WHALE-FL: Wireless and Heterogeneity Aware Latency Efficient Federated Learning over Mobile Devices via Adaptive Subnetwork Scheduling

Huai-an Su[1*], Jiaxiang Geng[2*], Liang Li[3], Xiaoqi Qin[2], Yanzhao Hou[2], Hao Wang[4], Xin Fu[1], Miao Pan[1]

[1]Department of Electrical and Computer Engineering, University of Houston
[2]School of Information and Communication Engineering, Beijing University of Posts and Telecommunications
[3]Frontier Research Center, Peng Cheng Laboratory
[4] Department of Electrical and Computer Engineering, Stevens Institute of Technology
{hsu3, xfu8, mpan2}@uh.edu, {lelegjx, xiaoqiqin, houyanzhao}@bupt.edu.cn, lil03@pcl.ac.cn, hwang9@stevens.edu

Abstract

As a popular distributed learning paradigm, federated learning (FL) over mobile devices fosters numerous applications, while their practical deployment is hindered by participating devices' computing and communication heterogeneity. Some pioneering research efforts proposed to extract subnetworks from the global model, and assign as large a subnetwork as possible to the device for local training based on its full computing capacity. Although such fixed size subnetwork assignment enables FL training over heterogeneous mobile devices, it is unaware of (i) the dynamic changes of devices' communication and computing conditions and (ii) FL training progress and its dynamic requirements of local training contributions, both of which may cause very long FL training delay. Motivated by those dynamics, in this paper, we develop a wireless and heterogeneity aware latency efficient FL (WHALE-FL) approach to accelerate FL training through adaptive subnetwork scheduling. Instead of sticking to the fixed size subnetwork, WHALE-FL introduces a novel subnetwork selection utility function to capture device and FL training dynamics, and guides the mobile device to adaptively select the subnetwork size for local training based on (a) its computing and communication capacity, (b) its dynamic computing and/or communication conditions, and (c) FL training status and its corresponding requirements for local training contributions. Our evaluation shows that, compared with peer designs, WHALE-FL effectively accelerates FL training without sacrificing learning accuracy.

Introduction

Federated Learning (FL) (McMahan et al. 2017) recently experienced a notable evolution, expanding its scope from conventional data center environments to harness the potential of mobile devices (Li et al. 2021a; Chen et al. 2023). This shift has been propelled by the continuous advancements in hardware, empowering mobile devices like the NVIDIA Xavier, iPhone 16, etc. with increasingly robust on-device computing capabilities tailored for local training. With the collective intelligence of edge devices and FL's fundamental principle of preserving data privacy, FL over mobile devices has paved the way for a diverse spectrum of innovative mobile applications, including keyboard predictions (Hard et al. 2018), smart home hazard detection (Yu et al. 2020), health event detection (Brisimi et al. 2018), and so on.

While FL over mobile device has great potentials, its practical deployment faces significant challenges due to the inherent heterogeneity among real-world mobile devices, varying in computing capability, wireless conditions and local data distribution (Lai et al. 2021). Existing FL studies often assume the model-homogeneous setting, where global and local models share identical architectures across all clients. However, as devices are forced to train models within their individual capability, developers have to choose between excluding low-tier devices, introducing training bias (Bickel, Hammel, and O'Connell 1975), or maintaining a low-complexity global model to accommodate all clients, resulting in degraded accuracy (Cho, Wang, and Joshi 2021; Ye et al. 2020). The trend towards large models like Transformers (Liu et al. 2023) exacerbates the issue, hindering their training on mobile devices. Furthermore, unlike GPU clusters with stable high-speed Internet connections, mobile devices' computing resources are constrained and heterogeneous, and their wireless transmissions are relatively slow and dynamic, both of which lead to huge latency in FL training (Chen et al. 2022) and may severely degrade the performance of associated applications.

To address the limitations of model-homogeneous FL, researchers have recently studied how to train different sized models across heterogeneous mobile clients and corresponding global model aggregation in FL training. Subnetwork training, exemplified by pioneering approaches like width-based subnetwork generation in Federated Dropout (Wen, Jeon, and Huang 2021) and HeteroFL (Diao, Ding, and Tarokh 2021), and depth-based generation in DepthFL (Kim et al. 2023), has proven effective by enabling mobile devices to train smaller subnetworks derived from the large global server model. These designs also offer solutions to aggregating diverse devices' subnetworks. By tailoring subnetwork architecture for the individual device, subnetwork training can ensure compatibility with mobile devices owning heterogeneous computing and communication capabilities. However, a prevalent challenge in current subnetwork approaches lies in their static fixed-size subnetwork assignment policy. Such a policy may fail to unleash the full potential of subnetwork based training, mainly due to the un-

*These authors contributed equally.

awareness of system dynamics (i.e., computing and communications dynamics) and FL training dynamics.

System dynamics encompass the time-varying computing loads of devices' background applications and the fluctuating wireless communication conditions across FL training rounds, which affects the sizes of subnetworks that a mobile device can support over rounds. Since most modern mobile devices (e.g., smartphones) participating in FL training have the ability to run multiple tasks simultaneously (Banabilah et al. 2022), the dynamic orchestration of CPU/GPU resources across these concurrent activities results in the fluctuations in computing power and available memory for FL tasks, consequently impacting the supported subnetwork sizes for on-device computing. Similarly, wireless communications dynamics caused by users' mobility, wireless channel fading, etc. lead to dynamic transmission rates, which directly affect candidate subnework sizes that a mobile device can support for local model updates.

FL training dynamics represents FL convergence's dynamic requirements for the contributions from local training at different training stages, which implicitly affects participating devices' selections on subnetwork sizes. Recent studies have revealed that critical learning periods (CLP) exist in the training process of deep neural networks (Achille, Rovere, and Soatto 2018; Yan, Wang, and Li 2022). As the FL training proceeds, the training contributions from each client gradually decrease. Thus, it is crucial to assign suitable subnetwork sizes to clients based on FL training dynamics.

