Journal of
Sensor and Actuator Networks

MDPI

Article

Resource Management and Model Personalization for Federated Learning over Wireless Edge Networks

Ravikumar Balakrishnan *, Mustafa Akdeniz, Sagar Dhakal, Arjun Anand, Ariela Zeira and Nageen Himayat

Intel Corporation, Hillsboro, OR 97124, USA; mustafa.akdeniz@intel.com (M.A.); sagar.dhakal@gmail.com (S.D.); arjun.anand@intel.com (A.A.); ariela.zeira@intel.com (A.Z.); nageen.himayat@intel.com (N.H.)
* Correspondence: ravikumar.balakrishnan@intel.com

Abstract: Client and Internet of Things devices are increasingly equipped with the ability to sense, process, and communicate data with high efficiency. This is resulting in a major shift in machine learning (ML) computation at the network edge. Distributed learning approaches such as federated learning that move ML training to end devices have emerged, promising lower latency and bandwidth costs and enhanced privacy of end users' data. However, new challenges that arise from the heterogeneous nature of the devices' communication rates, compute capabilities, and the limited observability of the training data at each device must be addressed. All these factors can significantly affect the training performance in terms of overall accuracy, model fairness, and convergence time. We present compute-communication and data importance-aware resource management schemes optimizing these metrics and evaluate the training performance on benchmark datasets. We also develop a federated meta-learning solution, based on task similarity, that serves as a sample efficient initialization for federated learning, as well as improves model personalization and generalization across non-IID (independent, identically distributed) data. We present experimental results on benchmark federated learning datasets to highlight the performance gains of the proposed methods in comparison to the well-known federated averaging algorithm and its variants.

Keywords: federated learning; wireless edge networks; resource management; personalization

Citation: Balakrishnan, R.; Akdeniz, M.; Dhakal, S.; Anand, A.; Zeira, A.; Himayat, N. Resource Management and Model Personalization for Federated Learning over Wireless Edge Networks. *J. Sens. Actuator Netw.* **2021**, *10*, 17. http://doi.org/10.3390/jsan10010017

Received: 16 December 2020
Accepted: 17 February 2021
Published: 23 February 2021

1. Introduction

An estimated 175 zettabytes of data will be generated by 2025 globally, according to [1]. Out of this, over 90 zettabytes are estimated to be created by IoT devices alone at the network edge including autonomous vehicles, factory automation, mobile devices, and so on. According to a report by Gartner Research [2], an estimated 75% of all data will need analysis and further processing at the edge by 2025. This is due to several reasons:

First, there was an estimated 24 billion IoT devices in 2020 that generated a massive amount of data. The rate of data generation means that shipping such enormous volumes of data for analysis will face key challenges. These include latency, bandwidth costs, data privacy, and legal and ethical compliance issues. As a result, there is a growing need to process the data closer to the data generation source.

Second, while deep learning has shown tremendous success with large data and model sizes (billions of parameters), scaling compute and storage resources becomes critical. There are significant advances in compressing large models and still retaining the key performance benefits. Another rapidly emerging area is distributed training where either data or the model or a combination of both are split across several computing devices and trained without significant loss in performance.

Third, 5G is a key enabler to perform distributed learning at the edge. Not only does 5G promise peak data rates of 10 Gbps and five millisecond end-to-end latencies, it is also expected to support and operate a massive number of devices (millions per square kilometer). Further, 5G base stations are ideal choices for edge servers for orchestrating the machine learning task due to their ultra-low latency capabilities for end devices.

Owing to these factors, distributed learning at the network edge (beyond data centers) has recently emerged. A prime example for this is federated learning (FL) [3] in which client devices collectively benefit from learning a shared model from all their data without the need for central storage or exchanging raw data among devices. An edge server coordinates the learning where clients exchange model updates with the server. FL has been applied successfully in many large-scale distributed smartphone use cases including models for face recognition, language models for text autocompletion (e.g., Google GBoard), and voice recognition (e.g., Siri in iOS 13). More recently, FL has also been successfully applied in the area of medicine where it facilitates multi-institutional collaborations without sharing patient data [4]. Similarly, FL is a promising approach for sensor/IoT devices including wearables and autonomous vehicles where data privacy plays a key role.

The goal of federated learning based system is to learn a global model that performs well/has good accuracy for all clients while minimizing the overall training time. At the same time, it is crucial to allow clients to have a highly personalized model that can achieve high accuracy for end users. We highlight a subset of the challenges associated with federated learning, specifically when learning is performed over a diverse set of connected compute devices.

