A APPENDIX

A DATA HETEROGENEITY AND HYPERPARAMETERS

A.1 DATA HETEROGENEITY

Figure 3 and 4 represents data distribution across 8 clients for CIFAR10 and CIFAR100 datasets, respectively. Sub-figure 3a and 4a represent the label distribution across clients, where a darker color indicates more images of that class/label. It can be seen from these sub-figures that the data distribution is heterogeneous label-wise, e.g, Client with ID 0 has class 0 and 9 in excess, whereas, Client with ID 7 has more samples of class 3 and 7 for the CIFAR10 dataset. We observe a similar trend of heterogeneity for CIFAR100 as well, where we have 100 class labels distributed across 8 clients via LDA distribution with α parameter = 0.2. Furthermore, the sub-figures 3b and 8c represent the total number of data samples preset at each client. These figures illustrate that the number of samples is also varying across silos. However, the variation in the total number of samples by silo is more prominent for the CIFAR10 dataset.

A.2 HYPER-PARAMETERS SEARCH

For empirical results of CIFAR10, CIFAR100, and CINIC10, we use a batch size of 32 for all our experiments. We use LDA distribution with a 0.2 parameter value. For SPIDER, we use the first 30 rounds as warmup rounds, for SPIDER-Searcher, we use a recovery period of 20. Furthermore, we use a learning rate in the search range of $\{0.01, 0.03\}$ for SPIDER. For SPIDER, we used λ search from the set of $\{0.01, 0.1\}$. For the other personalized schemes such as Ditto, perFedAvg, and local adaptation with Resnet18, we searched learning rate over the set $\{0.1, 0.3, 0.01, 0.03, 0.001, 0.003\}$. The reason for having a larger set of learning rates for these methods is that we found 0.001 and 0.003 work better for these methods. For Ditto, we used λ from the set $\{0.01, 0.1, 1, 2\}$. We used 1000 rounds of communication for the reported results and observed that they were sufficient to achieve convergence as shown in Figure 2.

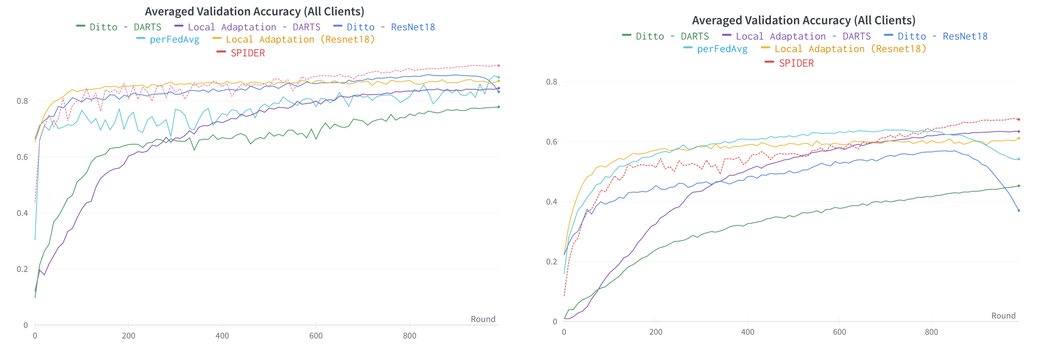
Another interesting feature we observed from empirical experiments is that the hyper-parameters for SPIDER were pretty stable for all three datasets, lr 0.01 and lambda 0.01 were the optimal values. However, the optimal hyper-parameters changed for the other three methods from dataset to dataset. We provide the best hyper-parameters selected for the reported results in Table 3.

