Resource-Aware Heterogeneous Federated Learning with Specialized Local Models

Sixing Yu, Pablo Munoz and Ali Jannesari

May 20, 2024

Resource-Aware Heterogeneous Federated Learning with Specialized Local Models

Sixing Yu[1], J. Pablo Muñoz[2], Ali Jannesari[1]

[1]Iowa State University, [2]Intel Labs
{yusx, jannesar}@iastate.edu, pablo.munoz@intel.com

Abstract. Federated Learning (FL) is extensively used to train AI/ML models in distributed and privacy-preserving settings. Participant edge devices in FL systems typically contain non-independent and identically distributed (Non-IID) private data and unevenly distributed computational resources. Preserving user data privacy while optimizing AI/ML models in a heterogeneous federated network requires us to address data and system/resource heterogeneity. To address these challenges, we propose Resource-aware Federated Learning (RaFL). RaFL allocates resource-aware specialized models to edge devices using Neural Architecture Search (NAS) and allows heterogeneous model architecture deployment by knowledge extraction and fusion. Combining NAS and FL enables on-demand customized model deployment for resource-diverse edge devices. Furthermore, we propose a multi-model architecture fusion scheme allowing the aggregation of the distributed learning results. Results demonstrate RaFL's superior resource efficiency compared to SoTA.

Keywords: Federated Learning, Resource Heterogeneous, AutoML

1 Introduction

Federated Learning (FL) has emerged as a privacy-aware decentralized AI/ML model training and optimizing paradigm, widely adopted by the technology industry, to leverage the massive data generated at the edge (e.g., mobile phones and IoT devices). FL involves local client nodes distributively training and optimizing AI models, with an aggregating server integrating the decentralized results without accessing users' private data.

However, FL introduces two key challenges: data heterogeneity and resource/system heterogeneity, making it difficult to scale. The private data collected on edge devices is non-independent and identically distributed (Non-IID), causing uncertainties and optimization failures when training with naïve decentralized methods [4]. Additionally, resource heterogeneity, where edge devices have different specifications and computational capacities, leads to inefficient resource utilization. Moreover, the expensive communication overhead produced by frequently sharing weights/gradients between edge clients and servers becomes a bottleneck for scaling up the network, especially in cross-device and model-centric FL settings involving possibly millions of edge devices, resulting in massive bandwidth consumption.

State-of-the-art (SoTA) FL baselines mainly focus on addressing data heterogeneity (i.e., non-IID data on edge devices), while ignoring system and resource heterogeneity among edge devices, which exhibit significant variations, particularly in cross-device FL settings. Simply deploying the same model architecture to every client leads to poor resource utilization, causing resource-hungry edge devices to consume precious energy supplies while underutilizing more capable edge devices. Although AutoML solutions, such as Neural Architecture Search (NAS) [20, 1], have achieved success in generating high-performance neural architecture models for resource-heterogeneous environments, integrating NAS into existing FL solutions presents challenges, as it requires identical neural architecture among clients for aggregating decentralized training results. Recent efforts [8, 27] dedicated to integrating NAS, network pruning, and weights dropout in FL rely on zero-padding model weights to enable multi-neural architectures, helping in model sparsity but offering limited contributions to model size reduction and communication efficiency.

To address both data and system heterogeneity, we propose Resource-aware Federated Learning (RaFL), a multi-architecture FL framework. RaFL leverages a weight-sharing supernet to efficiently specialize resource-aware models on edge devices, tailored to their specific resource constraints. This approach enables the deployment of diverse model architectures across edge-FL clients while maintaining the ability to aggregate and share knowledge. Central to RaFL is the use of a small-size **knowledge network** that cooperates with the resource-aware local model for neural knowledge sharing among clients. RaFL clients engage in deep mutual learning [30] to co-train their network pairs and diffuse knowledge into their knowledge networks. The RaFL server then aggregates the local knowledge from each client's knowledge network, effectively combining the decentralized training results and supporting multi-architecture FL. Given the availability of public data, RaFL provides the option of using ensemble distillation to improve the robustness of knowledge fusion. RaFL makes the following contributions:

- Mitigates resource/system heterogeneity by deploying resource-aware neural architectures, and maximizes resource utilization.
- Provides ensemble knowledge distillation and transfer learning algorithms specifically designed for federated learning, which aim to improve the robustness of knowledge fusion.
- Employs high-performing specialized neural architectures to accelerate inference at the edge.
- Reduces FL communication overhead by using a smaller interceding knowledge network.
- Provides alternative configurations to take advantage of transfer learning, and public data, making it scalable and applicable in real-world scenarios.