- Compute heterogeneity: Client devices encompass a range of form-factors of devices, CPU clock rates, and memory read/write times. As a result, the model updates from difference clients can arrive at different times at the server.
- Communication heterogeneity: Client devices experience different data rates depending on their wireless connectivities, location, and mobility characteristics. Further, such rates are also typically time-varying, which affects the overall training time.
- Statistical heterogeneity: Finding a good global model that performs well across different client data distributions will be a major challenge. Moreover, the different clients may also have unbalanced data with different numbers of training examples. This leads to variance in the model updates across the clients.
- Personalization and generalization for clients: The global model learned is an average of the local models and hence may not be optimal for individual clients. Further, there are no guarantees on the model performance even if clients are allowed to fine-tune or personalize the global model further based on local data. Furthermore, the generalization of the learned global model across a range of client data distributions is an open challenge.

Several of the above challenges motivate resource management techniques that are efficient in the presence of resource constraints at the edge including communication resources (e.g., bandwidth), computational resources (e.g., compute cycles, battery use), and data resources (dataset at clients). Efficient resource management can be viewed as an ensemble of approaches that aim to maximize distributed training performance by jointly factoring these diverse resource constraints and optimizing several training parameters such as the participating clients, time taken to complete one training iteration, etc.

This article focuses on our recent research findings on resource management and resource-efficient model personalization for federated learning. On the one hand, we develop a novel method for client selection utilizing importance sampling to jointly addresses compute, communication, and data heterogeneity. On the other hand, we develop a resource-efficient approach to perform model personalization when clients' data distributions are different that has superior per-client performance over federated learning benchmarks. To this end, we make an observation that although there exists heterogeneity in the clients' data distributions, clusters of clients are likely to form, which allows us to learn a federated meta-model from a subset of the overall clients, providing significant savings in communication and computation time.

The rest of work is organized as follows: In Section 2, we present the system model, provide a detailed review of the related literature, and highlight the open challenges addressed in this paper. In Section 3, we detail the proposed importance sampling based federated learning, as well as a resource-efficient federated meta-learning approach that

identifies and exploits client clustering. We, present through extensive experiments, the benefits of performing resource management and resource-efficient personalization for federated learning in Section 4. We summarize our conclusions and also identify related open research challenges in Section 5.

2. System Model and Related Work

2.1. System Model and Assumptions

Consider a federation of clients connected via wireless links to an edge server, as shown in Figure 1. This assumption is realistic as 5G base stations are increasingly expected to support edge computing functionality. Each client has a local training dataset that is not shared with the server. The clients support different compute rates depending on their hardware's form-factor, OS, and other running tasks. The overall goal is to learn a model from data across clients by only sharing model-specific parameters with the edge server with high accuracy and minimum overall wall clock time.

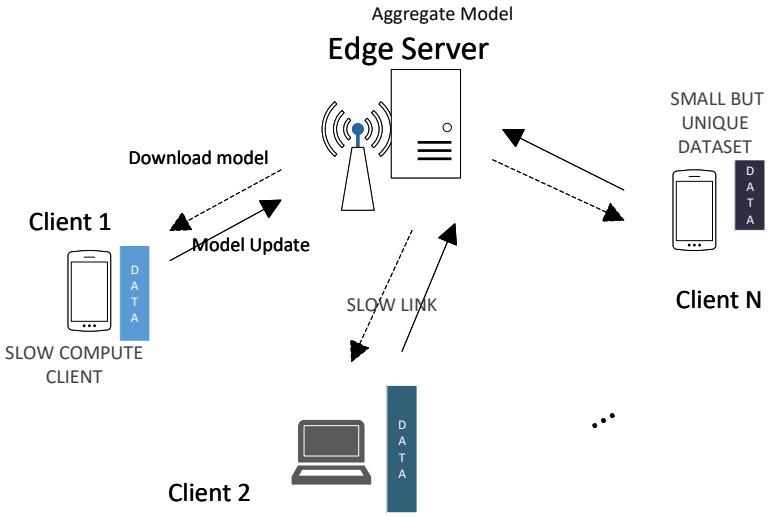

Figure 1. System model for federated learning over wireless edge networks.

2.2. Related Work

The federated averaging algorithm is a well-known approach that utilizes a random sampling of clients to perform model updates in each training round [3]. A weighted average of the local model updates from different clients is performed at the edge server iteratively to obtain the global model. The federated averaging algorithm is summarized below.

1. Initialization: The edge server selects model hyper-parameters such as the number of local training steps, step size, etc.
2. Client selection: The server selects a subset of clients at random.
3. Client computation: Each client updates the model based on the number of local steps τ and the step size η.
4. Model upload: Each client shares the updated model with the server.
5. Model aggregation: The server performs weighted combining of model updates where weights are determined by the fraction of data points at the clients.

While the optimality of this approach relies on the IID data assumption, the client data distributions, as well as the number of training examples are likely non-identical, leading to bias and variance in gradient estimates at clients. In addition, due to the heterogeneous compute and communication rates at clients, sampling clients randomly can lead to straggler issues and significantly increases the total training time for achieving model convergence.

