FLHetBench: Benchmarking Device and State Heterogeneity in Federated Learning

Junyuan Zhang[2*] Shuang Zeng[1*] Miao Zhang[3] Runxi Wang[2] Feifei Wang[1]
Yuyin Zhou[5] Paul Pu Liang[4] Liangqiong Qu[1†]
[1]The University of Hong Kong [2]Beihang University [3]New York University
[4]Carnegie Mellon University [5]UC Santa Cruz

Abstract

Federated learning (FL) is a powerful technology that enables collaborative training of machine learning models without sharing private data among clients. The fundamental challenge in FL lies in learning over extremely heterogeneous data distributions, device capacities, and device state availabilities, all of which adversely impact performance and communication efficiency. While data heterogeneity has been well-studied in the literature, this paper introduces FLHetBench, the first FL benchmark targeted toward understanding device and state heterogeneity. FLHetBench comprises two new sampling methods to generate real-world device and state databases with varying heterogeneity and new metrics for quantifying the success of FL methods under these real-world constraints. Using FLHetBench, we conduct a comprehensive evaluation of existing methods and find that they struggle under these settings, which inspires us to propose BiasPrompt+, a new method employing staleness-aware aggregation and fast weights to tackle these new heterogeneity challenges. Experiments on various FL tasks and datasets validate the effectiveness of our BiasPrompt+ method and highlight the value of FLHetBench in fostering the development of more efficient and robust FL solutions under real-world device and state constraints.

1. Introduction

Federated learning (FL) is a cutting-edge technology that enables collaborative training of deep learning models across multiple clients without sharing their local data, thereby protecting users' sensitive information and mitigating the risk of data leaks [26, 27, 61]. Despite its potential, the large-scale adoption of FL is challenged by various dimensions of heterogeneity, which can impede federated computation and lead to decreased performance and increased communication costs [3, 6, 29, 30, 60].

There are three main types of heterogeneity in FL: (1) *data heterogeneity*, referring to varying data distribution across clients, (2) *device heterogeneity*, characterized by diverse device capacities among clients [6, 30, 52], and (3) *state heterogeneity*, involving inconsistent client availability [49, 60]. Considerable research has been conducted to understand and address data heterogeneity in FL, including a range of simulated and real-world FL datasets [10, 29, 60], efficient assessment metrics for quantifying data heterogeneity [29, 39], and various efficient data heterogeneity optimization methods [5, 34, 37, 38, 50, 64, 65, 67, 68]. In contrast, efforts to investigate device and state heterogeneity are limited due to the lack of benchmarks and evaluation metrics reflecting these dimensions of heterogeneity in the real world. Prior studies either use simulated environments [8, 18, 26, 62] or are limited to a small set of real datasets [10, 29, 60]. This leads to a critical, yet unanswered question: *What happens to different FL algorithms when they are employed in real-world FL environments with varying degrees of device and state heterogeneity?*

FLHetBench: Device and state evaluation benchmark. To answer this key question, we introduce **FLHetBench**, the first real-world device and state heterogeneity evaluation benchmark in FL. As shown in Fig. 1, our FLHetBench consists of: (1) Two innovative Dirichlet process-based sampling methods - Dirichlet Process Gaussian Mixture Model (DPGMM) for continuous device data and Dirichlet Process Construction-Based Sampling Method (DPCSM) for discrete state data. DPGMM and DPCSM are capable of generating real-world device and state databases with diverse heterogeneity, as validated by our theoretical and empirical results. (2) Several isolated and interplay metrics, based on Monte Carlo (MC) simulations and clients' successful participating ratio, to assess device/state heterogeneity in FL. Our sampling methods and metrics enable FL practitioners to evaluate existing FL methods and inspire future work under real-world device and state constraints.

Benchmarking existing FL methods with FLHetBench. We then conduct the first comprehensive evaluation of existing FL methods using our FLHetBench and identify two key findings: (1) Most methods perform well under mild device/state heterogeneity, but struggle with increased het-

* These authors contributed equally to this work.
† Corresponding author. Email: liangqqu@hku.hk
This work was conducted when J. Zhang and R. Wang were interns at HKU.

