\|w_{i,t}^r-M_i^T\mathcal{A}^{r-1}\|^2$, observe that if $t=1$, $\phi_r=$

0, so lets start from $t \geq 2$

$$\mathbb{E}\|w_{i,t}^r - M_i^T \mathcal{A}^{r-1}\|^2 \tag{46}$$
$$= \mathbb{E}\|w_{i,t-1}^r - \eta_l(g_i(w_{i,t-1})) - M_i^T \mathcal{A}^{r-1}\|^2 \tag{47}$$
$$\leq \mathbb{E}\|w_{i,t-1}^r - \eta_l(\nabla f_i(w_{i,t-1})) - M_i^T \mathcal{A}^{r-1}\|^2 + \eta_l^2 \sigma^2 \tag{48}$$
$$\leq \mathbb{E}\|w_{i,t-1}^r - M_i^T \mathcal{A}^{r-1}\|^2 + \eta_l^2 \|\nabla f_i(w_{i,t-1})\| + \eta_l^2 \sigma^2 \tag{49}$$
$$\leq \mathbb{E}\|w_{i,t-1}^r - M_i^T \mathcal{A}^{r-1}\|^2 + \eta_l^2 H^2 + \eta_l^2 \sigma^2 \tag{50}$$

The we add all these recursion inequalities

$$\mathbb{E}\|w_{i,t}^r - M_i^T \mathcal{A}^{r-1}\|^2 \leq (\eta_l^2 H^2 + \eta_l^2 \sigma^2) T \tag{51}$$

So, we have

$$\phi_r \leq \frac{\bar{\eta}^2}{T}(H^2 + \sigma^2) \tag{52}$$

□

Lemma 12. *(one round progress) For updates in Algorithm 2, and Assumption 3, Assumption 4, Assumption 5 hold true for any $\bar{\eta} = \eta_l K$:*

$$\mathbb{E}[f(\mathcal{A}^r)] \leq \tag{53}$$
$$E[f(\mathcal{A}^{r-1})] + \frac{\bar{\eta}^2 L (H^2 + \sigma^2)(\bar{\eta} L + 1)}{2T} - \frac{\bar{\eta}}{2}\|\nabla f(\mathcal{A})\|^2 \tag{54}$$

Proof. We follow the conventional that the notation $\mathbb{E}_{r-1}[\cdot]$ means conditioned on filtration r.

Starting from the smoothness of f and taking conditional expectation gives

$$\mathbb{E}[f(\mathcal{A} + \Delta\mathcal{A})] \leq \tag{55}$$
$$f(\mathcal{A}) + \langle \nabla f(\mathcal{A}), \mathbb{E}_{r-1}[\Delta\mathcal{A}] \rangle + \frac{L}{2}\mathbb{E}_{r-1}\|\Delta\mathcal{A}^{r-1}\|^2 \tag{56}$$
$$\mathbb{E}[f(\mathcal{A} + \Delta\mathcal{A})] - f(\mathcal{A}) \tag{57}$$
$$\leq -\frac{\bar{\eta}}{T}\sum_{t,i}\langle \nabla f(\mathcal{A}), \mathbb{E}[\nabla f_i(w_{i,t-1})] \rangle + \frac{L}{2}\mathbb{E}\|\Delta\mathcal{A}\|^2 \tag{58}$$
$$\leq -\frac{\bar{\eta}}{T}\sum_{t,i}\langle \nabla f(\mathcal{A}), \mathbb{E}[\nabla f_i(w_{i,t-1})] \rangle + \underbrace{\frac{\bar{\eta}^2(\sigma^2 + H^2)L}{2T}}_{A_1} \tag{59}$$
$$\leq -\frac{\bar{\eta}}{2}\|\nabla f(\mathcal{A})\|^2 + \frac{\bar{\eta}}{2}\sum_{i,t}\mathbb{E}\|\frac{1}{TN}\sum_{i,t}\nabla f_i(w_{i,t-1}) \tag{60}$$
$$- \nabla f(\mathcal{A})\|^2 + A_1 \tag{61}$$
$$\leq -\frac{\bar{\eta}}{2}\|\nabla f(\mathcal{A})\|^2 + \frac{\bar{\eta}}{2TN}\sum_{i,t}\mathbb{E}\|\nabla f_i(w_{i,t-1}) \tag{62}$$
$$- \nabla f_i(M_i^T \mathcal{A})\|^2 + A_1 \tag{63}$$
$$\leq -\frac{\bar{\eta}}{2}\|\nabla f(\mathcal{A})\|^2 \tag{64}$$
$$+ \frac{\bar{\eta} L^2}{2TN}\sum_{i,t}\mathbb{E}\|w_{i,t}^r - M_i^T \mathcal{A}^{r-1}\|^2 + A_1 \tag{65}$$
$$\leq -\frac{\bar{\eta}}{2}\|\nabla f(\mathcal{A})\|^2 + \frac{\bar{\eta}^2(H^2 + \sigma^2)(\bar{\eta} L^2 + L)}{2T} \tag{66}$$
$$\mathbb{E}[f(\mathcal{A} + \Delta\mathcal{A})] \leq E[f(\mathcal{A})] \tag{67}$$
$$+ \frac{\bar{\eta}^2(H^2 + \sigma^2)(\bar{\eta} L^2 + L)}{2T} - \frac{\bar{\eta}}{2}\|\nabla f(\mathcal{A})\|^2 \tag{68}$$