Resource management for FL, especially over wireless networks, is becoming an active research area. The authors in [5] proposed a client selection approach to maximize the number of client updates under wireless and heterogeneous compute constraints although under IID assumptions. Analytical models to characterize the performance of FL over wireless networks were developed in [6]. The same authors also evaluated the different wireless scheduling policies such as round-robin and proportionally fair. In [7], a joint approach to learning, wireless resource allocation, and user selection was formulated as an optimization problem following which, power allocation and user selection policies were developed based on a derived closed-form expression of the convergence rate. A similar joint optimization approach was utilized to optimize federated learning, wireless resource allocation, and user selection together in the presence of non-IID data in [8]. However, such approaches are applicable in particular cases where fine-grained control of wireless resources such as transmit power or time-frequency resources are available at federated learning clients/servers. On the contrary, our approach to resource management is applicable to a broader context where federated learning application is optimized given the set of underlying communication and computational constraints. Several existing approaches for FL utilize a proximal term for local loss functions to address the non-IID data issue, most notably in [9]. Coded federated learning was developed in [10] to address the missing computations such as gradient updates due to unreliability and heterogeneity in wireless links and clients' computational capabilities. A q-fair federated learning approach was proposed in [11] where greater emphasis was given to weight updates from clients with higher local losses, thereby reducing the variance of the training accuracy distribution. Finally, approaches for model-agnostic meta-learning (MAML) were proposed in [12] where the goal was to allow efficient learning over new tasks. Very recently, a federated or distributed version of MAML was proposed in [13]. A comprehensive summary of the open research problems for FL was provided in [14].

3. Resource Management and Model Personalization for Federated Learning

3.1. Resource Management through Importance Sampling

The sampling of K out of N clients to optimize a performance objective such as maximizing a utility or minimizing the regret is well known in the scheduling literature as a combinatorial problem. One of the challenges to solving such problems is that they are NP-hard by nature. Additionally, pre-training must be conducted on at least a small amount of the dataset to determine the utility of different allocation policies. We develop a new approach to the client selection problem by considering it as a Monte Carlo sampling problem. The federated averaging algorithm aims to approximate an expectation of gradients by randomly sampling a subset of clients such that each client takes approximate gradient steps on local losses that are on average equal to the true gradient of the overall loss. However, due to the non-IID nature of data, the gradients computed at each client are not close approximations of the true gradients.

Importance sampling belongs to a set of Monte Carlo methods where an expectation with respect to a target distribution is approximated using a weighted average of sample draws from a different distribution. The application of importance sampling to stochastic gradient descent has received significant attention in recent years. The key idea is to focus the gradient computations on the "most informative" set of training data instead of providing equal priority to all training data. For example, humans are able to learn complex concepts from a small set of examples. Importance sampling methods for stochastic gradient descent and mini-batch gradient descent that achieve training speedup through variance reduction were proposed earlier, such as in [15]. While such approaches achieve training speedup in the number of global training epochs, the previously proposed importance sampling distributions do not account for the wall clock time to train each epoch.

3.2. Federated Learning with Importance Sampling

We developed an importance sampling method for federated learning that adaptively assigns sampling probabilities to clients in each training round based on the clients' compute, communication, and data resources to not only achieve training speedup in the number of training rounds, but also to reduce the wall clock training time. This is because the training time per epoch of federated learning depends on the clients' communication and compute times. Secondly, when the number of local epochs $\tau > 1$, importance sampling must be performed correctly in order to obtain unbiased gradients at each step. We addressed these two key issues in our importance sampling approach for federated learning.

Computing the importance sampling distribution from the loss with respect to each training example was proposed in [16]. Since the target loss function to be minimized is the aggregate of the losses across all training examples, intuitively, the examples with the largest losses require more training iterations. It is, however, impractical to compute gradients on a per-sample basis for federated learning as the number of communication rounds needed to exchange model updates can scale with the number of training examples per client. Instead, we treated the data at each client as a batch and considered the weighted average training loss across a clients' dataset $|\mathcal{D}_k|F_k$ as a proxy to indicate a client's priority. Here, $|\mathcal{D}_k|$ is the size of the training set and F_k is the average loss across the $\mathcal{D}_k$ examples. The weighting allows proportionally weighing the losses based on the number of training examples at the clients.

Such an approach, by itself, cannot reduce the wall clock training time as the compute and communication times can vary significantly between the clients. If the clients' upload time T_k^{up} can be estimated, representing the aggregate of the compute time T_k^{comp} and communication time T_k^{comm}, this will allow computing importance sampling probabilities accounting for not only the harder to train clients, but also the clients with short upload times. One such approach is to normalize the weighted loss by the upload time to compute the importance sampling distribution $\mathbf{s}$ where $s_k \propto |\mathcal{D}_k|F_k/T_k^{up}$ represents the sampling probability for client k. The clients are sampled based on the importance sampling distribution s. At the end of each training round, $\mathbf{s}$ is re-computed based on the updated quantities.