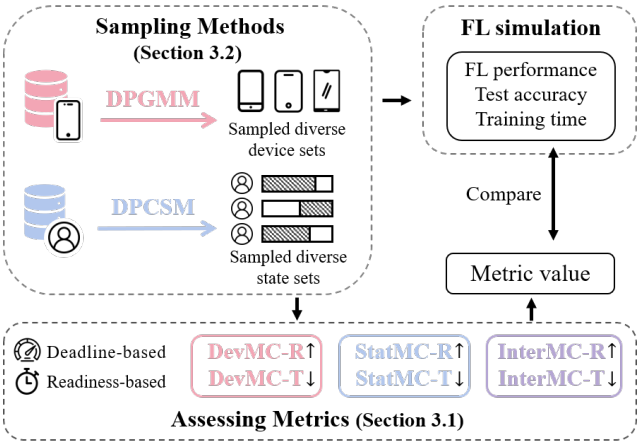

Figure 1. **Overview**: FLHetBench consists of 1) two sampling methods, DPGMM for continuous device database and DPCSM for discrete state database, to sample real-world device and state datasets with varying heterogeneity; and 2) new isolated (DevMC-R, DevMC-T, StateMC-R, StateMC-T) and interplay metrics (InterMC-R and InterMC-T) to evaluate FL methods under device and state heterogeneity. ↑= higher is better (less heterogeneous) and ↓= lower is better (more heterogeneous).

erogeneity. Moderate/High device/state heterogeneity poses a significant challenge to the efficiency of current FL methods. (2) The increased wall-clock time of clients and the low resource utilization of participating clients, caused by device/state heterogeneity, are the primary factors contributing to the performance degradation of current FL methods in heterogeneous real-world device/state scenarios.

Solution: Addressing heterogeneity with BiasPrompt+. Motivated by the above investigation, we introduce BiasPrompt+, a novel method employing gradient surgery-based staleness-aware aggregation (maximizing resource utility) and fast weights (minimizing communication/computation costs) to address device and state heterogeneity in FL. In contrast to existing FL methods that often impose extra computational burdens on local clients [55] or incur resource wastage [5], BiasPrompt+ concurrently reduces communication and computational burdens, maximizes the utilization of available clients, and maintains FL performance. Extensive experiments on a diverse range of FL tasks validate the superiority of BiasPrompt+ over competing methods. Nevertheless, BiasPrompt+ still encounters performance drops in highly heterogeneous situations, underscoring the need for future research with FLHetBench.

Contributions. We summarize our main contributions:

- We introduce FLHetBench, a *pioneering* benchmark for evaluating device and state heterogeneity in FL. Our real-world databases, sampling methods, and metrics are released at: https://carkham.github.io/FL_Het_Bench/, facilitating future exploration of this pivotal field.
- We conduct the *first* comprehensive evaluation of FL on varying degrees of device and state heterogeneity using FLHetBench, revealing that long wall-clock time and low resource utilization of participating clients contribute to the performance degradation of current FL methods in heterogeneous real-world device/state scenarios.
- We propose a simple and efficient method, BiasPrompt+, to mitigate device/state heterogeneity challenges. Extensive experimental results validate the superiority of our BiasPrompt+ over competing methods.

2. Background and Related Work

In this work, we aim to address the device and state heterogeneity challenge in cross-device FL [25].

Device heterogeneity in FL arises from varying capacities, hardware, and network speeds, leading to diverse communication costs and wall-clock time. To address this, synchronous FL and asynchronous FL have been proposed. Synchronous FL employs communication-efficient techniques like gradient compression [40, 53], local models [12, 33], knowledge distillation [41, 55], and model pruning [13, 23]. On the other hand, asynchronous FL addresses device heterogeneity by allowing clients to independently update their models, enhancing resource utilization for slower clients [16, 21, 36]. However, the use of stale models in asynchronous FL can lead to decreased convergence rate and model accuracy [57]. It is worth noting that both synchronous and asynchronous FL methods often evaluate the effectiveness of their approaches using homogeneous device simulation environments or by manually designating clients as stragglers [5, 25, 31, 32, 43], which may not accurately represent real-world device statuses. Assessing a device optimization method's ability to adapt to varying degrees of device heterogeneity remains challenging owing to the lack of datasets and metrics.

State heterogeneity in FL. State heterogeneity in FL refers to the varying and dynamic running environments of participating clients. This can lead to unselected or frequently disconnected clients, causing missed round deadlines and decreased FL performance. FLASH [60] was the first to demonstrate the significant impact of state heterogeneity on FL training and introduced a large-scale real-world state dataset. Later research [43] suggested setting a threshold for straggler devices and examining state heterogeneity effects with naive stragglers. However, measuring client state quality remains unclear, and generating diverse state databases has not been extensively explored.

Benchmarks in FL. Various efforts have been made to benchmark FL from different aspects, including real-world heterogeneous FL dataset [7, 29, 44], personalized FL [9, 56], heterogeneous device and state FL dataset [3, 4, 10, 29, 49, 54, 60]. Among these, research on benchmark heterogeneous devices and state FL is most closely related to our work. However, certain limitations exist in current work, as shown in Tab. 1. For example, FedScale's device data

Table 1. Comparison of FLHetBench with other FL benchmarks. #Heter.dev.dist. and #Heter.sta.dist. indicate the number of device and state distributions, respectively. ✘ implies no support, and ✔ indicates extensive support.