□

Now, we can use Lemma 12 and Lemma 9 to calculate the convergence rate of training supernet with fixed hyperparameters of α_i s.

Proof. By unrolling Lemma 12, we can have

$$\frac{\bar{\eta}}{2}\sum_{i\in[R+1]}\|\nabla f(\mathcal{A}^r)\|^2+\sum_{i\in[R+1]}\mathbb{E}[f(\mathcal{A}^r)] \tag{69}$$

$$\leq\sum_{i\in[R+1]}\mathbb{E}[f(\mathcal{A}^{r-1})]+ \tag{70}$$

$$(R+1)\frac{\bar{\eta}^2(H^2+\sigma^2)(\bar{\eta}L^2+1)}{2T} \tag{71}$$

$$\mathbb{E}[\|\nabla f(\mathcal{A})\|^2]\leq \tag{72}$$

$$\frac{1}{R+1}\sum_{i\in[R+1]}\left(\overbrace{\frac{\mathbb{E}[f(\mathcal{A}^{r-1})]}{\bar{\eta}}-\frac{\mathbb{E}[f(\mathcal{A}^r)]}{\bar{\eta}}}^{d}\right. \tag{73}$$

$$\left.+\underbrace{\frac{L(H^2+\sigma^2)}{T}}_{c_1}\bar{\eta}+\underbrace{\frac{L^2(H^2+\sigma^2)}{T}}_{c_2}\bar{\eta}^2\right) \tag{74}$$

$\because$ Lemma 9 (75)

$$\therefore\mathbb{E}[\|\nabla f(\bar{\mathcal{A}}^R)\|^2]\leq \tag{76}$$

$$\mathcal{O}\left(\frac{\sqrt{L(H^2+\sigma^2)F}}{\sqrt{RT}}+\right. \tag{77}$$

$$\left.\frac{L^{2/3}(H^2+\sigma^2)^{1/3}F^{2/3}}{T^{1/3}R^{2/3}}+\frac{F}{R\bar{\eta}}\right) \tag{78}$$

where $F=f(\mathcal{A}_0)-f(\mathcal{A}^*)$. □

Proof of convergence of architecture search

Proof of Theorem 7

Proof. According to the definition of the search space (supernet), the search dimension is fixed: D and for each search dimension $d\in[D]$, the number of candidate operations is fixed k_d. As a result, U is a finite set where $|U|=\prod_{d\in[D]}2^{k_d}$. Though the norm of the set U is not a polynomial, it is still a finite set. According to Assumption 2, on each client function f_i is bounded below, which means that $\forall M\in U$, $f_i(M^T\mathcal{A})>-\infty$. Since all possible values in finite set U are greater than $-\infty$, such M_i^* must exist. □

Derivation of Equation (31)

According to our definition we have

$$J(\alpha_i)=\nu_{\pi_{\alpha_i}}(\mathcal{A}) \tag{79}$$

First,

$$\nabla\nu_{\pi_{\alpha_i}}(\mathcal{A})=\nabla\left[\sum_a\pi(a|\mathcal{A})q_{\pi_{\alpha_i}}(\mathcal{A},a)\right] \tag{80}$$

$$=\sum_a[\nabla\pi(a|\mathcal{A})q_\pi(\mathcal{A},a)+\pi(a|\mathcal{A})\nabla q_\pi(\mathcal{A},a)] \tag{81}$$

And,

$$\nabla\nu_{\pi_{\alpha_i}}(\mathcal{A}_0) \tag{82}$$

$$=\sum_a[\nabla\pi(a|\mathcal{A}_0)q_\pi(\mathcal{A}_0,a)+\pi(a|\mathcal{A}_0)\nabla q_\pi(\mathcal{A}_0,a)] \tag{83}$$

$$=\sum_a[\nabla\pi(a|\mathcal{A}_0)q_\pi(\mathcal{A}_0,a)+ \tag{84}$$

$$\left.\pi(a|\mathcal{A}_0)\nabla\sum_{\mathcal{A}_1,r}p(\mathcal{A}_1,r|\mathcal{A}_0,a)(r+\nu_\pi(\mathcal{A}_1))\right] \tag{85}$$