As noted earlier, in FL, the number of local gradient steps τ can be greater than one. As the clients are sampled from a different distribution s instead of the original distribution $\mathbf{p}$, gradient correction is applied at each local step to un-bias the resulting gradient. Specifically, the gradient computed at each step at client k is multiplied by the likelihood ratio p_k/s_k before the next local gradient descent step. The final updated local model computed after τ local steps is shared with the edge server. The server aggregates the local updates from the K clients that were sampled to obtain the global model for the next epoch. The proposed federated learning with importance sampling (FedIS) algorithm is described in Figure 2, and more details can be found in [17]. While the computation of the importance sampling distribution requires additional computational and communication overhead, our extensive empirical results demonstrate that the proposed method converges faster than the baseline approaches in spite of the overheads.

Fairness in federated learning is crucial where the goal of fair federated learning over wireless networks is to minimize the worst-case loss of clients while satisfying the compute and communication time constraints. We note that the proposed federated learning with importance sampling approach assigns a higher sampling probability to clients with larger weighted loss. Due to the adaptive nature of the importance sampling approach, clients with larger weighted loss in each global epoch obtain priority in the FL training. This also led to a higher degree of fairness for the proposed approach.

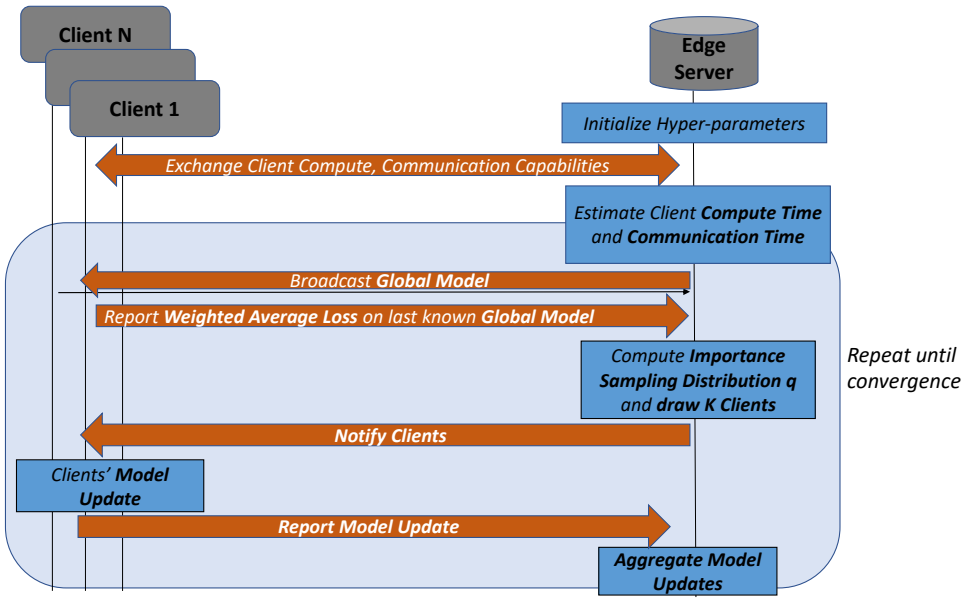

Figure 2. Federated learning (FL) with importance sampling.

3.3. Model Personalization and Task Generalization in Federated Learning

In federated learning, the goal is to find a global model that "on-average" minimizes the weighted loss across clients. Although several approaches have been proposed to address the non-IID issue, it is important to note the fundamental limitation of treating the federated learning problem as the minimization of the average local losses. This is due to the following reasons:

1. Distance of client distributions: When client data are highly non-IID, learning a single average model may achieve sub-optimal personalized performance for individual clients.
2. Ability to fine-tune: The performance of the resulting model by fine-tuning the global model is also not necessarily optimal.
3. Clustered client behavior: While clients have varying data distributions, it is possible to find cluster regions where clients have comparable data distributions. However, current approaches are agnostic of such cluster formations and are unable to take advantage of the client clusters to improve the training efficiency or provision personalization, especially under resource constraints.

In meta-learning, instead of training a single model on a variety of tasks, the goal is to instead train a meta-initializer over the task distribution. The meta-initializer is trained such that when a new task is encountered, a few training steps on a small amount of data from the new task are sufficient to fine-tune the meta-model for maximal performance on the new task. We utilize this important insight for a federated learning scenario under resource constraints.

3.4. Federated Meta-Learning with Client Clustering

Meta-learning can be applied for federated learning across a broad set of clients with heterogeneous datasets. As a result, each task in a federated meta-learning setting can refer to a client or a set of clients that have nearly identical datasets. Therefore, in federated meta-learning, given a set of tasks, the goal is not to find a global model that performs well on-average on the tasks. Instead, the approach relies on learning a meta-initializer such that both existing and new clients can fine-tune the meta-model with only a few training steps and a small amount of data.