2 Related Works

Federated Learning The pioneering FL research by [19] introduced FedAvg, which combined local stochastic gradient descent (SGD) model optimization with global model averaging. Recently, several variants of FedAvg have emerged. For example, FedProx [17] was proposed to address convergence issues associated with data and device heterogeneity, which was achieved by permitting the participation of "straggler" devices and introducing a proximal term in the local loss. FedNova [23] normalized and scaled local updates by modifying weights to mitigate gradient bias. SCAFFOLD [13] introduced gradient-based control variates to correct client drift and speed up convergence. These SoTA methods inherently assume uniformity in edge resource capacities by deploying architecturally identical models onto resource-heterogeneous edge devices. In contrast, heterogeneous edge computational capacities require the deployment of specialized client network architectures. As such, accounting for potential network architectural diversity is imperative.

Personalized FL Personalized FL [5, 3, 6, 12, 29] has gained attention as a solution to the challenges posed by *statistical heterogeneity*, such as non-IID data distribution, varied sample sizes among clients, and imbalanced class distributions across local datasets. Uniform FL models can degrade overall performance, while personalized models increase robustness. Personalized FL allows edge clients to tune model weights to better fit local data, but these approaches still adopt the same model architecture at the client, failing to address the heterogeneity of computational capacities.

Knowledge Distillation in Federated Learning Knowledge distillation (KD) has been adopted in FL as a means to address model heterogeneity [15, 7, 21, 22, 18]. For example, Fed-ensemble [22] integrates the prediction output of all client models; FedKD [24] proposes an adaptive mutual distillation framework to learn a student and a teacher model simultaneously on the client side; FedDF [18] distills the ensemble of client and teacher models to a server student model; [15], proposes the application of a public dataset as a medium of exchanging knowledge among customized client models; [28] proposes the use of KD to transfer knowledge between a global network and a student network rather than using a direct assignment. In contrast, our approach utilizes local deep mutual learning [30] coupled with an optional ensemble-based multi-model fusion in the cloud whenever public data is available. In the absence of public data, we perform global model aggregation. This makes our pipeline quite flexible and applicable in real-world scenarios.

3 Methodology

There are three main stages in RaFL: resource-aware model specialization, local knowledge fusion with deep mutual learning, and cloud knowledge aggregation (Figure 1). RaFL performs an on-demand model architecture search from weight-sharing supernet [1] to tailor to the resource utilization requirements of edge

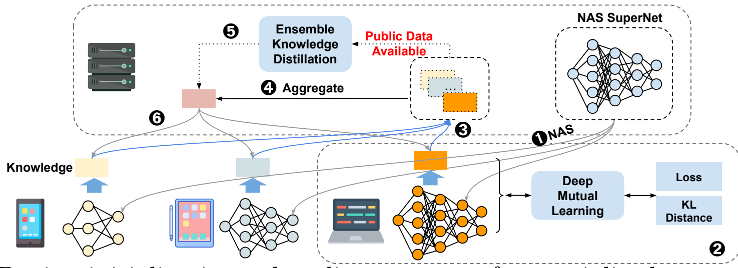


Fig. 1: During initialization, edge clients request for specialized resource-aware models from the weight-sharing supernet with neural architecture search (NAS). During local training, the local resource-aware model and knowledge network are co-trained together via deep mutual learning. The server fuses the neural knowledge from clients and performs optional ensemble learning if public data is present.

devices, unlike mainstream FL systems that deploy identical model architectures on all devices (❶ in Figure 1). Local devices train their specialized model on local data and transfer the model's knowledge to a smaller knowledge network via deep mutual learning (❷). Edge clients then communicate their local knowledge to the cloud server (❸), which aggregates the received knowledge into a global knowledge network (❹). When public data is available, RaFL optionally provides ensemble distillation to further improve the robustness of the knowledge aggregation stage (❺). Finally, the RaFL server transfers the global knowledge to the edge devices (❻).

3.1 Resource-aware Federated Learning using Specialized Models

Mainstream FL solutions assume homogeneous resource capacities across clients, which is impractical in cross-device settings with varying computational resources. Simply deploying an identical model architecture leads to poor resource utilization. RaFL proposes a resource-aware federated neural architecture search to obtain resource-tailored models for heterogeneous edge devices.

We first deploy a weight-sharing super-network on the cloud. During FL initialization, clients query this supernet [1] to obtain resource-aware subnets matched to their computational constraints. The supernet is trained by minimizing a validation loss over a distribution of subnet architectures (Equation 1).

$$\min_{\Theta} \sum_{arch_i} L_{val}(C(\Theta, arch_i)) \tag{1}$$

Where Θ represents the super-network weights, $arch_i$ is a subnetwork configuration, and $C(\Theta, arch_i)$ samples the subnetwork weights from the super-network. Once trained, subnetworks can be efficiently derived without retraining the super-network.

In the NAS process, RaFL defines a search space $\mathcal{S}$ that encompasses the possible architectures for the specialized models. The search space is constrained by a set of resource requirements $\mathcal{R}$, which includes memory, computational power, and energy consumption limitations of the edge devices. The objective is to find an optimal architecture $arch^*$ that maximizes performance while satisfying the resource constraints:

$$arch^* = \arg \max_{arch \in \mathcal{S}} P(arch) \quad \text{s.t.} \quad R(arch) \leq \mathcal{R} \tag{2}$$

where $P(arch)$ represents the performance metric of the architecture $arch$, and $R(arch)$ denotes the resource consumption of $arch$. RaFL employs a search strategy to efficiently explore the search space and extract the specialized models M_i from the supernet, tailored to the specific capabilities of each edge device i. This process ensures that the obtained models maximize resource utilization and performance in the federated learning setting.