(r is the reward) (86)

$$=\sum_a[\nabla\pi(a|\mathcal{A}_0)q_\pi(\mathcal{A}_0,a)] \tag{87}$$

$$+\sum_a\left[\nabla\pi(a|\mathcal{A}_0)\sum_{\mathcal{A}_1}p_\pi(\mathcal{A}_1|\mathcal{A}_0,a)\right] \tag{88}$$

$$*\sum_{a'}[\nabla\pi(a'|\mathcal{A}_1)q_\pi(\mathcal{A}_1,a')] \tag{89}$$

$$+\sum_a\left[\nabla\pi(a|\mathcal{A}_0)\sum_{\mathcal{A}_1}p_\pi(\mathcal{A}_1|\mathcal{A}_0,a)\right] \tag{90}$$

$$*\sum_{a'}\left[\pi(a'|\mathcal{A}_1)\nabla\sum_{\mathcal{A}_2,r}p(\mathcal{A}_2,r'|\mathcal{A}_1,a')\right. \tag{91}$$

$$\left.(r+\nu_\pi(\mathcal{A}_1))\right] \tag{92}$$

We can have

$$\nabla J(\alpha_i)=\nabla\nu_{\pi_{\alpha_i}}(\mathcal{A}_0) \tag{93}$$

$$=\sum_{\mathcal{A}}\left(\sum_{t=0}^{\infty}\Pr(\mathcal{A}_0\to\mathcal{A},t,\pi)\right) \tag{94}$$

$$\nabla\pi(a|\mathcal{A})q_\pi(\mathcal{A},a) \tag{95}$$

(one action each step) (96)

$$=\sum_{\mathcal{A}}\eta(\mathcal{A})\nabla\pi(a|\mathcal{A})q_\pi(\mathcal{A},a) \tag{97}$$

$$=\sum_{\mathcal{A}'}\eta(\mathcal{A}')\sum_{\mathcal{A}}\frac{\eta(\mathcal{A})}{\sum_{\mathcal{A}'}\eta(\mathcal{A}')}\nabla\pi(a|\mathcal{A})q_\pi(\mathcal{A},a) \tag{98}$$

$$=\sum_{\mathcal{A}'}\eta(\mathcal{A}')\sum_{\mathcal{A}}\mu(\mathcal{A})\nabla\pi(a|\mathcal{A})q_\pi(\mathcal{A},a) \tag{99}$$

$$\propto\sum_{\mathcal{A}}\mu(\mathcal{A})q_\pi(\mathcal{A},a)\nabla\pi(a|\mathcal{A},\alpha) \tag{100}$$

Convergence Proof of Table 4

Theorem 13. *For any L-smooth functions $\{f_i\}$ which satisfy Assumption 2, Assumption 3, Assumption 4, Assumption 5, if the supernet is fixed and transmitted as a global model according to the FedAvg, value $\|\nabla f(\mathcal{A})\|^2$ has expected error smaller than ϵ in for some value of η_l, with the following*

bound on rounds R,

$$R = \mathcal{O}\left(\frac{L(H^2+\sigma^2)F}{NT\epsilon^2} + \frac{L\sqrt{H^2+\sigma^2}F}{\sqrt{T}\epsilon^{\frac{3}{2}}} + \frac{F}{\eta_l T\epsilon}\right) \tag{101}$$

where $F = f(\mathcal{A}_0) - f(\mathcal{A}^*)$.

Proof. Let us go back to the Equation (40). In FedAvg, the mask is always the identity vector as all parameters are selected and the global model is shared through federated learning. As a result, $\frac{M_i}{\sum_{j\in N} M_j}$ will always be $\frac{1}{N}\mathbf{1}$. As a result, we can give a tighter bound over the $\mathbb{E}\|\Delta\mathcal{A}\|^2$. Substitute back into the variance of server update we can have

$$\mathbb{E}\|\Delta\mathcal{A}\|^2 \leq \frac{\bar{\eta}^2(\sigma^2+H^2)}{TN} \tag{102}$$

As a result, the progress in each round it will be

$$\mathbb{E}[f(\mathcal{A}^r)] \leq E[f(\mathcal{A}^{r-1})] + \tag{103}$$

$$\frac{\bar{\eta}^2 L(H^2+\sigma^2)(\bar{\eta}L + \frac{1}{N})}{2T} - \frac{\bar{\eta}}{2}\|\nabla f(\mathcal{A})\|^2 \tag{104}$$