While it is true that each client can also fine-tune a model obtained by the federated averaging algorithm, as will be shown later, the federated meta-model acts as a superior initializer compared to the federated averaging model and requires significantly fewer

resources (data points and fine-tuning steps) in order to achieve maximal performance for the clients. This is due to the fundamental formulation of the federated meta-learning problem to obtain the optimal meta-model **w*** as the solution to:

$$\mathbf{w}* = \min_{\mathbf{w}} E_p[F_k(\mathbf{w} - \beta * \nabla_{\mathbf{w}} F_k(\mathbf{w}))], \quad (1)$$

where $E_p[\cdot]$ is the expectation operator with respect to a distribution **p** of clients.

As can be noted above, the optimizer aims to find a meta-model initializer **w*** such that the loss of the fine-tuned model on average is minimized. In other words, the optimizer's goal is such that when clients perform a single gradient descent update using a learning rate β on the meta-model **w***, the resulting average loss across clients is minimized. Federated meta-learning, therefore, is a more powerful goal of achieving model personalization as we are interested in how the resulting meta-model can be tuned for achieving maximal performance for the clients.

We take this further to argue that a resource-efficient meta-model can be trained if we can identify client clusters. Client clusters are not uncommon in the federated learning setting where user groups can be observed, as shown in Figure 3. For example, in learning language models, distinct user groups can be formed based on the users' native languages. We propose to utilize the fact that clients form clusters to perform resource-efficient federated meta-learning.

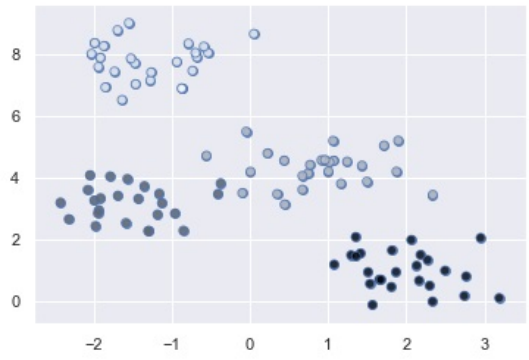

Figure 3. User clusters in a lower dimensional (2D) space.

To this end, we first allow grouping of the clients based on their data distribution. In one approach, each client can share a distribution over its training examples or labels. In a supervised learning setting, in order to reduce the dimensionality of this communication, the clients may simply send the probability mass function of their data labels y_k. Alternatively, clients can transform (linear or non-linear) their data to a different dimension and share the distribution over their examples with the server. Given a set of N clients, where each client k has a unique data distribution q_k, clusters of clients are determined based on the similarity of their distributions.

The edge server can apply a clustering algorithm based on the received vectors from the clients in order to determine distinct client clusters/tasks, as shown in Figure 3. Once clients are assigned to clusters, we apply a federated meta-learning algorithm such that we train only on a subset of the clients to obtain the meta-initializer subject to the communication and compute resource constraints. The resulting meta-initializer model is, in turn, utilized by clients to fine-tune on a small set of training examples using a few training steps. The federated meta-learning with clustering is described in Algorithm 1.

Algorithm 1 Federated meta-learning with clustering. S: MECserver, C: clients.

S: Initialize N, K, τ, η, initial model $\mathbf{w}_1$, target task distribution $\mathbf{p}$ if available
C: Initialize datasets $\mathcal{D}_k$ and D_k^{test} for gradient and meta-update steps, respectively
C: Report the PMF of local data points (or labels) to the server.
S: Determine cluster groups where each cluster is identified by a distribution q_k.
S: Optional step: prune the client set by selecting x% of clients per cluster for training
for all global epochs $t = 1, \tau, ..., \tau T$ **do**
    S: Broadcast global weight $\mathbf{w}_t$
    S: Draw K clients for meta-model update
    for each client k = 1, 2,..., K **do**
        for each local epochs e = 1, 2,..., τ **do**
            Compute model update on the global model using data $\mathcal{D}_k$ to obtain a local model $\bar{\mathbf{w}}_{t+e}^k$
            Compute its Hessian $\mathbf{h}_k$. The Hessian may be computed on a separate dataset as long as it is IID drawn from q_k
            Evaluate the gradient corresponding to the D_k^{test} using the computed local weights as $\mathbf{g}_k(\bar{\mathbf{w}}_{t+e}^k)$
            Compute the local meta update as $\mathbf{w}_{t+e}^k = \mathbf{w}_{t+e-1} - \alpha(\mathbf{I} - \beta\mathbf{h}_k)\mathbf{g}_k(\bar{\mathbf{w}}_{t+e}^k)$
        end for
        Report $\mathbf{w}_{t+\tau}^k$ to the server
    end for
    S: Compute the global weight at the server as $\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha \frac{1}{\sum_k^K |\mathcal{D}_k|} \sum_k^K |\mathcal{D}_k| \mathbf{w}_{t+\tau}^k$
end for

There are two major advantages of the proposed methods. First, sample-efficient learning can be performed from a subset of clients from each cluster to optimize communication and computational resources (e.g., clients that can speed up the wall clock time). Second, this allows a generalization of the learning of the model over a set of tasks where a task can represent a certain client group of interest. If a distribution over the client groups is known, the federated meta-learning algorithm can learn to generalize over the task distribution.