We observe that failing to capture system or training dynamics and always using the possible largest-sized subnetworks under devices' full capabilities may significantly prolong the FL training process. Different from prior static fixed-size subnetwork assignment methods, in this paper, we propose a wireless and heterogeneity aware latency efficient FL (WHALE-FL) approach to accelerate FL training via adaptive width-wise subnetwork scheduling. WHALE-FL characterizes system dynamics and FL training dynamics and tailors appropriate-sized subnetworks for heterogeneous mobile devices under dynamic computing/wireless environments at different FL training stages. As far as we know, WHALE-FL is the first paper that converts static fixed-size subnetwork allocation, e.g., HeteroFL (Diao, Ding, and Tarokh 2021), Federated Dropout (Wen, Jeon, and Huang 2021), etc., into adaptive subnetwork scheduling for each device by jointly considering system heterogeneity and FL training dynamics, and conducts system-level experiments for validation. Our salient contributions are summarized as follows:

- We design a novel subnetwork selection utility function to capture system and FL training dynamics, guiding mobile devices to adaptively size their subnetworks for local training based on the time-varying computing/communication capacity and FL training status.
- We develop a WHALE-FL prototype and evaluate its performance with extensive experiments. The experimental results validate that WHALE-FL can remarkably reduce the latency for FL training over heterogeneous mobile devices without sacrificing learning accuracy.

Preliminary

FL over Heterogeneous Mobile Devices

Consider that M mobile devices in a wireless network collaboratively engage in FL to train a deep neural network on locally distributed datasets $\{L_1, \cdots, L_i, \cdots, L_M\}$. Their local models are parameterized by $\{W_1, \cdots, W_i, \cdots, W_M\}$, which are updated using stochastic gradient descents (Ruder 2016) on the local data samples through local training. The server collects the local model updates and aggregates them into a global model W_g using model averaging (McMahan et al. 2017; Li et al. 2020). This aggregation occurs over multiple communication rounds, with the global model at the r-th round denoted as $W_g^r = \frac{1}{M}\sum_{m=1}^{M} W_m^r$. In the subsequent training round, W_g^r is transmitted to mobile devices, and their local models are updated as $W_i^{r+1} = W_g^r$. This process repeats until FL converges, while system heterogeneity (communications and computing) among mobile devices incurs huge training latency and significantly slows down FL convergence.

FL with Subnetwork Extraction

To address the system heterogeneity issue in FL training, the subnetwork training method was introduced in (Diao, Ding, and Tarokh 2021), which extracts subnetworks in different sizes from the global model.

Let $\mathcal{W}^P = \{W^1, W^2, \cdots, W^p, \cdots, W^P\}$ be a collection of candidate subnetworks to be selected by mobile devices for local training, where P complexity/size levels are considered. A lower size level p corresponds to a larger-sized subnetwork, and W^P is the smallest subnetwork for selection, i.e., $W^P \subset W^{P-1} \subset \cdots \subset W^1$. We follow the same approach as illustrated in (Diao, Ding, and Tarokh 2021) to extract subnetworks from the global model by shrinking the width of hidden channel with specific ratios. Let $s \in (0, 1]$ be the hidden channel shrinkage ratio. Then, we have $|W^p|/|W_g| = |W^p|/|W^1| = s^{2(p-1)}$. With this construction, different sized subnetworks can be assigned to participating mobile devices according to their corresponding capabilities. Suppose that the number of devices in each subnetwork size level is $\{M_1, \cdots, M_P\}$. The server has to aggregate the heterogeneous subnetworks in every training round. As demonstrated in (Diao, Ding, and Tarokh 2021), the global aggregation is conducted as follows.

$$\begin{aligned} W_g = W_g^1 &= W_g^P \cup (W_g^{P-1} \backslash W_g^P) \cup \cdots \cup (W_g^1 \backslash W_g^2) \\ &= W_g^P \cup \bigcup_{p=2}^{P} W_g^{p-1} \setminus W_g^p, \end{aligned} \tag{1}$$

where

$$W_g^P = \frac{1}{M}\sum_{m=1}^{M} W_m^P,$$

$$W_g^{p-1} \backslash W_g^p = \frac{1}{M - M_{p:P}} \sum_{m=1}^{M-M_{p:P}} W_m^{p-1} \backslash W_m^p, \forall p \in [2, P].$$

In this way, each parameter is averaged from those devices whose assigned subnetwork contains that specific parameter, which enables the global aggregation and FL training

with different sizes of subnetworks. Although the subnetwork method in (Diao, Ding, and Tarokh 2021) alleviates the system heterogeneity issue, it is a fixed policy. It cannot capture the dynamic changes of wireless transmission/on-device computing conditions, or the dynamic requirements of contributions from local training at different FL training stages, either of which may result in a huge training latency.

Fisher Information

Fisher information is utilized as a measurement of how much a change in weights can affect the output of neural networks (Achille, Rovere, and Soatto 2018). Fisher information is a 2nd-order approximation of the Hessian of the loss function (Amari and Nagaoka 2000; Martens 2014), providing information on the curvature of the loss landscape near the current weights. Such characteristics help to indicate how fast the gradient changes during training, which may be used to characterize the training dynamics from local device side and further help clients decide how to adjust their subnetwork sizes.