A.3 ARCHITECTURE SEARCH SPACE

As mentioned in Section 4.2, we have used DARTS (s2) search space Wang et al. (2021) in our proposed work. During the search, we are using a total of 8 cells. Each cell consists of 14 edges; each edge connects two intermediate representations (node) by a mixture of two operations frequently used in various modern CNNs (e.g., sep convolution 3x3 and skip connection). In this search space, there are two types of cells: normal and reduction cells. Hence, our supernet $\mathcal{A}$ consists of search space of normal cells and reduction cells, each with a matrix of size 14x2, 14 signifies the total number of edge connections and 2 denotes the total number of operations at each edge connection. Hence, the mask M for the supernet has dimensions 28x2. Initially, all entries are 1 in the search space masks for both supernet M and local child model M_k. As the search progresses via progressive NAS 2, some operations and edges are removed based on the perturbation criterion explained in Algorithm 2. We visualize the normal cells searched for CIFAR10 in Figure 7. It can be seen from Figure 7 that the searched cells are edge-wise and operation-wise heterogeneous from client to client. It is important to note that the same cells follow the same construction; therefore, if one operation is selected for one normal cell at a particular edge, the same operation will be selected for the same edge at all normal cells.

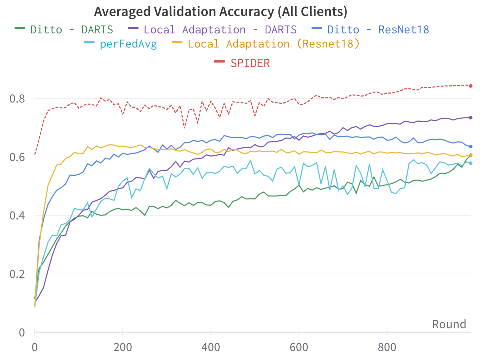
B PERSONALIZED NAS VERSUS GLOBAL NAS

In this section, we compare our proposed scheme of architecture personalization with three different settings of NAS. First, we compare our method to a basic setting of NAS, which is a centralized



(a) Average Validation Accuracy comparison on CIFAR10 Dataset

(b) Average Validation Accuracy comparison on CIFAR100 Dataset



(c) Average Validation Accuracy comparison on CINIC10 Dataset

Figure 2: Comparison of our proposed method, SPIDER, with other state-of-the-art personalization methods (perFedAvg, Ditto, and local adaptation) with CIFAR10 (a), CIFAR100 (b) and CINIC10 datasets. For both of these datasets, our method outperforms the representative sota personalization methods.

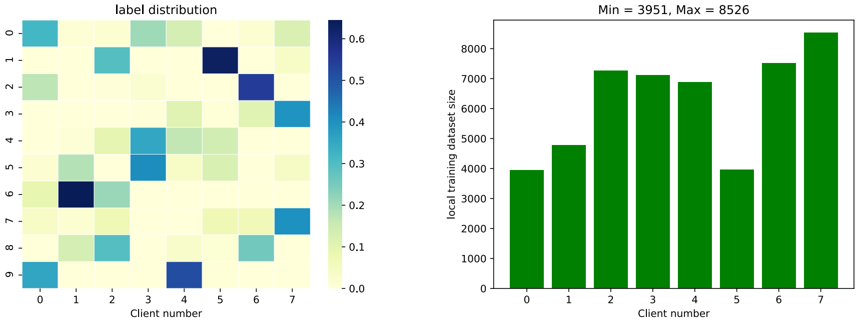
Table 3: Manually Searched Hyper-parameters (learning rate (lr) and λ for SPIDER and the other personalization schemes for three datasets; CIFAR10, CIFAR100 and CINIC10

Method	**CIFAR10**	**CIFAR100**	**CINIC10**
SPIDER	lr = 0.01, $\lambda = 0.01$	lr = 0.01, $\lambda = 0.01$	lr = 0.01, $\lambda = 0.01$
Local Adaptation - ResNet18	lr = 0.001	lr = 0.001	lr = 0.3
Ditto - ResNet18	lr = 0.001, $\lambda = 1$	lr = 0.001, $\lambda = 0.1$	lr = 0.1, $\lambda = 2$
perFedAvg - ResNet18	lr = 0.003	lr = 0.001	lr = 0.01
Local Adaptation - DARTS	lr = 0.01	lr = 0.01	lr = 0.01
Ditto - DARTS	lr = 0.01 , λ = 1	lr = 0.01, λ = 0.1	lr = 0.01, λ = 0.1