3.2 Local Knowledge Fusion

Traditional FL methods deploy a uniform model across clients, enabling simple weight/gradient averaging for aggregation. However, in RaFL, where varying resource-aware architectures are used, weight averaging is infeasible. Instead, we propose communicating extracted knowledge from local models via deep mutual learning (DML) [30].

For the l^{th} client C_l, we denote its resource-aware model as θ_l and a downloaded knowledge network as θ_l^k. C_l jointly optimizes θ_l and θ_l^k on its local data. For an input batch, we first computes the cross-entropy loss $L_c(\theta; B)$ (Equation 3) for both networks independently.

$$L_c(\theta; B) = -\frac{1}{|B|} \sum_{x,y \in B} y^T \log(\sigma(\theta(x))) \tag{3}$$

It then measures the Kullback-Leibler (KL) divergence $D_{KL}(\theta_l || \theta_l^k; B)$ (Equation 4) between the predicted distributions.

$$D_{KL}(\theta_l || \theta_l^k; B) = \frac{1}{|B|} \sum_{x,y \in B} \sigma(\theta_l(x)^T) \log(\frac{\sigma(\theta_l(x))}{\sigma(\theta_l^k)}) \tag{4}$$

The mutual learning loss (Equations 5 and 6) combines the cross-entropy loss and KL divergence, enabling learning from data while aligning the networks.

$$L_{\theta_l} = L_c(\theta_l; B) + D_{KL}(\theta_l^k || \theta_l; B) \tag{5}$$

$$L_{\theta_l^k} = L_c(\theta_l^k; B) + D_{KL}(\theta_l || \theta_l^k; B) \tag{6}$$

DML outperforms solo learning due to the dissimilarity between the participating models (θ_l and θ_l^k). Their varying architectures and the intermittent aggregation of θ_l^k enable learning different data representations, allowing them to capture each other's knowledge. Moreover, while the smaller θ_l^k reduces communication overhead, co-learning with the larger θ_l boosts its performance.

As clients communicate their θ_l^k networks, they share local knowledge and receive global knowledge. Successive DML steps further converge both θ_l and θ_l^k

on the global data distribution. Overall, initializing clients with resource-specific NAS subnetworks paired with a smaller θ_l^k addresses device heterogeneity and reduces communication overhead.

3.3 Cloud Knowledge Aggregation and Ensemble Distillation

RaFL utilizes the knowledge network as a medium to support interoperability among multi-model FL clients. The function of the knowledge network is to exchange knowledge among Non-IID clients (Figure 1 ❸ and ❻). RaFL provides two model fusion solutions, weight aggregating and ensemble knowledge distillation, to support comprehensive FL practical scenarios (Figure 1 ❹ and ❺).

Assume θ^k is the global knowledge network, and θ_l^k as the l^{th} client's knowledge network. We select a set of communication clients S in the current communication round. Once we receive the knowledge from edges, we aggregate the knowledge networks (Equation 7).

$$\theta^k \leftarrow \frac{n_l}{N_S} \sum_{l \in S} \theta_l^k \tag{7}$$

Here, N_S is the total data size in selected clients S, and n_l is the number of data samples in the l^{th} client. When public data is available, RaFL provides an ensemble knowledge distillation option for boosting cloud aggregation. RaFL ensembles knowledge networks using the average logits strategy (Equation 8) to produce a combined output on the public dataset. The outputted soft labels are then coupled with the unlabeled dataset to train the global knowledge network in tandem. The distillation loss is defined in Equation 9.

$$\theta_{ens}(x) = Avg(\theta_l^k(x))_{l \in S} \tag{8}$$

$$L_d = D_{KL}(\theta_{ens} || \theta^k; B) \tag{9}$$

Notably, the knowledge network is a small-sized network derived from the NAS super-network. Our ablation study shows the trade-off between the size and the overall performance of the knowledge network, indicating no significant gain in increasing the size of the knowledge network.

4 Experiments

4.1 Experimental Setup

Datasets and models. We conducted experiments on datasets consistent with the baselines: CIFAR-10/100 [14], FEMNIST [2] under Non-IID benchmark settings [16]. The models we deployedwere sampled from MobileNetV2/V3 [11, 10] or ResNet [9] super-networks [1]. To avoid confusion, we identify networks by their FLOPs, e.g., we identify ResNet-34 as ResNet with 76 MFLOPs.
Federated Learning settings. We set up different numbers of clients from 30 to

3000 in our experimental FL environment, with 10% to 70% client participation rate in each round of communication. We simulated scaled, dynamic sporadic, and asynchronous FL scenarios. Clients are allocated to Non-IID local datasets following the benchmark's Non-IID settings [16].