To enable distributed subnetwork scheduling, we use the Federated Fisher Information Matrix (FedFIM) from (Yan, Wang, and Li 2022) instead of the traditional definition of the Fisher Information Matrix (FIM) for centralized training to avoid requiring access to the entire dataset. That is, given that training data resides in each client, the local FIM on client i in the r-th training round is calculated by

$$FI_{i,r} = \mathbb{E}_{x_i \sim \mathcal{X}_i} \mathbb{E}_{\hat{y} \sim p_W(\hat{y}_i|x_i)} [\bigtriangledown(x_i, \hat{y}_i) \bigtriangledown (x_i, \hat{y}_i)^\intercal], \quad (3)$$

where x_i is the input data of and y_i is the corresponding output label of client i, W is the weight and $p_W(\hat{y}_i|x_i)$ is the approximate posterior distribution. $\mathcal{X}_i$ is the empirical distribution of the i-th client's local data. The corresponding gradient of the loss for (x, y) is denoted as $\bigtriangledown(x, y) = \frac{\partial}{\partial W}\ell(x, y; W)$, and $\hat{y}_i$ is a random variable rather than a true label with its distribution following $p_W(\hat{y}_i|x_i)$.

Motivation

Unawareness of system dynamics. Traditional subnetwork assignment (e.g., HeteroFL (Diao, Ding, and Tarokh 2021)) is fixed, which is based on the participating mobile device's maximum system capability (i.e., computing + communications), while ignoring the dynamic changes of the device's computing and communication conditions. Such an unawareness may lead to poor subnetwork assignment decisions and significantly delay the FL training process. For instance, a mobile device capable of computing a full-sized model may be experiencing poor wireless access (e.g., 4G/LTE) or running some computing-intensive background applications (e.g., GPU-intensive gaming) in a certain training round. In this case, the fixed full-sized subnetwork assignment may make this device a straggler and cause a big latency in FL training. Thus, *an adaptive subnetwork scheduling aware of system (computing + communication) dynamics is in need.*

Unawareness of FL training dynamics. The fixed subnetwork assignment is unaware of FL training progress and its dynamic requirements of learning contributions from local mobile devices. Since FL training starts from scratch, any contributions from any device's local training will be helpful. Using small-sized subnetworks can expedite on-device computing and wireless transmissions of local model updates. As FL training proceeds into the CLP, more accurate local model updates are needed for the global training model to converge. When FL training is close to convergence (i.e., the late stage), most mobile devices have already made substantial contributions to the global model. For those devices, sticking to large or full-sized subnetworks for local training offers limited learning benefits for FL convergence, while some computing/communications-constrained devices may incur significant training latency or even become stragglers. Therefore, it is necessary to develop an adaptive subnetwork scheduling method that captures FL training dynamics, recognizes computing/communication constraints, and selects appropriately sized subnetworks for local training, to improve delay efficiency in FL training over mobile devices.

WHALE-FL Design

Aiming to reduce FL training latency, WHALE-FL entitles mobile devices to distributedly schedule different sizes of subnetworks for local training, adapting to their system dynamics and FL training dynamics. To capture those dynamics, WHALE-FL presents a novel adaptive subnetwork selection utility function jointly considering system efficiency and FL training efficiency. Moreover, WHALE-FL provides a normalization procedure to convert the calculated subnetwork selection utility values to discrete size levels of subnetowrks for mobile devices' local scheduling decisions.

Adaptive Subnetwork Selection Utility

WHALE-FL's adaptive subnetwork selection performance hinges on two critical aspects: system efficiency and training efficiency. System efficiency encompasses the duration of each training round, including local computing and model uploading time consumption. Training efficiency gauges the local training's contributions to global convergence. The fluctuating wireless conditions and available computing resources of devices, as well as their training progress with local data, collectively determine the system and training efficiency, forming what we term as *adaptive subnetwork selection utility*.

To accelerate FL training without sacrificing learning accuracy, it is critical to trade-off system and training efficiencies to select the appropriate subnetwork size for individual device's local training per round. Briefly, WHALE-FL favors system efficiency over training efficiency at the early stage of FL training, and tends to schedule small-sized subnetworks for devices' local training. While FL training steps into the middle stage, if more accurate local training is needed for FL convergence, WHALE-FL prefers training efficiency to system efficiency and schedules to adaptively increase the size of subnetworks for participating mobile devices. Otherwise, WHALE-FL prioritizes system efficiency over training efficiency. When FL is close to convergence, WHALE-FL jointly considers system and training efficiencies, and gradually decreases the size of subnetworks for local training, given the fact that most devices have contributed

enough to the global model and it is unnecessary to keep large-sized subnetworks for local training.

System efficiency utility. We define the system efficiency ($SE_{i,r}$) for any given client i in the r-th round based on its wireless transmission rate and available computing resources at that time, which is calculated as follows:

$$SE_{i,r} = \frac{T}{T^{tr}_{i,r} + T^{co}_{i,r}}, \tag{4}$$

where $T^{tr}_{i,r}$ and $T^{co}_{i,r}$ are the transmission delay and the computing delay, respectively, for the unit/smallest subnetwork. T is the developer-preferred duration of each round, which may vary for different FL tasks. We assume that the wireless transmission rates and available computing resources dynamically change over rounds, but are relatively stable within a FL training round. Thus, given a learning task, transmission and computing workloads for the unit subnetwork are fixed, and $T^{tr}_{i,r}$ and $T^{co}_{i,r}$ can be easily estimated for device i in the r-th round. A higher $SE_{i,r}$ enables devices to opt for larger subnetwork sizes for local training within this round, and vice versa. The formulation in Eqn. (4) comprehensively covers the system efficiency for communication delay dominant cases (i.e., slow transmissions & fast computing), computing delay dominant cases (i.e., fast transmissions & slow computing), and communication-computing comparable cases.