NAS. Next, we compare it to Federated NAS, which searches a global architecture in a federated manner and then trains it via FedAvg. In the next experiment, we add an l2 regularization-based PFL scheme to train the global architecture searched via Federated NAS. This investigation can help us appreciate the benefits of architecture personalization in both centralized and federated learning settings.

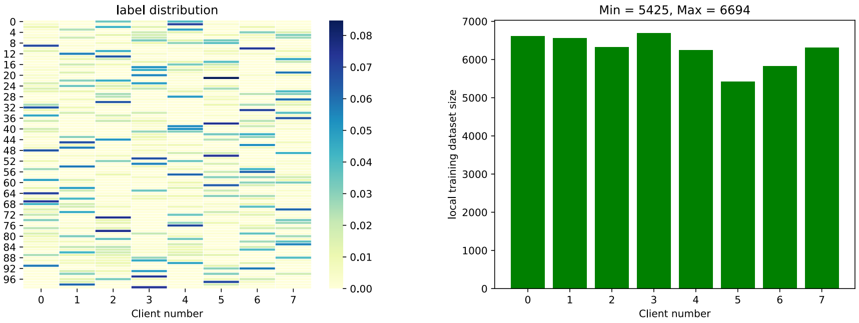
B.1 CENTRALIZED NEURAL ARCHITECTURE SEARCH

In this experiment setting, each silo performs a local neural architecture search. Each silo uses only one model, Supernet, and deploys the perturbation-based NAS Searcher locally **without any**



(a) Label distribution per client *(b) Image distribution per client*

Figure 3: CIFAR10: LDA distribution



(a) Label distribution per client *(b) Image distribution per client*

Figure 4: CIFAR100: LDA distribution

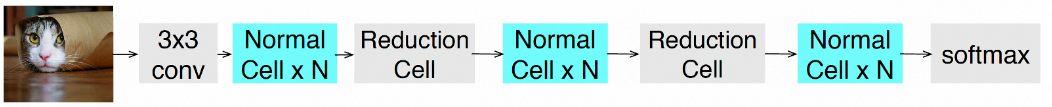
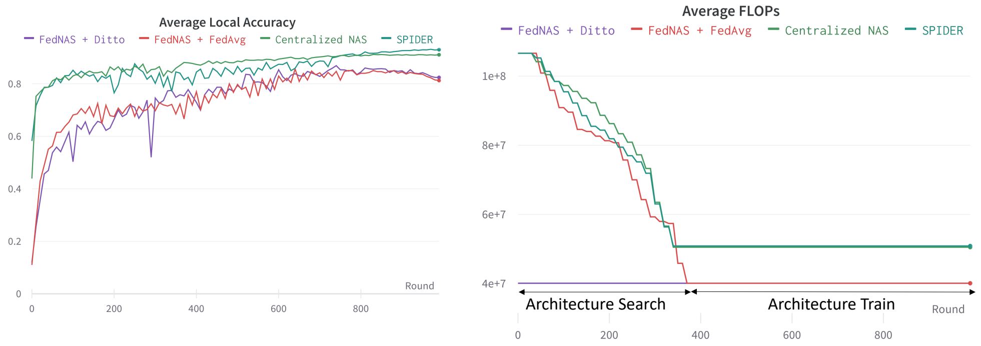


Figure 5: Supernet $\mathcal{A}$ search space. Note that since we have 8 cell based search space, N = 2 for our construction.

collaboration from the other silos. The Local Supernet has the same hyper-parameters of warmup (30 local epochs) and recovery period of 20 local epochs after every pruning step, followed by local training with a total of 1000 epochs. This search process can be analyzed from Figure 6b, where the local Supernet's average FLOPs decrease as the search progresses. Figure 6a represents the average validation accuracy across each silo. We obtain 91% average validation accuracy on the CIFAR10 dataset without any collaboration, which outperforms various PFL baselines, and performs as good as FedorAS Dudziak et al. (2022). This shows the significance of tailoring architectures to the silo's data distributions even for the centralized training case.