NAS settings. We initialize an weight-sharing supernet with architectures for ResNet, MobileNetV2, and MobileNetV3.

Baselines. We compare RaFL with state-of-the-art (SoTA) FL algorithms, including:

- Strong FL optimization methods, such as FedAvg [19], FedProx [17], FedNova [23], SCAFFOLD [13], SPATL [26].
- Knowledge distillation-based methods, such as FedDF [18].
- Neural architecture search in FL methods, such as FedNAS [8].

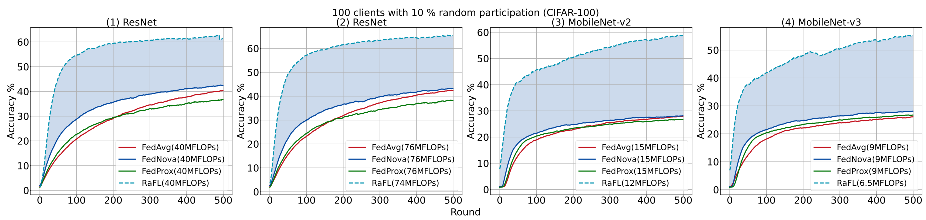


Fig. 2: Comparison of RaFL with SoTAs under **vanilla aggregation** setting.

4.2 Learning Efficiency

We explore the learning efficiency and optimization ability of RaFL by evaluating the training performance with respect to communication rounds and comparing it with FedAvg [19], FedNova [23], and FedProx [17]. We consider all possible practical situations ((a), (b), and (c) settings in below) for a fair and extensive comparison.

(a) Vanilla aggregation. RaFL with vanilla aggregation (Figure 1 step ❹ and Equation 7) is used to aggregate local models in each communication round without requiring public data. Figure 2 shows the comparison results on training performance versus communication rounds under different model architectures and capacities. RaFL outperforms the baselines with a large margin and exhibits a stable optimization process while consuming fewer resources. For example, when optimizing MobileNet-v3, RaFL produces 28% higher accuracy than the baselines. Figure 4a illustrates that RaFL consistently outperforms the baselines across diverse settings and achieves higher convergence accuracy. RaFL effectively resists overfitting and shows better learning capacity (Figure 5a).

(b) Public data with ensemble and knowledge distillation. RaFL offers an ensemble distillation option to enhance server aggregation (steps ❹ and ❺). Figure 3 RaFL(b) shows that RaFL equipped with ensemble distillation outperforms strong baselines with significant margins. Ensemble distillation improves and stabilizes training in sporadic connected FL (Figure 3 (4)). However, its effectiveness depends on the similarity of public and private client data distributions.

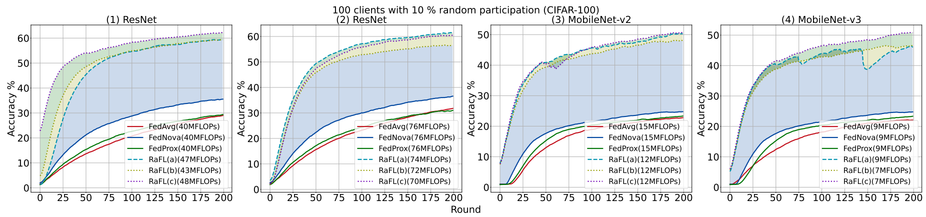
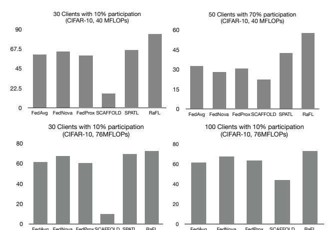
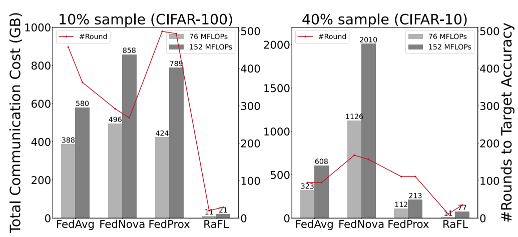


Fig. 3: Comparison of SoTAs with RaFL using **(a) vanilla aggregation**, **(b) ensemble distillation** and **(c) transfer learning** settings.

(c) Transfer learning with NAS. RaFL takes advantage of transfer learning on its pre-trained NAS subnetworks when specialized networks are applied to the Non-IID FL dataset. Figure 3 RaFL(c) shows that RaFL (c) benefits from pre-trained initial weights, resulting in high accuracy in the initial training stage and quick convergence during fine-tuning.

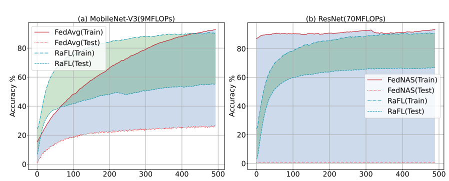


(a) Convergence accuracy overhead under various FL settings.