	FLASH	FedScale	FS-Real	**FLHetBench**
Device	Limited	Limited	Large-scale	Large-scale
State	✔	✔	✔	✔
#Heter.dev.dist.	✘	✘	3	✔
#Heter.sta.dist.	✘	✘	✘	✔
Metrics	✘	✘	✘	✔

format is single-point, which may not capture network fluctuations effectively. Although FS-Real [10] offers large-scale real-world device and state datasets, it only offers three types of device distributions. Extraction of device and state datasets with diverse distributions, which is essential for conducting comprehensive real-world FL experiments, remains unexplored. Moreover, there is a lack of metrics to evaluate the device/state heterogeneity FL. In this paper, we aim to fill these gaps with the proposed FLHetBench.

3. FLHetBench

Our FLHetBench (see Fig. 1) consists of 1) two sampling methods, DPGMM for continuous device database and DPCSM for discrete state database, to sample real-world device and state datasets with varying heterogeneity; and 2) various metrics to assess the device/ state heterogeneity in FL. We will now delve into the details of each component.

3.1. Metrics for Assessing Device/State Heterogeneity

Determining how challenging real-world device/state databases pose for FL is nontrivial due to a lack of consensus on how to measure it. Intuitively, one may consider using the statistical divergence metrics, which are commonly used to quantify the data heterogeneity in FL (e.g., JS distance [29] or pairwise KS statistics [39]), to assess the device/state heterogeneity. However, the FL performance is not directly related to this statistical divergence (see Sec. 5.2.1 for experimental results). In fact, the impact of a real-world device/state database on FL is shaped by various confounding factors, such as device capacities, device divergence, state status, and server aggregation strategies, see Fig. 2.

Two common server aggregation strategies in FL training are: (1) Deadline-based strategy where clients must complete tasks within a specified deadline (denoted as ddl), or the server proceeds without them. (2) Readiness-based strategy where the server waits for a specified proportion of clients to complete their tasks without imposing a ddl. Apparently, a single metric cannot accurately capture the influence of device/state heterogeneity under these two aggregation strategies. The same set of device/state databases can exhibit different levels of heterogeneity for FL under different aggregation strategies. For example, deadline-based

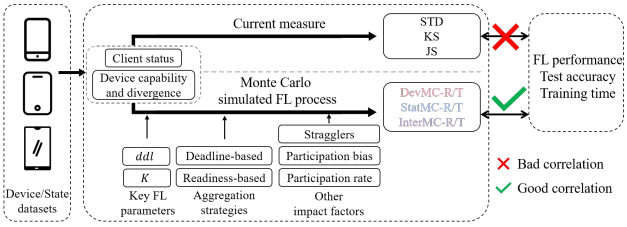

Figure 2. Our metrics overview: Existing metrics such as STD, KS, and JS primarily focus on statistical divergence, neglecting other confounding FL factors, thus failing to capture device/state heterogeneity. Our metrics use Monte Carlo simulations to mimic a realistic FL environment, taking into account various confounding factors (e.g., device capacities, state status, FL training strategy) to effectively capture device/state heterogeneity in FL.

FL performance may be impacted by successful participation rates, while readiness-based FL could be affected by the communication cost of straggler clients. In this paper, we introduce a comprehensive set of metrics that account for these confounding FL factors, enabling accurate assessment of device/state heterogeneity in FL.

3.1.1 Metrics for Deadline-based Strategy

As discussed above, device heterogeneity impacts FL performance by influencing the successful participation rate and participation bias. For example, mild heterogeneity results in a diverse set of clients successfully participating in FL training within a given ddl, whereas high heterogeneity may lead to fewer successful clients, causing slower convergence and poorer performance. However, estimating participation rate and participation bias through real-world FL training is impractical and time-consuming. Therefore, we propose using Monte Carlo (MC) simulations to mimic the real FL training process, enabling metric estimation while reducing computational costs and time. Specifically, we propose metrics **DevMC-R**, **StatMC-R**, and **InterMC-R** to measure the device, state, and interplay (refers to both device and state) heterogeneity in FL, respectively, by capturing the successful participation rate and participation bias.

Given N clients with their associated N device/state databases, let t^i_{cost} denote the actual cost for client i to complete one round of FL task, $\mathcal{S}_\text{i}$ denote the number of successful participation times of client i obtained via MC simulation. $\mathcal{S}_{ideal}$ denotes the ideal number of successful participation times, assuming all clients have the same device capacity and states are always available. Then the successful participation rate with these associated N clients can be defined as $\frac{\sum_{i=1}^{N} \mathcal{S}_\text{i}}{\mathcal{S}_{ideal} * N}$. However, this rate ignores the impact of participation bias on FL performance, where certain clients dominate the FL process may lead to decreased performance. We thus introduce decay functions $F(x) = \log(x+1)$ and $Clip\left(\mathcal{S}_i, 0, \mathcal{S}_{ideal}\right)$ (a clip function) to account for the participation bias. Consequently, our

DevMC-R, **InterMC-R** metrics are calculated as follows:

$$\textbf{DevMC-R}, \textbf{InterMC-R} = \frac{F(\sum_{i=1}^{N} Clip(\mathcal{S}_i, 0, \mathcal{S}_{ideal}))}{F(\mathcal{S}_{ideal} * N)}. \tag{1}$$

DevMC-R and InterMC-R effectively assess device and interplay heterogeneity in FL by accurately capturing the successful participation rate and participation bias. For detailed MC simulation of $\mathcal{S}_\text{i}$ for DevMC-R and InterMC-R, refer to Algorithm S3 and Algorithm S4 in Appendix, respectively.