B.2 FEDERATED NEURAL ARCHITECTURE SEARCH

Next, we analyze NAS performance in a federated setting. We analyze the federated neural architecture search setting, where only one Supernet is maintained at each silo. Each silo trains the Supernet following the FedAvg algorithm for 30 rounds of warmup. Next, the neural architecture search is performed following perturbation-based NAS at the server using the server's test/validation data. After NAS, the architecture is obtained and trained in a federated manner by the FedAvg algorithm. This can be considered a slight variation of FedNAS He et al. (2020b) where we replace MiLeNAS He et al. (2020d) with a state-of-the-art NAS method, Perturbation-based NAS Wang et al. (2021). We obtain a total of 86% average validation accuracy, highlighting that a global model, even searched via NAS, may not be able to achieve persoalization across all silos due to data heterogeneity.



(a) Average Validation Accuracy comparison on CIFAR10 Datast

(b) Average Flop counts of the Local Architectures

Figure 6: Average Validation Accuracy and Average Flops comparison of SPIDER with the centralized NAS, Federated NAS (FedNAS + FedAvg), and Federated NAS (FedNAS) combined with Ditto (FedNAS + Ditto).

B.3 Federated Neural Architecture Search and Personalized Federated Learning

Next, we analyze the significance of PFL in the context of federated NAS. We combine l2 regularization-based PFL method, Ditto Li et al. (2021b) for training the architecture searched via federated NAS in B.2. We find that this architecture only yields 87% average validation accuracy which is 6% lower than the average validation accuracy we obtain via SPIDER. This shows that architecture-based personalization as achieved via SPIDER can be stronger than the only weight parameters-based personalization.

B.4 Architecture Personalization Gain

In this subsection, we analyze the personalization gains of the architectures searched via SPIDER. In this regard, we visualize the normal cells of all the silos on the DARTS s2 search space for CIFAR10 datasets in Figure 7. We observed heterogeneity in architecture each client searches and trains using the proposed method SPIDER. It can be seen from Figure 7 that the searched cells are edge-wise and operation-wise heterogeneous from client to client. In order to further investigate whether the architecture searched by one client is best suited for its own data distribution, we perform the following experiment.

As also stated in subsection 5.2.2, for any given client i, we apply its final architecture with its learned weights to another client j's data and finetune i's architecture on j's data. We denote the accuracy we obtain on data j via architecture i as AP_{ij}. We calculate the architecture i personalization gain or drop (if negative) as follows,

$$\boldsymbol{g}_i = \frac{\sum_{j=0,j\neq i}^{c-1}(AP_{ii} - AP_{ji})}{c-1}, \tag{8}$$

where c is the total number of clients. In Table 4a, and 5a, we report AP_{ji} for the CIFAR10 and CIFAR100 datasets respectively across all silos. In addition, we report $AP_{jj} - AP_{ji}$ values in Table 4b and 5b for CIFAR10 and CIFAR100 datasets, respectively. We also list the quantity $\boldsymbol{g}_j$ that lists the architecture j's personalization gain against other clients's architectures. In addition to the personalization gain, we also report per-client accuracy for all the personalization schemes in Figure 8, where it can easily be observed that SPIDER outperforms the other state-of-the-art personalization methods on per silo based performance.