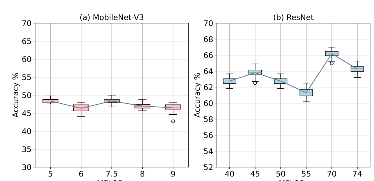


(b) Communication cost to achieve target accuracy.

Fig. 4: Performance comparison of RaFL and baselines.



(a) Local training accuracy vs. test accuracy.



(b) RaFL under resource heterogeneous setting.

Fig. 5: Performance analysis of RaFL.

4.3 Communication Efficiency

RaFL exhibits significant superiority in communication efficiency compared to SoTAs, attributed to utilizing smaller knowledge networks for communication. We demonstrate RaFL's communication efficiency by optimizing models to reach

Table 1: Communication cost to achieve target accuracy.

Method	Model Arc.	Resource Usage (MFLOPs)	Clients	Dataset	Target Accuracy	Communication Cost Round/Client	Total	Δ Cost	Speed Up
FedAvg [19]	ResNet	40	100	CIFAR100	40%	0.85 GB	388 GB	0GB	(1 ×)
		76				1.6 GB	580 GB	0GB	(1 ×)
		40		CIFAR10	60%	3.4 GB	323 GB	0GB	(1 ×)
		76				6.4 GB	608 GB	0GB	(1 ×)
FedNova [23]	ResNet	40	100	CIFAR100	40%	1.7 GB	496 GB	+108 GB	(0.78 ×)
		76				3.2 GB	858 GB	+278 GB	(0.68 ×)
		40		CIFAR 10	60%	6.7 GB	1126 GB	+803 GB	(0.29 ×)
		76				12.8 GB	2010 GB	+1402 GB	(0.30 ×)
FedProx [17]	ResNet	40	100	CIFAR100	38%	0.85 GB	424 GB	+36 GB	(0.92 ×)
		76			40%	1.6 GB	789 GB	+209 GB	(0.74 ×)
		40		CIFAR 10	60%	3.4 GB	377 GB	+54 GB	(0.86 ×)
		76				6.4 GB	710 GB	+102 GB	(0.86 ×)
RaFL (Ours)	ResNet	28	100	CIFAR100	40%	0.66 GB	13.2 GB	-374.8 GB	**(29.4 ×)**
		51				1.07 GB	31.0 GB	-549 GB	**(18.70 ×)**
		28		CIFAR10	60%	2.2 GB	77 GB	-246 GB	**(4.19 ×)**

target performance and achieve converged performance, considering the vanilla aggregation setting (Section 4.2.a).

Optimizing Model to Achieve Target Performance. Table 1 shows that RaFL consistently outperforms SoTAs in communication overhead under various configurations. For example, when using ResNet on CIFAR-100, RaFL reduces communication cost by up to 29.4× compared to FedAvg [19] while using significantly less computational resources. Figure 4b further demonstrates RaFL's superiority on larger model capacities (76 MFLOPs and 152 MFLOPs), requiring fewer communication rounds to achieve target performance.

RaFL's communication efficiency stems from two main factors: (1) high-performance neural architectures generated using Neural Architecture Search, enabling faster convergence, and (2) knowledge network communication, resulting in a reduced communication cost with up to 4× less bandwidth per round (Table 1).

Optimizing Model for Convergent Performance. Table 2 summarizes the results on CIFAR-100, showing that RaFL uses less communication cost while achieving significantly better model performance. For instance, RaFL reaches around 65% accuracy on ResNet-74MFLOPs, while the next best method, FedNova, converges at around 43% with a total cost 1200 GB higher.

Overall, RaFL results in a lower communication cost to achieve both target performance and overall convergence across a wide range of FL settings. Additionally, the model deployed by RaFL requires fewer FLOPs at the edge and demonstrates lower resource consumption while achieving higher accuracy, enabling faster inference and better downstream task applications.

4.4 Resource-aware System Heterogeneity

We investigated RaFL's ability to cope with resource heterogeneity by evaluating its resource utilization efficiency and learning efficiency under system heterogeneity.

Resource Utilization Comparison. Compared with uniform model deployment FL methods (such as FedAvg [19], FedNova [23], FedProx [17], SCAFFOLD [13]), Table 3 shows that RaFL demonstrates significant resource utilization efficiency and high accuracy overhead. At least 90% of the resources were utilized by RaFL across different resource constraints, with the highest utilization being 99%. In