4. Experimental Evaluation

4.1. Experimental Setup

The performance of the resource management and model personalization approaches was verified through simulations in Python. We simulated a network with N clients and an edge server. Client deployment and data rates were determined using the communication model described in this section. The compute rates of clients were determined based on the statistical compute model we describe in this section. The deep learning models for the clients and server were implemented using TensorFlow libraries in Python. A multi-class image classification task using deep neural networks (DNNs) was considered where the images were distributed across the clients and not shared with the server. For our DNN model, we considered 2 hidden layers where each of them consisted of 200 neurons followed by a ReLU non-linearity activation. A softmax function was applied at the output layer to return the probabilities over C image classes.

The well-known benchmark dataset MNIST, as well as the federated benchmark dataset LEAF [18] were utilized in the evaluation. Under MNIST, the data are distributed

among clients such that each client has only 1 class of images. For the FEMNISTdataset under LEAF, the dataset is unbalanced across clients such that each client's dataset corresponds to a unique user. The FEMNIST dataset is inherently non-IID across clients as each client contains data from a unique user that generated the alphanumeric character. Further, we only considered clients that contained at least 10 training examples in order to avoid scenarios where clients had too few data points. The proposed solutions were compared against FedProx [9] and q-fair FL [11], which addressed the non-IID and fairness issues in addition to the federated averaging algorithm (FedAvg).

4.1.1. Compute and Communication Models

A 5G deployment scenario and channel models as described in the ITU-R M.2412 report were utilized for clients to communicate with their corresponding base stations (BSs). The uplink data rates of clients were obtained according to the specific deployment scenario considered for each client. We considered broadly two scenarios for client deployment: (a) dense urban-eMBB, and (b) indoor hotspot-eMBB. The indoor hotspot-eMBB scenario was further classified into 4 GHz and 30 GHz operating frequencies. For the mmWave operating frequency of 30 GHz, the clients may support either 12 or 30 transmission reception points (TRxPs) where TRxPs determine the number of vertical and horizontal antenna elements within a panel per polarization and the number of polarizations and panels. Both single-user MIMO and multi-MIMO configurations can be supported for clients in all the above scenarios. Clients were randomly assigned a configuration of the deployment scenario, the carrier frequency, the antenna configuration, as well as the user locations. For example, under the indoor hotspot deployment scenario, a client's location was selected uniformly randomly in a 120 m $\times$ 50 m rectangular grid containing 12 BSs. Using this information, the average data rate for each client was obtained as r_k. We utilized a 32 bit representation for the payload D_{size} containing the weight matrices.

The compute time was modeled using a shifted exponential distribution and described in more detail in [17]. The mean of the multiply-and-add (MAC) rates for clients was set to 40% of the peak MAC rate of 384 TMACs. The rates for other clients were calculated using the shifted exponential distribution as $\mu_k = 0.9^k * \mu_0$. The compute distribution contained a deterministic and random component. The deterministic component was shared by clients with the edge server. The random component affected the compute times between different epochs and was utilized only for performance evaluation. The various simulation parameters and training hyper-parameters are described in Table 1.

Table 1. Experimental setup.

Parameter	Value(s)
Total Number of Clients N	100 (MNIST) 105 (LEAF: FEMNIST) 20 (Meta Learning)
Global Training Epochs	2000
Local Epochs τ	5, 1 (meta-learning)
SGD Learning Rate	0.01

4.2. Federated Learning with Importance Sampling

Two variants of the federated learning with importance sampling (FedIS) algorithm were evaluated, and the results are presented in Table 2. In FedIS I, the importance sampling probability s_k of client k was calculated proportional to its weighted training loss $|\mathcal{D}_k|F_k(\mathbf{w})$. For FedIS II, the sampling probability was calculated as $s_k \propto |\mathcal{D}_k|F_k/T_k^{up}$. For this experiment, we simulated $N = 100$ clients where each client's compute rates and communication rates were determined based on the models described in Section 4.1.1. This resulted in significant heterogeneity among clients' upload times.

The FedIS II approach adaptively computes the importance sampling distribution utilizing the weighted loss, as well as the communication and compute time of clients. Therefore, it achieves about a 4.5× reduction in the wall clock training time. The model performance of FedIS II as measured by the test accuracy/loss either exceeded or was comparable to the FedAvg and FedProx baselines on both MNIST and LEAF datasets. Two factors played a key role in this performance gain. While the upload time considered in FedIS II allowed clients with better wireless links and compute rates to gain a higher priority during training, the weighted loss component of FedIS ensured that under-trained clients were given higher priority. To validate this, we also evaluated FedIS I where the importance sampling distribution was constructed only from the weighted loss. FedIS I improved only the overall training performance without affecting the wall clock time.

Table 2. Performance evaluation of federated learning with importance sampling (FedIS).