Hence, we can get the resulted estimation of communication rounds. □

FedNAS

Proof. In FedNAS (He, Annavaram, and Avestimehr 2020), the huge supernet DARTS is directly distributed among the clients and the server. As a result, on the server, there is only the averaging aggregation and on the client, each client will do the DARTS training. In the FedNAS, they use the first-order gradient method in the DARTS training. During the DARTS training, we can view the training of the supernet is the same as training a traditional neural network as it also strictly follows the rules of forwarding and backwarding, and we can view the architecture parameters α in DARTS as a part of the model weights. However, in DARTS the architecture parameters are updated with the formulation

$$g_\alpha = \nabla_\alpha \mathcal{L}_{\text{valid}}(w, \alpha) \tag{105}$$

Hence, the local training dataset on each client should be $\{\mathcal{D}_{\text{valid}}, \mathcal{D}_{\text{train}}\}$, and as the update of α and the update of weights w use different learning rate. The expectation of convergence rate is dominated by slower learning rate. We donate the learning rate as η_α and η_w. Since FedNAS directly used the supernet as the global model, the process of FedNAS can be summarized with Algorithm 2. The convergence rate can be summarized with the Equation (30), except two changes. First, in each communication round the whole supernet is distributed. As a result, the matrix M is $\mathbf{1}$. We then can directly derive out the convergence rate according to Theorem 13. The learning rate η_l is $\eta_{\min} = \min(\eta_\alpha, \eta_w)$. We substitute these variables into the Equation (30) and we will get the result. □

DirectFedNAS

Proof. In DirectFedNAS (Garg, Saha, and Dutta 2021), it took a similar algorithm as the FedNAS accept they change the huge supernet into the DSNAS (Hu et al. 2020) where the update of α is not backwarded from the validation loss, but directly over the training loss. As a result, we can still view the α as a part of of the model parameters of a neural network $\mathcal{A}$ and the local training dataset on each client is only $\mathcal{D}_{\text{train}}$. We view α as a part of the parameters. So, the learning rate is some value η_l. In each round, the matrix M is also $\mathbf{1}$. □

SPIDER

In SPIDER (Mushtaq et al. 2021), the update of the global model has nothing to do with the personalized rules of the local model. As as result, in SPIDER, the convergence rate of the supernet is Equation (30) where $M = \mathbf{1}$. We then need to draw a connection between the convergence of the global model with the convergence of the local model update. Actually, the construction of local personalized models in SPIDER is similar to the update rule adopted by Ditto, so we first propose a similar theorem (Li et al. 2021).

Theorem 14. *(Local Convergence of personalized model.) Assume for $i \in [K]$, f_i is strongly convex and smooth, under common assumptions, if the supernet converges to $\mathcal{A}^*$ with rate $g(t)$, there exists a constant $C < \infty$ such that local customized models converge to the optimal with rate $C(gt)$.*

Before proof of this theorem, we first must be certain of some definitions. According to the definition $\mathcal{A}_{share} = \mathcal{A} \odot \alpha_{i,local}$ minimizes the f_i on the local validation set. So w_{share} is the empirically global model for f_i. Let v_i^* denote the optimal local customized model on the client i. We introduce and additional assumption on the distance between optimal local models and the $\mathcal{A}_{share}$ that

Assumption 15.

$$\|v_i^* - \mathcal{A}^*\| \leq J \tag{106}$$

and the L2 norm of the model parameters is also bounded upper:

Assumption 16.

$$\|\mathcal{A}\| \leq J' \tag{107}$$

Proof. The key to prove the Theorem 14 is to prove $\|v_i^* - \mathcal{A}^* \odot \alpha_{i,local}\|$ also exists a boundary.

$$\|v_i^* - \mathcal{A}^* \odot \alpha_{i,local}\| \tag{108}$$

$$= \|v_i^* - v_i^* \odot \alpha_{i,local} + v_i^* \odot \alpha_{i,local} - \mathcal{A}^* \odot \alpha_{i,local}\| \tag{109}$$

$$\leq \|v_i^* - v_i^* \odot \alpha_{i,local}\| + \|v_i^* \odot \alpha_{i,local} - \mathcal{A}^* \odot \alpha_{i,local}\| \tag{110}$$

$$= \|v_i^* \odot \bar{\alpha}_{i,local}\| + \|v_i^* - \mathcal{A}^*\| \tag{111}$$

$$\leq J' + J \tag{112}$$

□

Since we have proved the boundary of L2 distance between empirically local optimal model and the empirically (over the local validation set) optimal shared global model. We now have:

Theorem 17. *Proof of Theorem 14 If there exists a constant A such that $\frac{g(t+1)}{g(t)} \geq 1 - \frac{g(t)}{A}$, there there exists $C < \infty$ such that for ant client $i \in [K]$, $\mathbb{E}[\|v_i^t - v_i^*\|^2] \leq Cg(t)$ with some learning rate η.*