Table 2: Communication cost to model converge

Method	Model	datasets	Resource Usage (MFLOPs)	Clients	Round Cost	Total Cost	Converged Accuracy	Δ Accuracy
FedAvg [19]	ResNet	CIFAR-100	40 MFLOPs	100	0.85 GB	425GB	(40.00±1.21)%	0%
			76 MFLOPs		1.6 GB	800GB	(42.21± 0.80)%	0%
	MobileNet-V2		15 MFLOPs		280MB	137GB	(27.80±0.80)%	0%
	MobileNet-V3		9 MFLOPs		230MB	112GB	(25.89±0.87)%	0%
FedNova [23]	ResNet	CIFAR-100	40 MFLOPs	100	1.7 GB	850GB	(42.29±1.23)%	+2.29%
			76 MFLOPs		3.2 GB	1600GB	(43.02±0.76)%	+0.81%
	MobileNet-V2		15 MFLOPs		560MB	273GB	(27.98±0.55)%	+0.18%
	MobileNet-V3		9 MFLOPs		460MB	225GB	(26.25±0.82)%	+0.36%
FedProx [17]	ResNet	CIFAR-100	40 MFLOPs	100	0.85 GB	425GB	(36.43±1.10)%	-3.57%
			76 MFLOPs		1.6 GB	800GB	(38.04±1.59)%	-4.17%
	MobileNet-V2		15 MFLOPs		280MB	137GB	(26.65±0.82)%	-1.15%
	MobileNet-V3		9 MFLOPs		230MB	112GB	(24.39±0.52)%	-1.50%
RaFL (Ours)	ResNet	CIFAR-100	47 MFLOPs	100	0.66 GB	**340 GB**	**(62.01±1.84)%**	**+22.01%**
			74 MFLOPs		0.80GB	**400GB**	**(65.28±1.22)%**	**+23.07%**
	MobileNet-V2		12 MFLOPs		180MB	**88GB**	**(58.66±1.40)%**	**+30.86%**
	MobileNet-V3		6.5 MFLOPs		140MB	**68GB**	**(50.49±2.49)%**	**+24.6%**

Table 3: Resource utilization efficiency

Method	Model	Total Resource	Utilized Resources	Resource Utilization	Accuracy
Uniformed Baselines	ResNet	5GFLOPs	2.75GFLOPs	55%	41%
		8GFLOPs	5.02GFLOPs	63%	43%
	MobileNet-V2	1.2GFLOPs	1.07GFLOPs	89%	29%
	MobileNet-V3	0.7GFLOPs	0.49MFLOPs	70%	27%
RaFL	ResNet	5GFLOPs	4.63GFLOPs	**93%**	**65%**
		8GFLOPs	7.26GFLOPs	**91%**	**67%**
	MobileNet-V2	1.2GFLOPs	1.15GFLOPs	**99%**	**60%**
	MobileNet-V3	0.7GFLOPs	0.64GFLOP	**91%**	**57%**

contrast, the single model deployment in the baselines would have their resource utilization limited by the clients with the lowest resource capacity.

Model Performance under Different Resource Budget. Figure 5b illustrates the impact of different resource budgets on model accuracy optimized by RaFL. RaFL exhibits stable performance under various resource heterogeneity scenarios. Even with significant differences in preset resource overhead, RaFL maximizes learning efficiency and resource efficiency, producing stable final model performance. This adaptability and effectiveness highlight RaFL's ability to address diverse resource constraints while maintaining consistent performance across various systems.

Edge Performance under System Heterogeneity. We set up two systems with different average resource constraints (74MFLOPs and 47MFLOPs). Figure 6 (a) shows the heterogeneous resource distributions within these systems. Despite the significant variance in local client resources, Figure 6 (b) demonstrates that the models running at the edge exhibit similar performance. This can be attributed to the optimal architecture search performed by the neural architecture search under the provided resource constraints, emphasizing RaFL's effectiveness in addressing system heterogeneity and ensuring consistent performance across diverse edge devices.

4.5 Scaled and Sporadic FL

In practice, a typical FL system interacts with a large number of clients. The system is dynamic and asynchronous, meaning new clients may join and leave the network at any stage of FL training, and the number of participating clients may be large. Figure 7 shows RaFL's performance under scaled and sporadic FL settings. Large-scale federated learning is one of the biggest challenges in FL, as

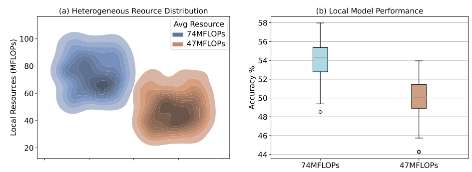


Fig. 6: Local model performance in resource heterogeneous FL.

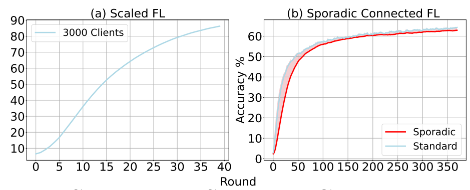


Fig. 7: Scaled and Sporadic Connected FL.

shown in Figure 7 (a), we set up 3000 clients with distinct heterogeneous local data on FEMNIST, RaFL converges quickly and remains stable. Figure 7 (b) shows the sporadic FL in CIFAR-100. We implemented a client stream to simulate a sporadic FL setting. For each round of communication, we uniformly replace 10% of clients with new ones, to mimic 10% of clients losing connections and 10% of clients joining the current stage of training. RaFL achieves impressive results under sporadic FL and produces competitive model performance compared to the standard FL setting. This is despite the total clients flowing through the sporadic FL setting equalling 3500, which is 35× more than the standard setting. The results highlight RaFL's robustness and adaptability in dynamic and large-scale federated learning scenarios.