Algorithm	Accuracy (Mean)	Accuracy (Variance)	Accuracy (10th %ile)	Total Time (h)
MNIST, N = 100, K = 10, τ = 5				
FedAvg	90.69%	0.005	79.83%	12.75
FedProx	90.42%	0.006	78.96%	12.75
q = 2 fair	91.24%	0.002	81.16%	13.12
FedIS I	**97.47%**	**0.001**	**91.16%**	12.4
FedIS II	96.32%	**0.001**	90.78%	**2.84**
LEAF, N = 105, K = 10, τ = 5				
FedAvg	78.33%	0.034	51.81%	8.19
FedProx	78.96%	0.031	**58.83%**	8.19
q = 2 fair	76.82%	**0.027**	**58.83%**	8.40
FedIS I	**78.97%**	0.030	56.06%	7.84
FedIS II	77.34%	0.035	50.0%	**2.09**

4.3. Federated Meta-Learning with Client Clustering

For federated meta-learning with clustering (CFedMeta), the clients are clustered using affinity propagation, which utilizes the KL divergence between client distributions to compute client similarities. One of the advantages of utilizing the affinity propagation approach for clustering is that the number of clusters in the system does not need to be pre-defined. At the beginning of the training, each client shares the probability mass function of its data (label) distribution over the different classes based on its training examples. The server utilizes the distributions to determine both the number and the size of the clusters. Once client clusters are identified, we apply our resource-efficient client sampling for federated meta-learning in order to obtain the meta-initializer. This is achieved by pruning a fixed fraction of clients from participating in training rounds. In our experiments, we sorted the clients based on their estimated upload time T_k^{up} and pruned the slowest fraction of clients from participating in the training rounds. The communication rates and compute rates of clients were obtained based on the models described in Section 4.1.1.

The trained meta-initializer from the proposed approaches was compared against a federated averaging based model, as well as a meta-initializer that utilized the entire client set for training. The FEMNIST dataset of LEAF was utilized for performance comparison. The training consisted of N = 20 clients each containing at least 50 training examples. Ten percent of dataset was set aside for the meta-update step in Algorithm 1. Once the global model initializer was learned, each client downloaded the global model and fine-tuned the model utilizing 5 stochastic gradient descent (SGD) steps to obtain their personalized models. The personalization performance was evaluated using the clients' test

data. The number of clients dropped for the clustering approach was varied from 25–75%, and the performance tradeoff between training time and model accuracy was observed.

These results are presented in Table 3. The FedAvg method had a lower-bound of 59.7% on the performance measured by the average test accuracy across the N clients. When each client was allowed to perform five local update steps on the global model (FedAvg*), the average accuracy improved marginally. The FedMeta* approach, which utilized the entire client set for training the meta-initializer, clearly showed an overall improved the test accuracy measured by the mean and variance. However, it led to a marginally increased total training time. With task similarity based client clustering, the CFedMeta approach allowed dropping a different fraction of clients to achieve a tradeoff between the model performance and the training time. By dropping 25% of the clients with the largest upload time during training, the training time was reduced by about 1.5× without affecting the generalization performance measured by the variance of accuracy. Further, up to a 3× reduction in training time while achieving comparable model accuracy to FedAvg* was observed. Overall, the clustering approach presented a range of tradeoffs for applying federated meta-learning under different resource constraints while consistently outperforming a FedAvg based model initializer in the overall test accuracy.

Table 3. Clustering based federated meta-learning (FedMeta) approach. CFedMeta, federated meta-learning with clustering. * indicates 5 rounds of fine-tuning on the global model.

Algorithm	Accuracy (Mean)	Accuracy (Variance)	Total Time (min)
FedAvg	59.7%	0.021	-
FedAvg * (* $->$ 5 steps)	60.0%	0.024	25.1
FedMeta *	**67.2%**	0.018	25.1
CFedMeta * (drop 25%)	63.7%	**0.017**	18.73
CFedMeta * (drop 50%)	62.3%	0.024	12.65
CFedMeta * (drop 75%)	60.0%	0.030	**8.71**

5. Conclusions

As the adoption of federated learning, especially over wireless networks, is accelerating, it is important to address the key challenges in order to realize the full potential of this technology. These challenges include resource management and model personalization, as well as the increased burden of this emerging workload on the wireless infrastructure. In this paper, we presented new resource management strategies, as well as model personalization approaches that can benefit not only the training performance in the presence of diverse sources of heterogeneities, but also help achieve a timely learning and efficient utilization of the wireless resources available for training. Empirical results show that an importance sampling approach to federated learning can achieve a range of tradeoffs from optimizing the model accuracy to reducing the training time significantly. Further, our novel approach to provide a resource-efficient model personalization allows a resource-efficient way to train a model initializer in the presence of a range of client data distributions in the system. The resulting model initializer is shown to improve clients' local test accuracy after only a few steps of fine-tuning. We note that model personalization is becoming an increasingly important aspect of federated learning since it is critical to have highly accurate on-device models. In this context, we aim to investigate the extension of our importance sampling approach to further enhance the efficiency of the model personalization method in our future work.