StatMC-R for state heterogeneity. Eq. (1) is not suitable for StatMC-R, since t^i_{cost} is not specified when we only consider the state heterogeneity. For easier calculation, we assume t^i_{cost} follow a prior uniform distribution $p\left(t^i_{cost}\right)$ of $(0, ddl)$, ensuring our state metric is applicable under varying device capacities. Our StatMC-R is then updated from Eq. (1) with integration to t^i_{cost} as follows:

$$\textbf{StatMC-R} = \int_0^{ddl} \left(\frac{F\left(\sum_{i=1}^{N} Clip(\mathcal{S}_i, 0, \mathcal{S}_{ideal})\right)}{F(\mathcal{S}_{ideal} * N)} \right) p\left(t^i_{cost}\right) dt^i_{cost}. \tag{2}$$

We use Algorithm S4 to derive $\mathcal{S}_\text{i}$ for StatMC-R, considering sole state databases. DevMC-R, StatMC-R, and InterMC-R accurately measure participation rate and bias through MC mimic FL processes, providing valuable insights into the impact of device, state, and interplay heterogeneity on FL. A lower participation rate with higher participation bias results in smaller DevMC-R, StatMC-R, and InterMC-R, indicating higher levels of device, state, and interplay heterogeneity, respectively.

3.1.2 Metrics for Readiness-based Strategy

In FL training with a readiness-based strategy, the impact of device/state heterogeneity is represented by the total communication cost required for clients to achieve their target performance. However, the specific FL task and dataset are unknown during metric calculation. To this end, we use MC simulation to emulate an actual FL training process, allowing simulated clients to train until a pre-defined number of training trips ($trips$) are completed, rather than having simulated clients reach a target performance goal. We track the total communication cost T as our evaluation metrics and introduce **DevMC-T**, **StatMC-T**, and **InterMC-T** to measure device, state, and interplay heterogeneity in FL, respectively. Please refer to Algorithm S1 and Algorithm S2 for the detailed simulation of T for our metrics.

These metrics enable us to assess the efficiency of a readiness-based FL aggregation strategy by directly estimating the communication cost with their associated devices/states, thereby quantifying the device/state heterogeneity in the FL process. A longer simulated training time T corresponds to larger DevMC-T, StatMC-T, and InterMC-T values, indicating a higher level of heterogeneity.

3.2. Simulating Device and State Heterogeneity

Simulating large device/state databases with varying degrees of heterogeneity is crucial for evaluating the impact of device/state heterogeneity on FL. [20] effectively used the Dirichlet distribution to simulate data distributions with varying label heterogeneity. However, the Dirichlet distribution is inadequate for simulating more complex device/state heterogeneity, as its concentration parameter can only control the shape and concentration of the distribution. For example, device heterogeneity includes factors like device speed variations. Comparing speeds between 100 and 1000 shows higher heterogeneity than between 100 and 110. The Dirichlet distribution, however, cannot account for these differences. Similarly, a client's state is a collection of discrete data (e.g., available, unavailable, available), and capturing these nuances is a complex task that the Dirichlet distribution cannot directly handle. Here we propose two advanced Dirichlet process-based methods to simulate device and state heterogeneity in FL.

3.2.1 Dirichlet Process Gaussian Mixture Model

Baseline real-world device database construction. We build a comprehensive real-world device database, i.e., **FL-Device**, by collecting data from around 10,000 popular mobile devices. Specifically, for each device, we will characterize 1) its computational latency using [22] and our custom app, and 2) its communication latency using [1]. Please refer to Appendix Sec. II for more details about our APP.