C Computational Cost Analysis of Training

In this section, we analyze the compute cost of the proposed method and the other representative personalized federated learning (PFL) methods. It is important to note that we obtain the peak memory and training time values for all methods on the NVIDIA RTX 2080Ti GPU card, i.e, each

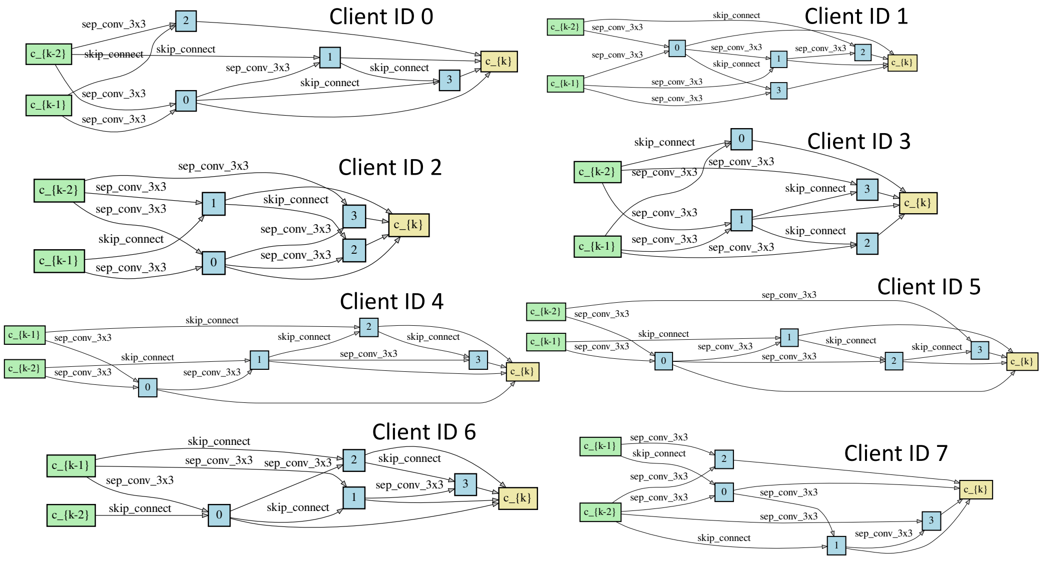


Figure 7: Searched Architectures (Normal Cells): Each normal cell k takes the outputs of previous cells, cell $k-2$ and cell $k-1$, as its input. Each cell contains seven nodes: two input nodes, one output node, and four intermediate nodes inside the cell. The output node concatenates all intermediate nodes' output depth-wise. It can be observed that the searched cells are edge-wise and operation-wise heterogeneous from client to client.

Table 4: CIFAR10: Personalization Gain Analysis

(a) Accuracy values of Architecture i on j's data (where i and j represent client IDs.

(j, i)	i = 0	i = 1	i = 2	i = 3	i = 4	i = 5	i = 6	i = 7
j = 0	88.15	85.02	88.54	85.58	84.46	86.71	88.54	85.67
j = 1	92.99	95.25	93.75	92.78	92.24	92.6	92.45	91.9
j = 2	92.64	92.71	95.07	95.00	92.5	94.23	93.26	93.6
j = 3	90.83	90.83	91.83	92.12	91.12	90.69	89.98	90.8
j = 4	89.17	88.51	90.47	89.24	91.13	90.18	88.2	89.17
j = 5	89.97	89.97	90.62	89.45	89.32	92.32	88.93	90.01
j = 6	90.96	89.94	91.92	89.94	89.74	91.84	93.40	91.51
j = 7	91.74	91.27	91.86	91.62	89.74	91.68	91.39	92.81

(b) Accuracy Gain/Drop Matrix ($AP_{jj} - APij$) and the resultant $\boldsymbol{g}_j$ vector