Author Contributions: Conceptualization, R.B., M.A., S.D., A.A., and N.H.; methodology, R.B.; software, R.B. and M.A.; formal analysis, R.B.; writing—original draft preparation, R.B., A.A., N.H., and A.Z.; writing—review and editing, R.B., S.D., A.A., N.H., and A.Z.; supervision, N.H. All authors read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not Applicable.

Informed Consent Statement: Not Applicable.

Data Availability Statement: Not Applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Reinsel, D.; Gantz, J.; Rydning, J. The Digitization of the World From Edge to Core. In *IDC Data Age 2025 Whitepaper*; IDC: Framingham, MA, USA, 2018.
2. Gill, B.; Rao, S. *Technology Insight: Edge Computing in Support of the Internet of Things*; Gartner Research Report; Gartner Inc.: Stamford, CT, USA, 2017.
3. McMahan, H.B.; Moore, E.; Ramage, D.; Hampson, S.; Arcas, B.A. Communication-efficient learning of deep networks from decentralized data. In Proceedings of the Conference on Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA, 20–22 April 2017.
4. Sheller, M.J.; Edwards, B.; Reina, G.A.; Martin, J.; Pati, S.; Kotrotsou, A.; Milchenko, M.; Xu, W.; Marcus, D.; Colen, R.R.; et al. Federated learning in medicine: Facilitating multi-institutional collaborations without sharing patient data. *Sci. Rep.* **2020**, *10*, 12598. [CrossRef] [PubMed]
5. Nishio, T.; Yonetani, R. Client Selection for Federated Learning with Heterogeneous Resources in Mobile Edge. In Proceedings of the 53rd IEEE International Conference on Communications (ICC), Shanghai, China, 20–24 May 2019.
6. Yang, H.H.; Liu, Z.; Quek, T.Q.S.; Poor, H.V. Scheduling Policies for Federated Learning in Wireless Networks. *arXiv* **2019**, arXiv:1908.06287.
7. Chen, M.; Yang, Z.; Saad, W.; Yin, C.; Poor, H.V.; Cui, S. A Joint Learning and Communications Framework for Federated Learning over Wireless Networks. *arXiv* **2019**, arXiv:1909.07972.
8. Dinh, C.T.; Tran, N.H.; Nguyen, M.N.H.; Hong, C.S.; Bao, W.; Zomaya, A.Y.; Gramoli, V. Federated Learning over Wireless Networks: Convergence Analysis and Resource Allocation. *arXiv* **2019**, arXiv:1910.13067.
9. Li, T.; Sahu, A.K.; Zaheer, M.; Maziar, S.; Talwalkar, A.; Smith, V. Federated Optimization for Heterogeneous Networks. In Proceedings of the Conference on Machine Learning and Systems (MLSys), Austin, TX, USA, 4 March 2020.
10. Dhakal, S.; Prakash, S.; Yona, Y.; Talwar, S.; Himayat, N. Coded Federated Learning. IEEE GLOBECOM Workshop. *arXiv* **2019**, arXiv:2002.09574.
11. Li, T.; Sanjabi, M.; Smith, V. Fair Resource Allocation in Federated Learning. In Proceedings of the International Conference on Learning Representations (ICLR), Virtual Conference, 26 Apri–1 May 2020.
12. Finn, C.; Abbeel, P.; Levine, S. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In Proceedings of the 34th International Conference on Machine Learning (ICML), Sydney, Australia, 6–11 August 2017.
13. Fallah, A.; Mokhtari, A.; Ozdaglar, A. Personalized Federated Learning: A Meta-Learning Approach. *arXiv* **2020**, arXiv:2002.07948.
14. Kairouz, P.; McMahan, H.B.; Avent, B.; Bellet, A.; Bennis, M.; Bhagoji, A.N.; Bonawitz, K.; Charles, Z.; Cormode, G.; Cummings, R. Advances and Open Problems in Federated Learning. *arXiv* **2019**, arXiv:1912.04977.
15. Alain, G.; Lamb, A.; Sankar, C.; Courville, A.; Bengio, Y. Variance Reduction in SGD by Distributed Importance Sampling. *arXiv* **2016**, arXiv:1511.06481.
16. Loshchilov, I.; Hutter, F. Online Batch Selection for Faster Training of Neural Networks. In Proceedings of the 4th International Conference on Learning Representations, ICLR, San Juan, Puerto Rico, 2–4 May 2016.
17. Balakrishnan, R.; Akdeniz, M.; Dhakal, S.; Himayat, N. Resource Management and Fairness for Federated Learning over Wireless Edge Networks. In Proceedings of the IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC), Atlanta, GA, USA, 26–29 May 2020.
18. Caldas, S.; Duddu, S.M.K.; Wu, P.; Li, T.; Konečný, J.; McMahan, H.B.; Smith, V.; Talwalkar, A. LEAF: A Benchmark for Federated Settings. *arXiv* **2019**, arXiv:1812.01097.