Sampling varying degrees of device heterogeneity. As depicted in Appendix Fig. S1a, we propose using a Dirichlet Process Gaussian Mixture Model (DPGMM) to generate device databases with varying heterogeneity degrees from a baseline dataset $\mathcal{D}$, while maintaining consistent average speed across the sampled databases. We control heterogeneity using the total number of distinct devices K_n allocated to M clients ($K_n <= M$) and the variations σ of the assigned device capacities (speeds). Firstly, we specify the Gaussian distribution for each of the K_n devices by setting the k^{th} as device $\mathcal{N}\left(\mu_k, \sigma_k^2\right)$, which can be drawn from a modified prior base distribution $G_{0'}$, with $\mu_k \mid G_{0'} \sim \mathcal{N}\left(\mu_0, (\sigma \cdot \sigma_0)^2\right)$. Here μ_0 and σ_0 are the mean and standard speeds of the baseline device database $\mathcal{D}$ and σ is used to control the variation of the selected K_n devices. σ_k is obtained from the base device database $\mathcal{D}$ by identifying a device with the closest mean to μ_k and assigning its standard deviation to σ_k. In real applications, μ_0 can be set to any reasonable value, e.g., set to a high value to generate a device database with high capacities. Secondly, we assign each device of K_n to M clients via Dirichlet distribution [17]. Denote c_i as the device assigned to the i^{th} client, then $c_i \mid \pi_1, \cdots, \pi_{K_n} \sim \text{Discrete}\left(\pi_1, \cdots, \pi_{K_n}\right)$, where the mixing proportions π are generated from a Dirichlet distribution $\pi_1, \cdots, \pi_{K_n} \mid \alpha \sim \text{Dir}\left(\frac{\alpha}{K_n}, \cdots, \frac{\alpha}{K_n}\right)$. We set α as a constant 1000 and use K_n to control the diver-

gence of the device database, where large values indicate more distinct samples.

Validation of DPGMM. Lemma 3.1, Theorem 3.1 (see Appendix Sec. I.2 for proof) and sampled varying device heterogeneous databases in Fig. S4 prove that our DPGMM can effectively generate samples with varying degrees of device heterogeneity, ranging from lower to higher heterogeneity, by manipulating K_n and σ.

Lemma 3.1. *For each specified set of K_n and σ, the variance of the sampled database is $\frac{\alpha}{K_n}\sum_{k=1}^{K_n}\sigma_k^2 + (\alpha - K_n)\mu_0^2 + K_n(\sigma\cdot\sigma_0)^2$.*

Theorem 3.1. *With different specified sets of K_n and σ, the variance of the sampled database can cover a range of $\left(\min_{K_n}(\frac{\alpha}{K_n}\sum_{k=1}^{K_n}\sigma_k^2 + (\alpha - K_n)\mu_0^2), +\infty\right)$.*

3.2.2 Dirichlet Process Construction-based Sampling

Baseline real-world state database. We use an existing large-scale state database [60] with 136k states as our baseline real-world state database.

Sampling varying degrees of state heterogeneity. A client's state is discrete and cannot be characterized by Gaussian distributions like device. Additionally, a client's state is a collection of discrete data and is thus unsuitable for Dirichlet process. We employ the idea of StatMC-R and use a single state metric (adapted from StatMC-R, see Appendix Sec. V) to transform the discrete state into one single data point, making them compatible with the Dirichlet process [48]. We then introduce a Dirichlet process construction-based sampling method (DPCSM) to generate state databases with varying heterogeneity levels (see Appendix Fig. S1b). Our DPCSM considers state heterogeneity influenced by two factors: $startRank$ and α and selects a new state database with K_n states from a baseline dataset of N states ($K_n < N$) as follows: (1) Sort states in the baseline set by single state metric, denoted as $D_{(1)} > \cdots > D_{(N)}$. (2) Introduce $startRank$ to represent the rank of the optimal state from the baseline dataset, i.e., selecting states from $D_{(i)}, i = startRank, \cdots, N$. A lower $startRank$ indicates a higher quality optimal state. (3) Use the stick-breaking process to determine the probability of state $D_{(k)}$ being selected as $\pi_k = b_k(1-\sum_{j=startRank}^{k-1}\pi_j)$, $k = startRank+1, \cdots, N$; $\pi_{startRank} = b_1$, and $b_k \overset{i.i.d}{\sim} beta(1,\alpha)$. A smaller α leads to a lower probability of selecting subsequent states, concentrating on states with higher single state metrics.

Validation of DPCSM. As evidenced by Lemma 3.2 and Theorem 3.2 (see Appendix Sec. I.2 for proof), DPCSM enables the generation of samples with varying degrees of state heterogeneity, by manipulating $startRank$ and α.

Lemma 3.2. *For each specific set of $startRank$ and α, the variance of the sampled database is given by $\sum_{i=startRank}^{N}\pi_i(X_i - \sum_{i=startRank}^{N}\pi_i X_i)^2$, where $\pi_i = b_i(1-\sum_{j=startRank}^{i-1}\pi_j)$, $i > 1$; b_1, $i = 1$ with $b_i \overset{i.i.d}{\sim} beta(1,\alpha)$, and X_i denotes the single state metric of state $D_{(i)}$.*

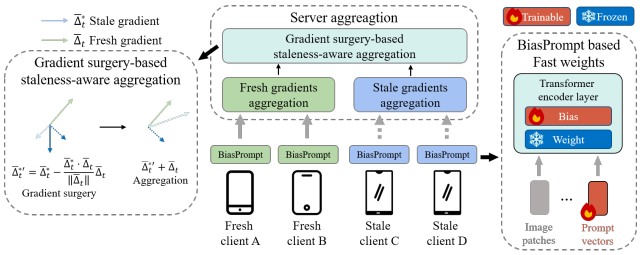

Figure 3. **BiasPrompt+** consists of 1) a gradient surgery-based staleness-aware aggregation strategy for maximizing resource utility, and 2) a BiasPromt module based on fast weights for minimizing communication/computation cost.