(j, i)	i = 0	i = 1	i = 2	i = 3	i = 4	i = 5	i = 6	i = 7	$\boldsymbol{g}_j$
j = 0	0.	3.13	-0.39	2.57	3.69	1.44	-0.39	2.48	1.79
j = 1	2.26	0.	1.5	2.47	3.01	2.65	2.8	3.35	2.58
j = 2	2.43	2.36	0.	0.07	2.57	0.84	1.81	1.47	1.65
j = 3	1.29	1.29	0.29	0.	1.	1.43	2.14	1.32	1.25
j = 4	1.96	2.62	0.66	1.89	0.	0.95	2.93	1.96	1.85
j = 5	2.35	2.35	1.7	2.87	3.	0.	3.39	2.31	2.57
j = 6	2.44	3.46	1.48	3.46	3.66	1.56	0.	1.89	2.56
j = 7	1.07	1.54	0.95	1.19	3.07	1.13	1.42	0.	1.48

Table 5: CIFAR100: Personalization Gain Analysis

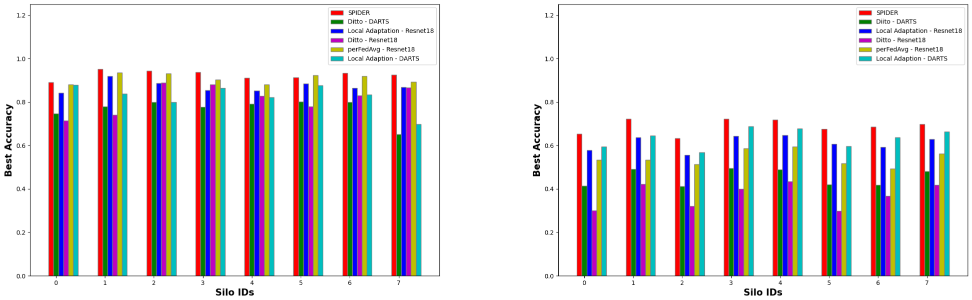
(a) Accuracy values of Architecture i on j's data (where i and j represent client IDs.

(j, i)	i = 0	i = 1	i = 2	i = 3	i = 4	i = 5	i = 6	i = 7
j = 0	65.32	62.65	62.42	62.88	62.95	62.04	62.80	62.42
j = 1	68.67	72.17	67.68	68.75	68.52	69.81	69.66	67.45
j = 2	60.41	60.08	63.38	60.17	59.37	59.61	62.09	61.05
j = 3	68.81	69.05	67.37	72.32	68.06	67.45	68.90	69.43
j = 4	68.83	69.87	67.30	69.8	71.79	69.07	70.03	69.31
j = 5	65.81	65.65	64.96	66.19	65.05	67.67	65.81	62.21
j = 6	66.49	64.84	62.76	63.88	64.40	64.93	68.48	63.97
j = 7	67.22	66.74	67.47	67.78	67.38	67.22	67.30	69.87

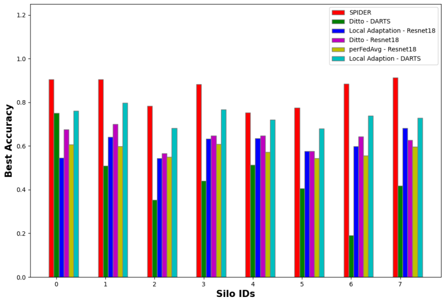
(b) Accuracy Gain/Drop Matrix ($AP_{jj} - APij$) and the resultant $\boldsymbol{g}_j$ vector

(j, i)	i = 0	i = 1	i = 2	i = 3	i = 4	i = 5	i = 6	i = 7	AP_j
j = 0	0.	2.67	2.9	2.44	2.37	3.28	2.52	2.9	2.73
j = 1	3.5	0.	4.49	3.42	3.65	2.36	2.51	4.72	3.52
j = 2	2.97	3.3	0.	3.21	4.01	3.77	1.29	2.33	2.98
j = 3	3.51	3.27	4.95	0.	4.26	4.87	3.42	2.89	3.88
j = 4	2.96	1.92	4.49	1.99	0.	2.72	1.76	2.48	2.62
j = 5	1.86	2.02	2.71	1.48	2.62	0.	1.86	5.46	2.57
j = 6	1.99	3.64	5.72	4.6	4.08	3.55	0.	4.51	4.01
j = 7	2.65	3.13	2.4	2.09	2.49	2.657	2.57	0.	2.57