Proof. We proceed with the proof by induction. First of call $C > \frac{\mathbb{E}[\|v_i^0 - v_i^*\|^2]}{g(0)}$. If $\mathbb{E}[\|v_i^t - v_i^*\|^2] \leq Cg(t)$ holds, it holds for $t+1$ because from Lemma 13 in Ditto, we can have

$$\begin{aligned}\mathbb{E}[\|v_i^t - v_i^*\|^2] \leq & \left(1 - \frac{2g(t)}{A}\right) Cg(t) \\ & + \frac{g(t)^2}{A} \frac{4}{Ap_k^2} \left(\frac{\left(G_1 + \lambda(M + \frac{G_1}{\mu})\right)^2}{(\mu+\lambda)^2} \right. \\ & + g(t) \\ & \left. + \frac{2(G_1 + \lambda(M + \frac{G_1}{\mu}))\sqrt{g(t)}}{\mu + \lambda} \right) \\ & + g(t)^2 \frac{4\lambda\sqrt{C}}{\mu+\lambda} \\ & \leq \left(1 - \frac{2g(t)}{A}\right) Cg(t) + \frac{Cg(t)^2}{A}\end{aligned} \tag{113}$$

hold for some $C < \infty$. Hence

$$\mathbb{E}[\|v_i^t - v_i^*\|^2] \leq Cg(t) \tag{114}$$

$$\leq \left(1 - \frac{2g(t)}{A}\right) Cg(t) + \frac{Cg(t)^2}{A} \tag{115}$$

$$= \left(1 - \frac{g(t)}{A}\right) Cg(t) \tag{116}$$

$$\leq Cg(t+1) \tag{117}$$

□

Now we use the Theorem 14 and the Equation (30) and in SPIDER all clients are participated in each round, so we can get the resulted convergence rate.

FedorAs

Proof. In FedorAS (Laskaridis, Fernandez-Marques, and Dudziak 2022), the complete algorithm has three phases. The supernet training phase, the search phase and the fine-tune phase. In the supernet training phase, as in each communication round, a subspace is sampled instead of using the whole search space, the M are no longer fixed. In the search phase of FedorAS, $N = [K]$. We let

$$\overline{M} = \mathbb{E}[\frac{M_i}{\sum_{i \in N} M_i}] \tag{118}$$

Another thing worths noticing is that according to the Aggregation with OPA in FedorAS, if any element in the matrix $\frac{M_i}{\sum_{i \in [K]} M_i}$ is 1, it will be set to 0 without update of gradients in the global weights. And we also have

$$\phi_{\text{comm}}^T M_i < B_{\phi_{\text{comm}}} \tag{119}$$

where ϕ_{comm} is the cost of each each operation and $B_{\phi_{\text{comm}}}$ is the total cost budget. Unfortunately, as in the FedorAS, the authors deploy a randomized sample policy, so we can only give a slack estimation of the upper bound of $\|\overline{M}\|^2$.

$$\|\overline{M}\|^2 = \|\mathbb{E}[\frac{M_i}{\sum_{i \in [K]} M_i}]\| \tag{120}$$

$$\leq \mathbb{E}[\|\frac{M_i}{\sum_{i \in [K]} M_i}\|^2] \tag{121}$$

$$< \mathbb{E}[\frac{\|M_i\|^2}{K}] \tag{122}$$

$$< \mathbb{E}[\frac{B_{\phi_{\text{comm}}}}{\phi_{\min} K}] \tag{123}$$

$$= \frac{B_{\phi_{\text{comm}}}}{\phi_{\min} K} \tag{124}$$

where $\phi_{\min}$ is the minimal cost of the operations. During the local training of each client, the process is the same as training a neural network with the neural architecture search method. As a result, we substitute the upper bound of $\|\overline{M}\|^2$ into Equation (40) and

$$\mathbb{E}\|\Delta\mathcal{A}\|^2 = \bar{\eta}^2 \mathbb{E}\| \frac{1}{T} \sum_{t,i \in N} \left(\frac{M_i}{\sum_{j \in N} M_j}\right)^T g_i(w_{i,t-1})\|^2 \tag{125}$$

$$\leq \frac{\bar{\eta}^2}{T} \|\frac{M_i}{\sum_{j \in N} M_j}\|^2 \| \sum_{t,i \in N} g_i(w_{i,t-1})\|^2 \tag{126}$$

$$< \frac{\bar{\eta}^2}{T} \frac{B_{\phi_{\text{comm}}}}{\phi_{\min} K} \| \sum_{t,i \in N} g_i(w_{i,t-1})\|^2 \tag{127}$$

We then follow the similar steps in the proof of Theorem 6 and will get the bound on rounds R.

Regarding the search phase, the time complexity of the search algorithm is KC^2. According to the original NSGA-II, K is the number of objectives, where we have K clients with local optimal objective and C is the population size which is set to 64 or 128 in FedorAS.

Regarding the fine-tune phase, the time cost is the same as the time cost of conventional federated averaging training. The fine-tune phase is a further step to minimize errors. As a result, we assume after the supernet training phase and the search phase the error can be only minimized to ϵ_1, then the rest error to optimize is $\epsilon_2 = \epsilon - \epsilon_1$. And in the supernet training phase we achieve the $\mathcal{A}^*$, in the search phase we get the M^*. In the find-tune phase, the process is to find the $(M^{*^T}\mathcal{A})^*$.