Training efficiency utility. By employing fisher information FI, we define the training efficiency utility $TE_{i,r}$ for device i in the r-th round as follows:

$$TE_{i,r} = |\mathbf{B}_i| \sqrt{\frac{1}{|\mathbf{B}_i|} \sum_{k \in \mathbf{B}_i} \sum_{d \in \mathbf{D}} \frac{FI_{i,r-d}(k)^2}{|\mathbf{D}|}}, \tag{5}$$

where B_i represents the batched datasets for device i. Here, we utilize a window-averaged local Fisher information to measure the dynamic utility during training with $\mathbf{D} = \{1, .., d, .., D\}$ as the set of window sizes. Here, the sliding window operation helps to prevent frequent zigzag changes in subnetwork sizes, as Fisher information across different local training iterations within the i-th round may be highly unstable, and directly using the Fisher information of each iteration could result in unstable subnetwork selection strategies (Achille, Rovere, and Soatto 2018).

Adaptive subnetwork selection utility function. WHALE-FL trades-off the system and training efficiencies to determine the utility values for subnetwork scheduling over rounds. The adaptive subnetwork selection utility function is shown in Eqn. (6), where $Util(i, r)$ associates system and training efficiencies with developer-specified factor β. Aware of both system and FL training dynamics, a large/small value of $Util(i, r)$ suggests that device i should opt for a large/small sized subnetwork in the subsequent r-th round.

$$Util(i,r) = \underbrace{|\mathbf{B}_i| \sqrt{\frac{1}{|\mathbf{B}_i|} \sum_{k \in \mathbf{B}_i} \sum_{d \in \mathbf{D}} \frac{FI_{i,r-d}(k)^2}{|\mathbf{D}|}}}_{\textit{Training efficiency utility}} \times \underbrace{\left(\frac{T}{T^{tr}_{i,r} + T^{co}_{i,r}}\right)^{\beta}}_{\textit{System efficiency utility}}. \tag{6}$$

Utility Value to Subnetwork Size Conversion

The calculated utility in Eqn. (6) cannot directly be used by individual mobile devices to decide their subnetwork size selection. To facilitate mobile devices' decisions, it is necessary to convert subnetwork selection utility values into available/candidate subnetwork sizes.

Given the definitions above, the next step is to normalize devices' utility values into the range of $[0, 1]$, in order to identify the model shrinkage ratio. We propose to use a piecewise linear function to normalize $Util(i, r)$ into $U_n(i, r)$ as follows.

$$U_n(i,r) = \begin{cases} \frac{Util(i,r)}{U_{th}}, & Util(i,r) \leq U_{th}, \\ 1, & \text{otherwise}, \end{cases} \tag{7}$$

where U_{th} is a configurable threshold that represents the utility level at which the full-sized model should be adopted.

After the utility value normalization, device i selects its subnetwork for the r-th round local training by

$$W(i,r) = \begin{cases} \hat{W}(i,r), & \textbf{if } |W^{max}_i| > |\hat{W}(i,r)|; \\ W^{max}_i, & \textbf{if } |W^{max}_i| \leq |\hat{W}(i,r)|. \end{cases} \tag{8}$$

Here, $|W^{max}_i|$ denotes the maximum subnetwork size that device i can support with its full computing capacity, where $W^{max}_i \in \mathcal{W}^P$ as defined in Sec. . $\hat{W}(i,r) \in \mathcal{W}^P$ is a subnetwork derived from normalized utility value $U_n(i, r)$, which can be expressed as

$$\hat{W}(i,r) = \begin{cases} W^1, & \textbf{if } U_n(i,r) \geq \frac{(P-1)}{P}; \\ W^2, & \textbf{if } U_n(i,r) \in [\frac{(P-2)}{P}, \frac{(P-1)}{P}); \\ \cdots, & \cdots \\ W^p, & \textbf{if } U_n(i,r) \in [\frac{(P-p)}{P}, \frac{(P-p+1)}{P}); \\ \cdots, & \cdots \\ W^P, & \textbf{if } U_n(i,r) < \frac{1}{P}, \end{cases} \tag{9}$$

where $|W^p|/|W_g| = s^{2(p-1)}, \forall W^p \in \mathcal{W}^P$.

Then, mobile devices conduct local computing according to their selected subnetworks, respectively, followed by transmitting local model updates to the FL server. Following the same aggregation method in (Diao, Ding, and Tarokh 2021), the FL server aggregates updated local models with heterogeneous subnetworks and updates the global model as

$$W(g, r+1) = W^{P,r+1}_g \cup \bigcup_{p=2}^{P} W^{p-1,r+1}_g \setminus W^{p,r+1}_g. \tag{10}$$

In summary, during FL training, mobile devices collect their local information at runtime, including up-link channel quality, background computational loads, memory usage, training loss, etc. Based on the collected information, at the beginning of the r-th training round, each device leverages Eqn. (6) to trade-off system efficiency and training efficiency, and calculates its adaptive subnetwork selection utility value $Util(i, r)$. The utility value is then normalized into $U_n(i, r)$. Device i uses $U_n(i, r)$ to determine the subnetwork size and select an appropriate subnetwork for its local training according to Eqn. (8) and Eqn. (9). After that, FL server

aggregates locally trained subnetworks with different sizes and updates the global model for the next round training. In the appendix, we provide the convergence analysis for WHALE-FL based on (Wang et al. 2023), and show that WHALE-FL can converge under adaptive subnetwork size scheduling.