Theorem 3.2. *With different specified sets of $startRank$ and α, the variance of the single state metric of the sampled database can cover a range of $\left(0, (X_1 - X_N)^2\right)$, where X_i denotes the single state of state $D_{(i)}$.*

4. BiasPrompt+ to Tackle Heterogeneity

In this section, we introduce BiasPrompt+, a novel method designed to address device and state heterogeneity challenges in FL. As depicted in Fig. 3, BiasPrompt+ comprises two modules: a gradient surgery-based staleness-aware aggregation strategy for maximizing resource utility, and a communication-efficient module BiasPrompt based on fast weights. We will discuss each component in detail below.

Staleness-aware aggregation. When facing device and state heterogeneity, FL methods tend to favor clients with high capability, leading to poor resource utilization of low-end clients and subpar performance. We develop a gradient surgery-based staleness-aware strategy, allowing low-end clients to submit stale gradients beyond the deadline while preventing delayed clients from deviating significantly from fresh clients. This strategy is motivated by the opposing gradient directions between stale gradients and fresh gradients observed in [63]. Specifically, let $\bar{\Delta}_t^*$ represent the averaged gradients of delayed clients in t round, and $\bar{\Delta}_t$ denote the averaged fresh gradients in t round. The gradient surgery-based staleness-aware aggregation is calculated as: $\bar{\Delta}_t^{*\prime} = \bar{\Delta}_t^* - \frac{\bar{\Delta}_t^*\cdot\bar{\Delta}_t}{\|\bar{\Delta}_t\|}\bar{\Delta}_t$. Our gradient surgery strategy projects stale gradients onto the normal plane of fresh gradients and adds only the non-conflicting component, reducing the impact of stale clients and emphasizing those with greater similarity to the average fresh clients.

Communication efficient weight adaptation. In addition to the above strategy, we also introduce communication-efficient BiasPrompt based on fast weights. Inspired by prompt tuning [24], we introduce several extra prompts as fast weights [46], enabling our model to quickly acquire new information. However, while the fast weights can learn rapidly, they also decay rapidly. To address this issue, we also update a set of "stable weights", the bias term of the

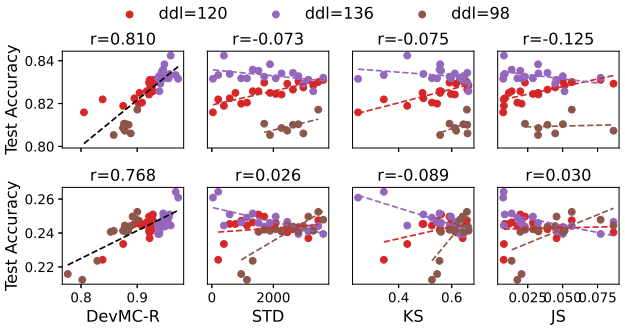

Figure 4. Empirical relationship between device heterogeneity metrics and FedAVG test accuracy on COVID-FL (first row) and OpenImage (second row) using three deadline-based strategies (*ddl*=120, 136, 98). Each point in the scatter-plot corresponds to an experiment with different device sets or *ddl*, along with metrics DevMC-R, Standard deviation (STD), Kolmogorov-Smirnov test (KS) and Jensen–Shannon divergence (JS). DevMC-R metric is the most effective metric in capturing device heterogeneity, as indicated by the highest correlation coefficient r.

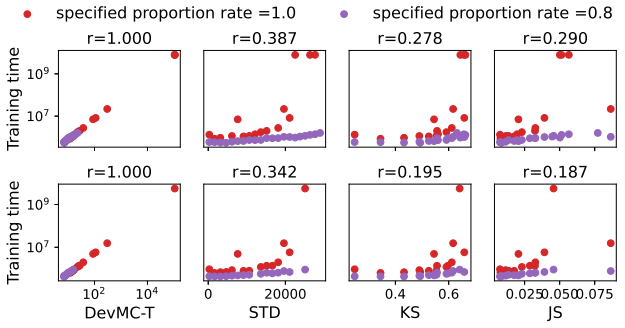

Figure 5. Empirical relationship between metrics and FedAVG training time on COVID-FL (first row) and OpenImage (second row) using two readiness-based strategies: waiting for all clients (proportion ratio=1.0) and 80% of clients (proportion ratio=0.8). Points represent varying device sets or proportion rates. Please note that all readiness-based strategy graphs in this paper use logarithmic scales for clear comparison. DevMC-T exhibits a higher correlation ($r > 0.95$) with actual training time than other metrics (STD, KS, JS), emphasizing its efficiency.