We define the F_1 as $f(\mathcal{A}_0) - f(\mathcal{A}^*)$, T_1 is the local steps to carry and η_{l_1} is some learning rate during supernet training phase. We also define F_2 as $f(M^{*^T}\mathcal{A}^*) - f(M^{*^T}\mathcal{A}^*)$, T_2 is the local steps to carry and η_{l_2} is some learning rate during fine-tune phase. □

FedRLNAS

Proof. In FedRLNAS (Yao et al. 2021), during each communication round, they also used a mask for each client and sampled a sub-network for each client. However, they still searched for a global model instead of customized models

Table 5: A toy experiments of comparing LeViT-128 model with T2T-ViT-14 in federated settings. We use 100 clients and 5 clients are selected during each communication round. The datasets are of non-i.i.d with $D_\alpha = 0.3$ on basis of CIFAR10. Hyperparameters are set for those led to their best performance in centralized settings.

Model	Averaged accuracy	Std of accuracy
LeViT-128	**72.22**	6.70
T2T-ViT-14	65.56	7.78

Table 6: A toy experiments of comparing MobileNetV3 models with ResNet50 in federated settings. We use 100 clients and 5 clients are selected during each communication round. The datasets are of non-i.i.d with $D_\alpha = 0.3$ on basis of CIFAR10. Hyperparameters are set for those led to their best performance in centralized settings.

Model	Averaged accuracy	Std of accuracy
ResNet50	71.89	6.77
MobileNetV3-Small	70.31	6.87
MobileNetV3-Large	**74.44**	6.11

for each client. Since during the process of training supernet, the sample process can also be expressed as line 7 in Algorithm 1. We can apply the Theorem 6 to the warmup phase and search phase in FedRLNAS. We assume the local training epochs, local learning rate and the expected error of these two phases are T_1 and T_2, η_{l_1} and η_{l_2}, ϵ_1 and ϵ_2. Respectively. We define the F_1 as $f(\mathcal{A}_0) - f(\mathcal{A}_1)$, where $\mathcal{A}_1$ is the end state of supernet at the warmup phase, which can not further be optimized by the policy in warmup phase. We define F_2 as $f(\mathcal{A}_1) - f(\mathcal{A}^*)$. We can first derive out the convergence rate of the warmup and search phases.

At the third phase, the fine-tune phase is just the same as FedAvg where we use the searched structure as the fixed model. As a result, we define F_3 as $f(M^{*^T}\mathcal{A}^*) - f(M^{*^T}\mathcal{A}^*)$, T_3 as the local training epochs, η_{l_3} as the local training rate and ϵ_3 as the expected error and put it into Theorem 13. □

Experiment settings

Datasets

CIFAR10 and CIFAR100 (Krizhevsky, Hinton et al. 2009) datasets are image classification tasks with 60000 samples. CIFAR10 has 10 categories and CIFAR100 has 100 categories. Among them, 50000 samples are used for training and 10000 samples are used for testing. The input resolution is scaled up to the required resolution of each supernet. The input resolution for DARTS and MobileNetV3 is 224×224. The input resolution for NASVIT includes 128, 192, 224, 256 as input resolution is also one of its search dimensions. For the input into all baselines, the resolution is 224×224. We evenly distributed these samples over K clients with fixed random seeds.

Models

Manually Designed Models In conventional personalized federated learning methods, we use the state-of-the-art manually tuned models as the backbone models for these baselines. The major reason behind choosing selected models is they perform the best performance on a wide range of tasks, specifically in the field of computer vision and they also achieve a good performance regarding the efficiency. We use these models directly from `Plato` APIs.

- LeViT (Graham et al. 2021) is a family of efficient models leveraging a hybrid architecture of convolutions and transformers. The NASVIT search space is constructed on the basis of LeViT. They are from the same ViT family which provides a relatively fair comparison. In LeViT, the convolutions are introduced to handle high resolution inputs thanks to their efficiency from local computation while the transformers are leveraged for lower resolution features to extract global information. LeViT can achieve 76.6% top-1 accuracy with 305M FLOPS on ImageNet. In a centralized setting, LeViT can be viewed as one of the best ViTs. We use the LeViT-128. We have a toy experiment to show that LeViT also outperforms T2T-ViT (Yuan et al. 2021) as shown in Table 5. T2T-ViT is a family of models leveraging transformer blocks which has a similar structure to the conventional ViTs but has a better performance.
- MobileNetV3 (Howard et al. 2019) is a novel architecture search space using the third generation designed Mobilenet blocks. MobilenetV3 achieves better accuracies over the well adopted structure ResNet (He et al. 2016) and because of its design objectives for implementing on mobile devices, it is efficient. Besides, as federated learning is usually carried over mobile clients, it is also reasonable to use such an architecture. MobileNetV3-Large and MobileNetV3-Small are searched using NAS complemented by the NetAdapt algorithm. Their architectures are further manually tuned after being searched. We have a toy experiment to show that MobileNetV3 also outperforms ResNet in federated settings (FedAvg). As shown in Table 6, we choose the MobileNetV3-Large which has the highest performance.