Experimental Setup

WHALE-FL Testbed

The testbed consists of an FL aggregator and a set of heterogeneous mobile devices as FL clients. A NVIDIA RTX 3090 serves as the FL server, whose memory capacity is 24 GB. For heterogeneous FL clients, we have incorporated 5 types of mobile devices, i.e., MacBookPro2018, NVIDIA Jetson Xavier, NVIDIA Jetson TX2, NVDIA Jetson Nano, and Raspberry Pi 4, representing a range of on-device computing capabilities from high to low. The WHALE-FL system involves a total of 20 mobile devices, 4 devices per type. Communication between FL clients and the FL server is facilitated through LTE, BlueTooth, and Wi-Fi 5 transmission environments. The corresponding transmission rates are 80 Mbps (Wi-Fi 5), 20 Mbps (LTE), and 10 Mbps (BlueTooth 3.0), respectively. We set hidden channel shrinkage ratio $s = \frac{1}{2}$ and adopt 5 subnetwork size levels. Accordingly, the model shrinkage ratios for the 5 size levels (i.e., $p = 1, 2, \cdots, 5$) are 1, $\frac{1}{4}$, $\frac{1}{16}$, $\frac{1}{64}$, and $\frac{1}{256}$, respectively.

Datasets, Models, Parameters and Baselines

We conduct our experiments with three different FL tasks: image classification, human activity recognition and language modeling. As for the image classification task, we train a CNN on MNIST dataset (Deng 2012) and a ResNet18 on CIFAR10 dataset (He et al. 2015). Human activity recognition involves training a CNN on the HAR dataset (Gupta et al. 2022), and a Transformer is trained on the WikiText2 dataset (Devlin et al. 2019) for the language modeling task. We use the balanced non-IID data partition (Li et al. 2021b). Take the MNIST dataset as an example, the total number of classes is 10. Our default setup is that each device has $\sigma = 2$ classes. We apply a similar non-IID setup to other tasks. The Fisher information's window size $|\mathbf{D}|$ = 10. We employ the following peer designs for performance evaluation: (i) FedAvg (McMahan et al. 2017), where all the clients train with full-sized models; (ii) HeteroFL (Diao, Ding, and Tarokh 2021), where subnetwork assignments are fixed and aligned with clients' full computation and communication capabilities; (iii) FedDropout (Wen, Jeon, and Huang 2021), which generates subnetworks by choosing the neurons at random; and (iv) FedRolex (Alam et al. 2024), which uses a rolling subnetwork extraction scheme in each FL training round. In particular, we compare the peer design with WHALE-FL's corresponding extension, i.e., the integration of the peer design and WHALE-FL, e.g., FedRolex vs WHALERolex.

Evaluation and Analysis

Latency Efficiency and Learning Performance

As the results shown in Fig. 1, the proposed WHALE-FL consistently achieves remarkable training speedup across various FL tasks without sacrificing learning accuracy. Compared with FedAvg, WHALE-FL accelerates the FL training to the target testing accuracy by approximately 1.5x, 1.9x, 1.3x, and 2.1x for FL tasks including CNN@MNIST, ResNet18@CIFAR10, Transformer@WikiText2, and CNN@HAR, respectively. As detailed in Sec. , HeteroFL's static fixed-size subnetwork assignment policy is not aware of system and training dynamics, which may slow down FL convergence. In contrast, considering both system efficiency and training efficiency, WHALE-FL appropriately assesses the subnetwork selection utility for individual devices and adaptively adjusts the local subnetwork size to suit for time-varying communication and computational conditions and dynamic changing requirements of FL training at different FL training stages, in order to reduce training latency. Consequently, compared with HeteroFL, WHALE-FL achieves a notable speedup of 1.74x, 1.25x, 1.21x and 1.06x for the tested 4 learning tasks, respectively. Results in Tables 1 and 2 further demonstrate that WHALE-FL and WHALE-FL based extensions (i.e., WHALEDropout and WHALERolex) achieve faster convergence and better testing accuracy than the peer designs across different FL tasks.

Subnetwork Size and Fisher Information Changes

As shown in Fig. 2, across the three heterogeneous devices - MacBookPro 2018 (high-end), NVIDIA Jetson TX2 (medium), and Raspberry Pi 4 (low-end) - the subnetwork sizes adjust following the $|\mathbf{D}|$-averaged changes of local Fisher information. The results align with our expectations: When Fisher information is high, the subnetwork size increases to enhance the global model's accuracy; as training proceeds and Fisher information decreases, indicating that its impacts on learning decrease, the subnetwork is becoming smaller to improve the training time efficiency. On the server side, the averaged size of the aggregated local subnetworks changes along with the global model's Fisher information, which exhibits a similar trend to the local Fisher information. Figure 2 demonstrates that WHALE-FL effectively captures training dynamics while selecting appropriate subnetwork sizes for heterogeneous devices.

System Efficiency vs Training Efficiency

To differentiate system efficiency's contributions from training efficiency's ones, we compare WHALE-FL with system efficiency utility only and training efficiency utility only schedulings. As the results shown in Fig. 3, WHALE-FL converges faster than training efficiency only subnetwork scheduling when achieving the target accuracy, since training efficiency only design has no consideration of system dynamics and its impacts on subnetwork size selection; WHALE-FL has better testing accuracy but proceeds slower than system efficiency only subnetwork scheduling at the early training stage. The reason behind is that the system efficiency only design prioritizes system dynamics while ignoring dynamic model accuracy requirements for local training at different FL training stages. WHALE-FL trades-off system and training efficiencies and jointly considers their benefits for FL training.

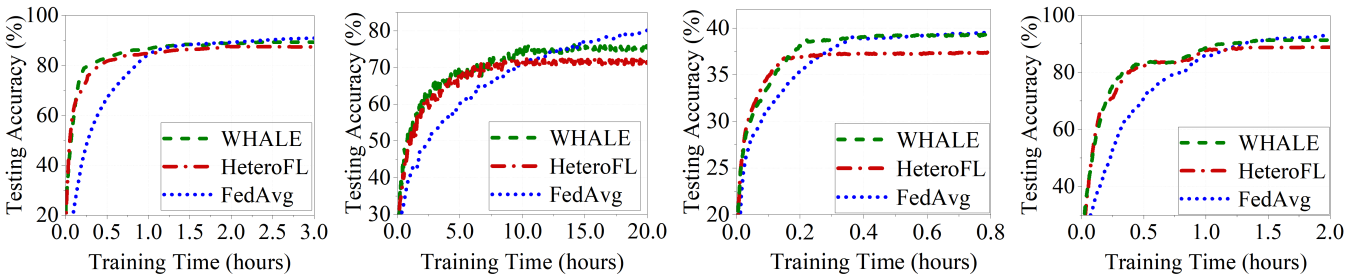


Figure 1: Performance comparison of different FL training approaches under various learning tasks. Figures from left to right are CNN@MNIST, ResNet18@CIFAR10, Transformer@WikiText2, and CNN@HAR with non-IID datasets.