networks, in order to stabilize the training process without extra communication/computational burdens. Specifically, for a plain Vision Transformer (ViT) with N layers, let $\mathbf{A}_i, i = 1, 2, \cdots, N$ denotes the intermediate image patch embeddings of d dimension as $(i-1)^{th}$ layer output, and $\mathbf{x}_i$ is the [class token] at $(i)^{th}$ layer's input space. We introduce a set of p continuous embeddings $\mathbf{P}_i \in \mathbb{R}^{p \times d}$ which is prepended to the input of i^{th} layer as $(\mathbf{x}_i, \mathbf{P}_i, \mathbf{A}_i)$. Only the extra prompts and the bias terms are trainable, while all the others are kept frozen during the entire training procedure. We also incorporate server momentum [20] into BiasPrompt to deal with heterogeneous datasets. BiasPrompt significantly reduces communication and computational cost without compromising performance.

5. Experiments

5.1. Experimental Setup

Dataset. We validate our metrics, sampling methods, models using COVID-FL [59] and OpenImage [2]. For COVID-FL, we sample 1,000 images, randomly assign 10 images to each client's local training set, and use the original test set as our global test set. For OpenImage, we randomly choose 100 real-world clients [29], using their respective validation and test sets to form global sets.

Training recipe. We use ViT-B [15] pre-trained on ImageNet-1k [14] as the baseline network [39]. For deadline-based strategy, we set a target total communication round and use its final prediction accuracy as true FL performance. For readiness-based strategy, we allow FL training to continue until a target accuracy is reached and use the total training time as true FL performance. Please refer to Appendix Sec. III for other experimental details.

5.2. Validating Heterogeneity Metrics

In this section, we validate the effectiveness of our heterogeneity assessment metrics by correlating them with actual FL performance (e.g., prediction accuracy and total training time) on COVID-FL and OpenImage datasets. We utilize the popular FedAVG [35] algorithm in our analysis. The Pearson correlation coefficient r [11] is employed as a quantitative measure of the relationship.

5.2.1 Validating Device Heterogeneity Metrics

We apply our DPGMM approach to sample 21 sets of device databases from our base device dataset, with each set consists of 100 devices. We set the state of all clients to be available to eliminate the influence of state heterogeneity. More details could refer to Appendix Sec. III.

FL prediction accuracy under deadline-based strategy and DevMC-R metric. Fig. 4 displays the correlation between our DevMC-R and three common heterogeneity measuring metrics (STD, JS, and KS) with the actual prediction accuracy of FedAVG on COVID-FL and OpenImage using our sampled 21 device sets. As depicted in Fig. 4, STD, JS, and KS metrics focus on the divergence between device speeds but neglect confounding factors in FL training, resulting in a low correlation with test accuracy. Conversely, our DevMC-R metric consistently exhibits a high correlation with FL performance across all settings, highlighting the superiority of our DevMC-R metric in accurately representing the impact of device heterogeneity on federated learning performance.

FL training time under readiness-based strategy and DevMC-T metric. The total training time under the readiness-based strategy is typically determined by the slowest client, known as the straggler. Notably, our DevMC-T metric effectively captures the influence of strag-

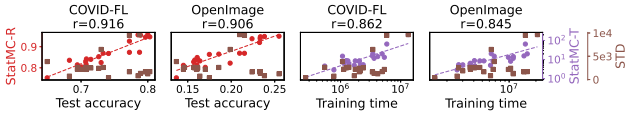

Figure 6. Left two images: State assessment metrics (StatMC-R, STD) vs. FedAVG test accuracy using deadline-based strategy. Right two images: StatMC-T and STD vs. FL training time for FedAVG with readiness-based strategy. StatMC-R shows a strong correlation ($r > 0.84$) with FL performance, emphasizing its effectiveness in capturing state heterogeneity in FL.

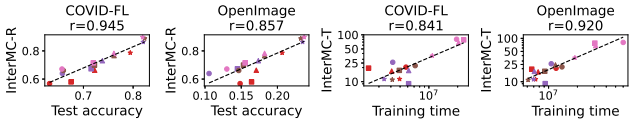

Figure 7. Left two images: InterMC-R vs. FedAVG test accuracy using deadline-based strategy. Right two images: InterMC-T vs. FL training time using readiness-based strategy. The same color indicates the same device sets. The same marker denotes the same state sets. InterMC-R and InterMC-T exhibit strong correlation ($r > 0.84$) with actual FL performance, highlighting their effectiveness in capturing interplay heterogeneity in FL.

glers within a device set, leading to a strong correlation with the true training time in all settings. This is evidenced by a Pearson correlation coefficient r near 1 in Fig. 5. In contrast, the comparison metrics (STD, KS, and JS) only measure the statistical divergence between devices, without providing insights into the impact of specific stragglers, thus exhibiting a weaker correlation with the actual training time.