4.6 Comparison with FL approaches that use Neural Architecture Search

Learning Efficiency Comparison. We compared RaFL to FedNAS [8], which deploys super-networks directly to edge devices and searches sub-networks locally for edge applications. Figure 5a(b) shows that FedNAS diverged due to overfitting in our experiment settings. We also analyzed DecNAS [25]. It is important to note that RaFL has a different objective compared to FedNAS [8] and DecNAS [25]. While FedNAS and DecNAS aim to optimize the NAS super-network via federated learning on user private data, RaFL focuses on efficiently deploying resource-aware specialized networks to heterogeneous clients to maximize resource utilization and achieve robust performance in dynamic and large-scale federated learning scenarios.

Distinguishing RaFL from Other NAS-based Federated Learning Approaches. Other approaches may not yield efficient neural architectures. For instance, FedNAS performs neural architecture search at the beginning of the training, which may not result in high-performance architectures. Existing methods might not effectively tackle resource heterogeneity in edge devices. For example, FedNAS conducts neural architecture search directly on edge devices, which can be computationally demanding and unsuitable for resource-limited edge devices.

Optimizing super-networks in FL environments can easily lead to divergence. These approaches might work when the number of clients is limited. However, when the number of clients in FL becomes large, it becomes almost impossible to converge the over-parameterized super-networks. In contrast, RaFL aims to efficiently deploy resource-aware specialized networks to heterogeneous clients to maximize resource utilization and achieve robust performance in dynamic and large-scale federated learning scenarios.

RaFL ingeniously avoids the above limitations by dynamically deploying specialized high-performance networks to edge clients based on their local resource overhead, enabling efficient utilization of local resources and better downstream tasks.

4.7 Comparison with KD-based FL

As shown in Table 4, utilizing knowledge distillation to aggregate local models may not enhance the learning efficiency of the FL system. The performance of knowledge distillation heavily relies on the similarity between public data and local data, making it unsuitable for all general FL environments. We compared RaFL with the knowledge distillation baseline FedDF [18] under various public data scenarios. RaFL is more robust; it initializes the global knowledge network by weighted averaging local knowledge networks using Equation 7 and then optionally performs ensemble knowledge distillation. In cases where the public data significantly deviates from the overall local data distribution in FL, RaFL experiences fewer negative effects.

4.8 Extra Burdens of Knowledge Networks

To address concerns regarding the potential extra computational overhead introduced by incorporating knowledge networks and performing local deep mutual learning in the RaFL framework, we designed and conducted experiments using two distinct FL environments: Environment A: 12 MFLOPs MobileNet-V3 and an 8 MFLOPs MobileNet-V3 knowledge network deployed at each local client. Environment B: 20 MFLOPs MobileNet-V3 deployed at each local client, directly updating the model with the SGD algorithm without knowledge distillation. Both FL environments were trained for 500 rounds separately using the same nodes and the same type of GPU (NVIDIA V100). Results show that Environment B required 49 hours and 19 minutes to complete training, while Environment A only took 48 hours and 25 minutes. This indicates that incorporating knowledge networks and performing deep mutual learning did not add significant computational overhead to the edge.

Table 4: Comparison with FedDF under different public data.

Public Data	Method	Local Dataset	Clients	Participation	Accuracy
CINIC-10	FedDF	CIFAR-100	100	10%	diverge
	RaFL-(b)		100	10%	50.51
CIFAR-10	FedDF	CIFAR-10	30	40%	73.35
	RaFL-(b)		30	40%	72.49
TinyImageNet	FedDF	CIFAR-100	100	10%	diverge
	RaFL-(b)		100	10%	54.95

5 Conclusion

In this paper, we present RaFL, a resource-aware FL approach that seamlessly integrates AutoML solutions, such as NAS, into FL environments. By providing clients with specialized networks tailored to their individual resource constraints, RaFL offers a robust solution for achieving efficient learning and improved resource utilization. Experiments conducted on various heterogeneous FL environments demonstrate RaFL's practicality and efficacy in enabling heterogeneous FL with faster convergence and enhanced communication efficiency. The promising results obtained by RaFL highlight the potential of integrating AutoML solutions into FL environments to better accommodate the diverse resource constraints of edge devices, paving the way for wider applications in real-world situations. Moreover, RaFL exhibits effectiveness in addressing FL scenarios, showcasing its potential for easy extension to decentralized and distributed training frameworks.

Acknowledgment

This research was supported by the National Science Foundation under Grant number 2243775. We also appreciate the provision of computational resources by Intel Labs for this project. Furthermore, we thank the ResearchIT team [1] at Iowa State University for their continuous and helpful assistance.