Task	CV		NLP	HAR
	CNN@MNIST	Resnet@CIFAR10	Transformer@Wikitext2	CNN@HAR
Target Accuracy	85%	70%	37%	88%
Method	Speedup			
WHALE vs HeteroFL	1.74x	1.25x	1.21x	1.06x
WHALERolex vs FedRolex	1.75x	1.32x	1.24x	1.10x
WHALEDropout vs FedDropout	1.70x	1.24x	1.20x	1.05x

Table 1: Performance comparison under different subnetwork methods (Speedup).

Task	CV		NLP	HAR
	CNN@MNIST	Resnet@CIFAR10	Transformer@Wikitext2	CNN@HAR
Method	Final Accuracy Improvement			
FedAvg	92.71%	80.61%	40.54%	92.94%
HeteroFL ⇒ WHALE	87.42% ⇒ 89.29%	71.65% ⇒ 75.32%	37.40% ⇒ 39.28%	88.86% ⇒ 91.38%
FedRolex ⇒ WHALERolex	87.82% ⇒ 89.87%	72.52% ⇒ 79.57%	38.02% ⇒ 39.66%	89.11% ⇒ 92.03%
FedDropout ⇒ WHALERolex	86.16% ⇒ 87.52%	70.08% ⇒ 73.45%	37.19% ⇒ 39.06%	88.25% ⇒ 89.55%

Table 2: Performance comparison under different subnetwork methods (Final Accuracy Improvement).

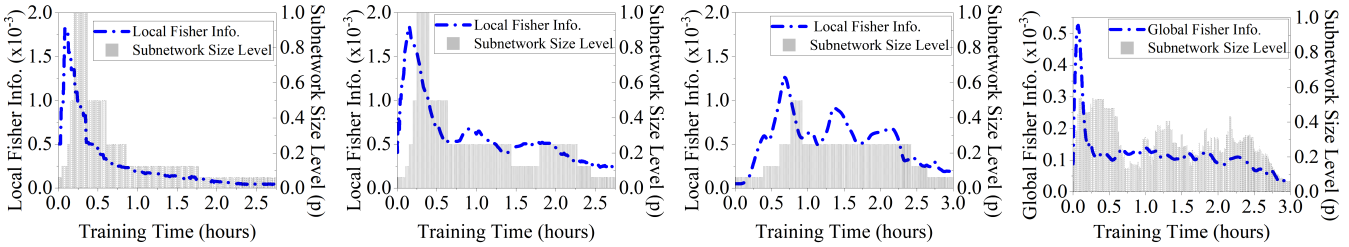


Figure 2: Fisher information and subnetwork size level changes over training time (CNN@MNIST). From left to right, the performance of the user-side models on MacBookPro 2018, NVIDIA Jetson TX2, and Raspberry Pi 4, as well as the global model's performance, are shown.

Sensitivity Analysis

We further evaluate the impacts of β, $D = |\mathbf{D}|$, and U_{th}, defined in the subnetwork selection utility function, on subnetwork scheduling.

The hyperparameter β trades-off system efficiency and training efficiency utilities. The large/small β value means that the device prioritizes system/training efficiency. As the results shown in Fig. 4(a) and Fig. 4(d), we find that the FL training converges slower but achieves higher testing accuracy when β is small, e.g., $\beta = 1$, while FL training is faster at early stages but achieves lower testing accuracy when β is larger, e.g., $\beta = 5$. System efficiency and training efficiency are somehow balanced when $\beta = 2$. Thus, although β is a developer-specified factor, a proper selection of β value helps FL training converge fast while achieving good learning performance. The hyperparameter D represents the window size for calculating the averaged Fisher information. A small window size, such as $D = 1$ in Fig. 4(b) and Fig. 4(e), makes the subnetwork size updates sensitive to changes in Fisher information, leading to fluctuations in model accuracy during training. Conversely, employing a larger window size, like $D = 20$, results in slower subnetwork-size changes. This may cause a situation where a small-sized subnetwork is well-trained while the clients have no chance

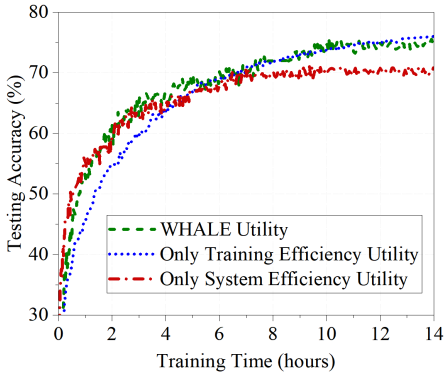
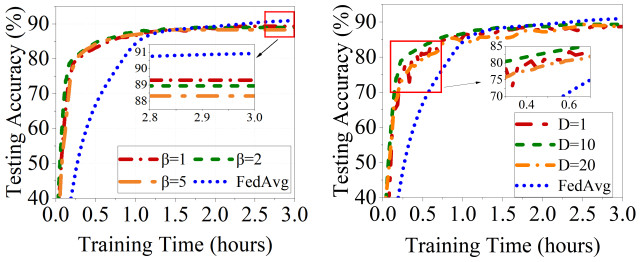
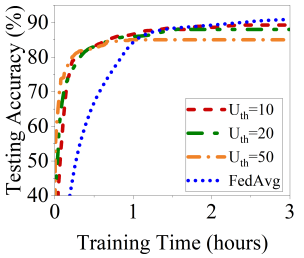


Figure 3: Performance comparison of WHALE-FL, system efficiency only and training efficiency only designs (ResNet18@CIFAR10).