Search Space

- NASVIT (Gong et al. 2022) is a state-of-the-art architecture search space in one-shot architecture search and block-wise search designed for searching ViTs. It achieves higher accuracy over image classification tasks and the searched models consume fewer resources than previous work of VIT NAS. The supernet 40 search dimensions in total and $2^{20} \times 3^{10} \times 7$ possible architectures.
- MobilenetV3 search space is a search space in block-wise architecture search and we use has 23 search dimensions in total and $2^{14} \times 3^7$ possible architectures.
- DARTS (Liu, Simonyan, and Yang 2019) is a differential architecture search method based search space where they are built on basis of cells. We use 8 cells in all the methods using DARTS. For each cell, it has 14 edges, where each edge represents an operation connection the feature maps.

On each edge, there are 8 possible candidate operations. As a result, it has 14 search dimensions and 2^{42} possible architectures.

Implementation details

Experiment platform

We conduct the simulation on the open source platform PLATO, a framework to facilitate scalable federated learning research. We can directly call the models, datasets and simulate the network and system environment through the provided APIs. We develop our codes on `PyTorch`. We ran our experiments with GPUs NVIDIA A100 and CPUs AMD Milan 7431.

Hyperparameters and training settings

General settings: by default, we train the models with 100 clients, 5 clients are randomly selected during each communication round. During each local training loop, $T = 5$. We set the batch size as 32 for all the experiments. The random seed for generating a non-i.i.d. dataset and for generating other random numbers are fixed. Each client has 600 samples for training and 600 samples for validation.

For PerFedRLNAS, we set the architecture learning rate as 0.5, with Adam optimizer. The weight decay is 0 and $\beta = (0.5, 0.99)$. For architecture parameters α_i, they are all initialized as zero. In the reward function, we set $\lambda_{\text{time}} = 0.001$. Regarding local training on the client, the hyperparameter of optimizers and schedulers on three search space are shown in Table 7. The scheduler is a cosine annealing scheduler by default.

For conventional personalized federated learning methods using manually tuned models, training hyper-parameters are set by default the same as the settings in centralized training as. They are the same as in Table 7. Except for LeViT, the weight decay is 0.025 and the $\eta_{\min}$ for the scheduler is $1e^{-5}$. For methods FedRep and FedBabu, as defined in the paper, we choose the last two layers as the classifier and the rest layers as the representation. For FedTP, all settings are the same as the paper's settings.

For other federated neural architecture search methods, as the search space is DARTS, the hyperparameters to train the supernet is the same as it is listed in Table 7. For method FedNAS and SPIDER, as they only have the search phase as our method, we train them directly to the convergence. For the method FedRLNAS, we conduct 800 rounds of warmup phase and 400 rounds of search phase and then federated train the model to convergence. For the method FedorAS, we set the number of tiers to 1 as the number of tiers will not affect the highest accuracy that can be achieved. The budget $B_{\phi_{\text{comm}}}$ is set to $1/2$ of the size of the supernet. We conduct 1500 rounds of supernet training and then 1000 steps of model search and train the model of this tier to the convergence. According to the results and algorithm of FedorAS, leveraging one tier can achieve the best performance as the the functionality of assigning different tiers is to meet different budgets of the devices. As a result, with setting one tier, there will be no restrictions on the model sizes or the flops, which can represent the best performance can be achieved by the FedorAS. On the other hand, we also conducted an experiment with using four tiers during the fine-tuning phase on the setting of CIFAR10. The accuracy of the four tiers is:75.52 ± 5.92, 74.91 ± 6.63, 75.39 ± 6.11 and 74.32 ± 5.87. As a result, the one tier can achieve the best performance of FedorAS as we report.

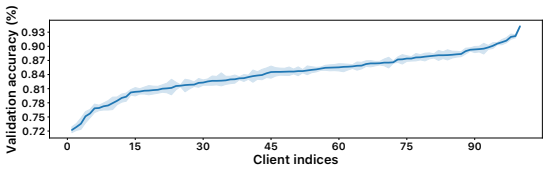

Figure 8: Repeat PerFedRLNAS over NASVIT search space five times. Clients are sorted along the average accuracy.