References

1. H. Cai, C. Gan, T. Wang, Z. Zhang, and S. Han. Once for All: Train one network and specialize it for efficient deployment. In *Proc. of International Conference on Learning Representations (ICLR)*, 2020.
2. S. Caldas, S. M. K. Duddu, P. Wu, T. Li, J. Konečný, H. B. McMahan, V. Smith, and A. Talwalkar. LEAF: A benchmark for federated settings. In *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, 2019.
3. C. T. Dinh, N. H. Tran, and T. D. Nguyen. Personalized federated learning with moreau envelopes. *arXiv preprint arXiv:2006.08848*, 2021.
4. P. K. et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.
5. A. Fallah, A. Mokhtari, and A. Ozdaglar. Personalized federated learning: A meta-learning approach. In *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, 2020.
6. F. Hanzely, S. Hanzely, S. Horváth, and P. Richtárik. Lower bounds and optimal algorithms for personalized federated learning. In *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, 2020.
7. C. He, M. Annavaram, and S. Avestimehr. Group Knowledge Transfer: Federated Learning of Large CNNs at the Edge. In *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, 2020.
8. C. He, M. Annavaram, and S. Avestimehr. Towards non-iid and invisible data with fednas: federated deep learning via neural architecture search. *arXiv preprint arXiv:2004.08546*, 2020.

[1] https://researchit.las.iastate.edu

9. K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc. of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
10. A. Howard, M. Sandler, G. Chu, L. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q. V. Le, and H. Adam. Searching for MobileNetV3. In *Proc. of International Conference on Computer Vision (ICCV)*, 2019.
11. A. Howard, A. Zhmoginov, L.-C. Chen, M. Sandler, and M. Zhu. Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation. In *Proc. of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
12. Y. Huang, L. Chu, Z. Zhou, L. Wang, J. Liu, J. Pei, and Y. Zhang. Personalized cross-silo federated learning on non-iid data. In *Proc. of AAAI Conference on Artificial Intelligence (AAAI)*, 2021.
13. S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh. SCAFFOLD: Stochastic controlled averaging for federated learning. In *Proc. of International Conference on Machine Learning (ICML)*, 2020.
14. A. Krizhevsky and G. Hinton. *Learning multiple layers of features from tiny images.* Master's thesis, Department of Computer Science, University of Toronto, 2009.
15. D. Li and J. Wang. FedMD: Heterogenous Federated Learning via Model Distillation. 2019.
16. Q. Li, Y. Diao, Q. Chen, and B. He. Federated learning on non-iid data silos: An experimental study. In *Proc. of IEEE International Conference on Data Engineering*, 2022.
17. T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith. Federated optimization in heterogeneous networks. In *Proc. of Conference on Machine Learning and Systems (MLSys)*, 2020.
18. T. Lin, L. Kong, S. U. Stich, and M. Jaggi. Ensemble distillation for robust model fusion in federated learning. *Advances in Neural Information Processing Systems*, 33:2351–2363, 2020.
19. H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proc. of International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.
20. J. P. Munoz, N. Lyalyushkin, Y. Akhauri, A. Senina, A. Kozlov, and N. Jain. Enabling nas with automated super-network generation. *CoRR*, abs/2112.10878, 2021.
21. H. Seo, J. Park, S. Oh, M. Bennis, and S.-L. Kim. Federated Knowledge Distillation. *Machine Learning and Wireless Communications*, page 457, 2022.
22. D. Sui, Y. Chen, J. Zhao, Y. Jia, Y. Xie, and W. Sun. FedED: Federated learning via ensemble distillation for medical relation extraction. In *Proc. of Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.
23. J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. In *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, 2020.
24. C. Wu, F. Wu, L. Lyu, Y. Huang, and X. Xie. Communication-efficient federated learning via knowledge distillation. 2022.
25. M. Xu, Y. Zhao, K. Bian, G. Huang, Q. Mei, and X. Liu. Federated neural architecture search. *arXiv preprint arXiv:2002.06352*, 2020.
26. S. Yu, P. Nguyen, W. Abebe, W. Qian, A. Anwar, and A. Jannesari. SPATL: Salient parameter aggregation and transfer learning for heterogeneous federated learning. In *Proc. of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, 2022.

27. S. Yu, P. Nguyen, A. Anwar, and A. Jannesari. Adaptive dynamic pruning for non-iid federated learning. *arXiv preprint arXiv:2106.06921*, 2021.
28. T. Yu, E. Bagdasaryan, and V. Shmatikov. Salvaging federated learning by local adaptation. *arXiv preprint arXiv:2002.04758*, 2020.
29. M. Zhang, K. Sapra, S. Fidler, S. Yeung, and J. M. Alvarez. Personalized federated learning with first order model optimization. In *Proc. of International Conference on Learning Representations (ICLR)*, 2021.
30. Y. Zhang, T. Xiang, T. M. Hospedales, and H. Lu. Deep mutual learning. In *Proc. of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.