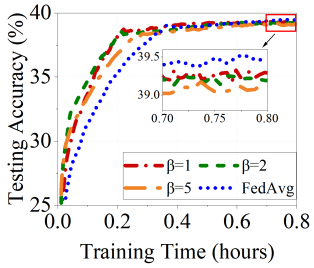


(a) β, CNN@MNIST.

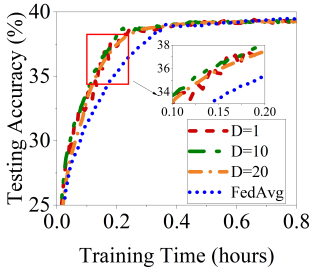
(b) D, CNN@MNIST.



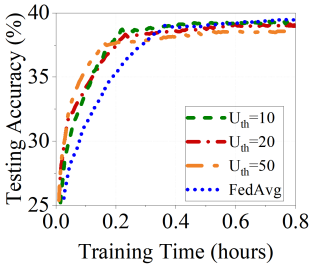
(c) U_{th}, CNN@MNIST.



(d) β, Transf.@WikiText2.



(e) D, Transf.@WikiText2.



(f) U_{th}, Transf.@WikiText2.

Figure 4: Sensitivity analysis under different β, D, and U_{th} values (a-c: CNN@MNIST; d-e: Transformer@WikiText2).

to switch to the larger-sized subnetworks, thus impairing the training performance during the critical learning periods. A window size of $D = 10$ strikes a good balance, achieving faster convergence. Similarly, a higher U_{th}, e.g., U_{th} = 50 shown in Fig. 4(c) and 4(f), leads clients to choose smaller subnetworks, which speeds up FL convergence in the early stages by reducing transmission and computation delays but results in lower final accuracy. Conversely, with U_{th} = 10, clients select larger subnetworks, which slows down convergence but yields higher accuracy. A proper U_{th} selection helps to balance learning performance and delay efficiency.

Local Model	CNN@MNIST		
non-IID Level	$\sigma = 2$	$\sigma = 5$	$\sigma = 10$
Target Acc.	85%	90%	95%
Metric	Hours (SP)		
FedAvg	1.12 (1.00x)	0.33 (1.00x)	0.30 (1.00x)
HeteroFL	1.06 (1.06x)	0.26 (1.27x)	0.10 (3.00x)
WHALE	**0.61 (1.84x)**	**0.19 (1.74x)**	**0.08 (3.75x)**
FedDropout	1.09 (1.03x)	0.28 (1.18x)	0.11 (2.73x)
WHALEDropout	**0.64 (1.75x)**	**0.21 (1.57x)**	**0.08 (3.75x)**
FedRolex	1.03 (1.09x)	0.25 (1.32x)	0.10 (3.00x)
WHALERolex	**0.59 (1.90x)**	**0.18 (1.83x)**	**0.07 (4.29x)**

Table 3: Performance comparison under different data heterogeneity (CNN@MNIST), where "SP" is the speedup.

Impacts of Data Heterogeneity

We further evaluate the impacts of data heterogeneity on WHALE-FL's performance. Here, we take CNN@MNIST as an example and use the balanced non-IID data partition (Li et al. 2021b). The total number of classes in the MNIST dataset is 10. We study the cases where each device has $\sigma = 2, 5$ or 10 classes, where the data distribution is IID if $\sigma = 10$, i.e., every device has all classes. The results are shown in Table 3, where we find that (i) FL training with non-IID data takes longer time to converge, and (ii) embracing both system and training efficiency utilities, WHALE-FL can remarkably improve FL training delay efficiency when applied to existing subnetwork methods under various data heterogeneity scenarios.

Conclusion

In this paper, we have proposed WHALE-FL, a wireless and heterogeneity aware latency efficient federated learning approach, to accelerate FL training over mobile devices via subnetwork scheduling. Unlike existing static fixed-size subnetwork assignments, WHALE-FL has incorporated an adaptive subnetwork scheduling policy, enabling mobile devices to flexibly select subnetwork sizes for local training, with a keen awareness of mobile devices' system dynamics and FL training dynamics. At its core, WHALE-FL has employed a well-designed subnetwork selection utility function, capturing changes in the device's system conditions (including available computing and communication capacities) and evolving FL training requirements for local training, to schedule appropriate subnetworks for mobile devices in each FL training round. Experimental results have demonstrated that WHALE-FL surpasses peer designs, significantly accelerating FL training over heterogeneous mobile devices without sacrificing learning accuracy.

Acknowledgements

The work of L. Li was supported in part by the NSFC 62201071. The work of H. Su and M. Pan was supported in part by the US National Science Foundation under grants CNS-2107057, CNS-2318664, CSR-2403249, and CNS-2431596. The work of X. Fu was partially supported by NSF grants CCF-2130688, and CNS-2107057. The work of X. Qin was supported in part by NSFC 62371072 and Beijing Nova Program 2024102. The work of Y. Hou was supported in part by Beijing NSF L232001 and NSFC U21A20449. The work of H. Wang was supported in part by NSF 2315612, 2403247, and 2431595.

References

Achille, A.; Rovere, M.; and Soatto, S. 2018. Critical Learning Periods in Deep Networks. In *International Conference on Learning Representations*.