silo is a physical node that has an NVIDIA RTX 2080Ti GPU card inside. The peak compute cost per silo of the representative PFL methods is 1.99GB, 2.03 GB, 2.32GB, 2.03 GB for FedorASDudziak et al. (2022), perFedAvg, Ditto and perFedAvg, respectively as shown in Table 6. SPIDER requires peak memory of around 3.42 GB per silo at the start. However, as training progresses the local model gets pruned and becomes smaller and the compute cost reduces to 2.6 GB. Since we are working in a cross-silo regime, peak memory or communication cost may not be a big concern. SPIDER takes 21 hours to complete one run, which includes both personalized architecture search and personalized architecture training. On the other hand, Local adaptation, Ditto, and perFedAvg require 2.5, 4, and 6 hours for one run of training. However, it does not include the practical scenario of the cost of hyper-parameter tuning and architecture selection tuning. For example, for Ditto, we have 18 possible iterations for our hyper-parameter set for resnet18 and the total cost becomes 72



(a) Best Accuracy per client on CIFAR10 Dataset *(b) Best Accuracy per client on CIFAR100 Dataset*



(c) Best Accuracy per client on CINIC10 Dataset

Figure 8: Per Client Performance of different personalization schemes on CIFAR10, CIFAR100 and CINIC10 datasets. Note that SPIDER outperforms the other personalization methods for most of the methods over three datasets.

hours. If we were to include the cost of hyper-parameter tuning on DARTs architecture as well, it becomes 144 hours. That's where the run time costs become comparable. Even by paying this price, we are not able to achieve optimal performance with these representative PFL schemes. Hence, in data heterogeneous settings, it is challenging for the server to select one architecture that would work for all the silos as the manual architecture search can be time-consuming.

Table 6: Accuracy versus Computational Cost Tradeoff on CIFAR10 dataset for SPIDER versus other representative personalized federated learning techniques such as Local Adaptation, Ditto, perFedAvg, FedorAS.

Method	**Training Time**	**Accuracy**	**Peak Memory Cost**
SPIDER	21 h	93%	3.42GB
Local Adaptation - ResNet18	2.5 h	85%	2.03GB
Ditto - ResNet18	4 h	90%	2.32GB
perFedAvg - ResNet18	6h	91%	2.03GB
FedoARS Dudziak et al. (2022)	–	91%	1.99GB

It is also important to study it from a tradeoff between accuracy and compute power perspective. As an example, we compare SPIDER with FedorAS Dudziak et al. (2022), a NAS-based scheme that personalizes architectures to compute-based clusters such that silos belonging to one cluster are assigned the same architecture. We find that with SPIDER, we can achieve high achieve accuracy (93%) across silos by paying the price of high compute (3.4 GB peak memory cost), whereas FedorAS, which personalizes architecture on a compute-power-based cluster level basis, consumes (1.9GB) peak memory cost only but pays the price of lower accuracy (91%). In our ablation study, we find that if each silo performs centralized NAS and local training, they can achieve 91% accuracy as well. However, SPIDER assists each silo to search for a personalized architecture tailored to its own data distribution while also learning from other silos. This architecture personalization yields a 2% performance gain on the CIFAR10 dataset that can be a significant gain for organizations that require high accuracy without the constraint of compute cost. Hence, the main motivation of our work has been to empower clients to search for architectures that are better suited for their local data distribution. This yields better performance in terms of accuracy as compared to the state-of-the-art personalization schemes. In addition, this architecture personalization technique can reduce the cost of manual search over architectures that can become expensive because data is heterogeneous across silos and completely unknown at the server in a federated learning setting.