Additional Experiments

Stability of PerFedRLNAS

During the experiment, our random seeds are fixed. To show the stability of our algorithm, we change the random seeds except the random seed for generating non-i.i.d datasets to see the stability of PerFedRLNAS. We repeated the experiments with different random seeds. The seed for generating the datasets is fixed. As a result, the dataset on each client will not change. We repeat the experiment of PerFedRLNAS using NASVIT (which has the largest search space) in 5 different runs in total. As shown in Figure 8, we sort the clients along the local accuracy averaged over these 5 runs. The scale of variance among different runs is much less than the variance among different clients. The std over these runs is 0.498. As a result, randomness will not affect the integrity of our reported results and PerFedRLNAS is stable.

Experiments over text dataset

Apart from the image classification datasets, we also compared our methods with the baselines over the text dataset. We use the Shakespeare dataset, which is a non-i.i.d. text dataset from LEAF (Caldas et al. 2018). We have 1129 clients in total and during each round, 10% clients participate in the federated training. We use the transformer structured models where the fixed model is BERT large (Devlin et al. 2018) and the supernet is a block-wise searching supernet with BERT blocks. During each training loop, $T = 1$. The batch size is 64. The hyperparameter setting is shown in Table 7. The experiment results is shown in Table 8. Our method can also outperforms the baselines in terms of efficiency and accuracy over the non-i.i.d. text dataset.

Discussion

In this paper, we propose a one-for-all personalized federated NAS framework which can apply arbitrary NAS supernets and solve the heterogeneity problems in federated learning through setting different reward functions. Here, we provide more discussion about our basic design idea and provide several directions for future work and social impacts.

Table 7: Hyperparameter settings for local training in PerFedRLNAS on three search space.

Search space	optimizer			scheduler	
	Type	Learning rate	Other parameters	$\eta_{\min}$	Global scheduler
NASVIT	AdamW	$5e^{-4}$	$\epsilon = 1e^{-8}$, $\beta = (0.9, 0.999)$, weight decay $5e^{-4}$	$6e^{-6}$	✓
MobileNetV3	SGD	0.35	momentum 0, weight decay $5e^{-6}$	0	✓
DARTS	SGD	0.025	momentum 0.9, weight decay $3e^{-4}$	0.001	✓
BERT	SGD	0.1	momentum 0.9, weight decay $5e^{-4}$	0.001	✓

Table 8: Comparison of accuracy and efficiency over 1129 clients with participation rate 10% using BERT structure model over non-i.i.d. Shakespeare dataset. The latency is calculated in hours.

Method	$\overline{\text{Acc}}s \pm$Std	Latency
FedAvg	56.96 ± 4.90	2.35
FedRep	57.12 ± 4.58	2.40
FedBabu	57.00 ± 4.90	2.15
HeteroFL	54.32 ± 4.84	2.59
PerFedRLNAS	64.32 ± 3.76	2.07

Limitation and Future work

Various supernets and targets As we have illustrated, we can apply arbitrary supernets, supernets in different tasks can also be leveraged. We choose several representative supernets in the field of centralized NAS such as DARTS, NASVIT. Apart from NAS for vision tasks, we can also leverage the language model supernet in our framework. Apart from the supernet, we can set different reward functions to realize different objectives of the heterogeneity. For example, with $\lambda_{\text{time}}\text{RT}$ and accuracy in the reward function, we can achieve a good tradeoff between statistic heterogeneity and client running speed heterogeneity. We can also add the ϕ penalties in the reward function to achieve memory budget aware search. Other reward functions can also be designed to resolve different use cases. Hence, one limitation is that the performance is related to the supernet we choose. Better supernet under our method can help achieve better performance.

Personalized reward function According to our design of the reward function, each client will have a reward function and a virtual RL agent on the server to calculate their best model architectures. As a result, the calculation of architecture and the reward function is individual for each client. As a result, an interesting future direction is that we can set different reward functions for different clients. For example, if a client cares a lot about its efficiency, we can give more weight to efficiency related metrics in its reward function and if a client cares a lot about its memory budget, we can give a heavy penalty over its ϕ_i item in the reward function.

Complete asynchronous framework As we have discussed, the architecture search phase for each client is individually updated. As a result, we can embed this feature into another topic in the field of federated learning which is asynchronous federated learning. Clients do not need to synchronize with each other and they can upload their weights and leverage the virtual RL agent on the server to calculate their customized architectures asynchronously, which can further improve efficiency in federated learning.

Broader Impacts

We propose a new personalized federated neural architecture search algorithm to address heterogeneous problems in federated learning. For the broader impact, the feature of being capable with different neural architectures supernets and reward function can help solve various applications in the real world and produce the best performance in federated learning.

On the other hand, our solution can apply different reward functions. A malicious server can also deliberately design somewhat reward function to guide the client to train particularly customized model which can leak the privacy of the data on the client, especially when the model are highly personalized with different architectures according to their personalized dataset. It will be a future work that how we can detect such attacks where the server design particular reward functions to steal privacy through architecture search.