Alam, S.; Liu, L.; Yan, M.; and Zhang, M. 2024. FedRolex: model-heterogeneous federated learning with rolling submodel extraction. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc.

Amari, S.; and Nagaoka, H. 2000. Methods of information geometry. In *Tanslations of Mathematical Monographs, 191*.

Banabilah, S.; Aloqaily, M.; Alsayed, E.; Malik, N.; and Jararweh, Y. 2022. Federated learning review: Fundamentals, enabling technologies, and future applications. *Information Processing, Management*, 59(6): 103061.

Bickel, P. J.; Hammel, E. A.; and O'Connell, J. W. 1975. Sex Bias in Graduate Admissions: Data from Berkeley. *Science*, 187(4175): 398–404.

Brisimi, T. S.; Chen, R.; Mela, T.; Olshevsky, A.; Paschalidis, I. C.; and Shi, W. 2018. Federated learning of predictive models from federated Electronic Health Records. *International Journal of Medical Informatics*, 112: 59–67.

Chen, R.; Wan, Q.; Zhang, X.; Qin, X.; Hou, Y.; Wang, D.; Fu, X.; and Pan, M. 2023. EEFL: High-Speed Wireless Communications Inspired Energy Efficient Federated Learning over Mobile Devices. In *Proceedings of the 21st Annual International Conference on Mobile Systems, Applications and Services*, MobiSys '23, 544–556. New York, NY, USA: Association for Computing Machinery. ISBN 9798400701108.

Chen, Y.; Chen, Z.; Wu, P.; and Yu, H. 2022. FedOBD: Opportunistic Block Dropout for Efficiently Training Large-scale Neural Networks through Federated Learning. In *International Joint Conference on Artificial Intelligence*.

Cho, Y. J.; Wang, J.; and Joshi, G. 2021. Client Selection in Federated Learning: Convergence Analysis and Power-of-Choice Selection Strategies.

Deng, L. 2012. The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]. *IEEE Signal Processing Magazine*, 29(6): 141–142.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *North American Chapter of the Association for Computational Linguistics*.

Diao, E.; Ding, J.; and Tarokh, V. 2021. HeteroFL: Computation and Communication Efficient Federated Learning for Heterogeneous Clients. In *International Conference on Learning Representations*.

Gupta, N.; Gupta, S.; Pathak, R.; Jain, V.; Rashidi, P.; and Suri, J. 2022. Human activity recognition in artificial intelligence framework: a narrative review. *Artificial Intelligence Review*, 55.

Hard, A.; Rao, K.; Mathews, R.; Beaufays, F.; Augenstein, S.; Eichner, H.; Kiddon, C.; and Ramage, D. 2018. Federated Learning for Mobile Keyboard Prediction. *ArXiv*, abs/1811.03604.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

Kim, M.; Yu, S.; Kim, S.; and Moon, S.-M. 2023. DepthFL : Depthwise Federated Learning for Heterogeneous Clients. In *The Eleventh International Conference on Learning Representations*.

Lai, F.; Zhu, X.; Madhyastha, H. V.; and Chowdhury, M. 2021. Oort: Efficient Federated Learning via Guided Participant Selection. In *15th USENIX Symposium on Operating Systems Design and Implementation (OSDI 21)*, 19–35. USENIX Association. ISBN 978-1-939133-22-9.

Li, L.; Shi, D.; Hou, R.; Li, H.; Pan, M.; and Han, Z. 2021a. To Talk or to Work: Flexible Communication Compression for Energy Efficient Federated Learning over Heterogeneous Mobile Edge Devices. In *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, 1–10.

Li, Q.; Diao, Y.; Chen, Q.; and He, B. 2021b. Federated Learning on Non-IID Data Silos: An Experimental Study. arXiv:2102.02079.

Li, X.; Huang, K.; Yang, W.; Wang, S.; and Zhang, Z. 2020. On the Convergence of FedAvg on Non-IID Data. In *International Conference on Learning Representations*.

Liu, C.; Qu, X.; Wang, J.; and Xiao, J. 2023. FedET: A Communication-Efficient Federated Class-Incremental Learning Framework Based on Enhanced Transformer. In *International Joint Conference on Artificial Intelligence*.

Martens, J. 2014. New Insights and Perspectives on the Natural Gradient Method. *J. Mach. Learn. Res.*, 21: 146:1–146:76.

McMahan, H. B.; Moore, E.; Ramage, D.; Hampson, S.; and Arcas, B. A. y. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proc. of AISTATS 2017*, 1273–1282.

Ruder, S. 2016. An overview of gradient descent optimization algorithms. *ArXiv*, abs/1609.04747.

Wang, Y.; Zhang, X.; Li, M.; Lan, T.; Chen, H.; Xiong, H.; Cheng, X.; and Yu, D. 2023. Theoretical convergence guaranteed resource-adaptive federated learning with mixed heterogeneity. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2444–2455.

Wen, D.; Jeon, K.-J.; and Huang, K. 2021. Federated Dropout—A Simple Approach for Enabling Federated Learning on Resource Constrained Devices. *IEEE Wireless Communications Letters*, 11: 923–927.

Yan, G.; Wang, H.; and Li, J. 2022. Seizing Critical Learning Periods in Federated Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(8): 8788–8796.

Ye, D.; Yu, R.; Pan, M.; and Han, Z. 2020. Federated Learning in Vehicular Edge Computing: A Selective Model Aggregation Approach. *IEEE Access*, 8: 23920–23935.

Yu, T.; Li, T.; Sun, Y.; Nanda, S.; Smith, V.; Sekar, V.; and Seshan, S. 2020. Learning Context-Aware Policies from Multiple Smart Homes via Federated Multi-Task Learning. In *2020 IEEE/ACM Fifth International Conference on Internet-of-Things Design and Implementation (IoTDI)*, 104–115.