5.2.2 Validating State Heterogeneity Metrics

We vary the $StartRank$ and α of DPCSM to generate 21 sets of heterogeneous state datasets, with each set containing 100 states. We use device-homogeneous setting, thereby eliminating the influence of device heterogeneity. More details could refer to Appendix Sec. III.

FL prediction accuracy under deadline-based strategy and StatMC-R metric. As illustrated in the left two images of Fig. 6, the commonly used statistical divergence metric, STD, is unable to capture confounding FL training strategies with its associated state database, resulting in a low correlation with FL performance. In contrast, our StatMC-R consistently demonstrates a high correlation with the true FL performance across all settings, with correlation coefficients r of 0.916 and 0.906, respectively, emphasizing the effectiveness of our proposed metric.

FL training time under readiness-based strategy and StatMC-T metric. Evaluating state heterogeneity in readiness-based FL strategy necessitates accounting for both wall-clock time and stragglers, which is challenging to model using statistical metrics but can be effectively revealed through our MC methods. As shown in the right two images of Fig. 6, our StatMC-T consistently demonstrates high correlation across all settings, emphasizing the efficiency of our proposed metric. See Appendix Sec. VI.1 for more results showcasing our metric's effectiveness.

5.2.3 Validation of Interplay Metrics

The interplay heterogeneity incorporates both device and state heterogeneity, making characterization challenging. We independently extract four sets of device and state databases with our DPGMM and DPCSM, ranging from mild to severe heterogeneity. Each set consists of 100 devices and 100 states. By creating all potential pairwise combinations, we generate 16 distinct interplay device and state sets, representing diverse real-world FL scenarios with varying device and state heterogeneity. See Appendix Sec. III for more details.

In the deadline-based strategy (depicted in the first row of Fig. 7), it becomes apparent that state heterogeneity has a greater impact on final accuracy, as seen in the larger variations among same-colored points. Conversely, in the readiness-based strategy (second row of Fig. 7), slow devices from device heterogeneity dominate, as evidenced by longer training times for pink points. This highlights the complexity of characterizing device and state heterogeneity interplay in FL. Nevertheless, both InterMC-R and InterMC-T metrics show strong correlations ($r > 0.84$) with FL performance, indicating their ability to capture this intricate interaction and provide accurate estimations.

5.3. Benchmarking FL Methods with FLHetBench

I. Stragglers fail to capture real-world device/state heterogeneity. Current methods often attempt to mimic real-world heterogeneity by artificially setting some clients as stragglers and varying the ratio of stragglers to normal clients [5, 25, 31, 32, 51, 58]. We conduct experiments to highlight the limitations of these approaches in evaluating FL optimization methods. As shown in the first row of Fig. 8, when training FL with a readiness-based strategy, the training time for FedAVG [35] and FetchSGD [40] remains nearly unchanged (under random selection strategy) or decreases (under identifying slower clients as straggler strategy) for different straggler ratios (from 0.3 to 0.9). This contradicts real-world situations where the training time should increase as more clients fail to successfully participate in FL training. These observations demonstrate that existing straggler-mimic strategies do not accurately represent real-world FL scenarios, *underlining the need for a more comprehensive benchmark to better evaluate and understand the performance of FL optimization methods*.

II. Benchmarking FL methods with FLHetBench. In this section, we benchmark current FL methods by conducting extensive experiments with our FLHetBench, assessing the effectiveness of different FL algorithms across a wide range of real-world heterogeneous scenarios. We involve several representative FL heterogeneity optimization algorithms including (1) Regularization based optimization methods for fast convergence, such as FedProx [31], Fed-

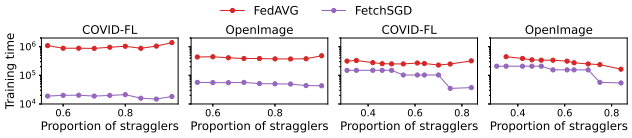

Figure 8. The FL training time of random selection (left two images) [5, 31] and straggler identification (right two images) [25, 32, 51, 58]. Both strategies do not capture the decrease in FL training time as the number of successful clients increases.

Table 2. Test accuracy(%) vs. InterMC-R for various FL algorithms under mild, moderate, severe interplay (device and state) heterogeneity on OpenImage. BiasPrompt+ consistently outperforms competitors.

InterMC-R	FedAVG	FedProx	FedKD	FedDyn	FetchSGD	FedBuff	**BiasPrompt+**
0.56 (severe)	14.84	14.78	13.94	19.22	20.78	15.87	**35.75**
0.73 (moderate)	17.36	19.97	16.24	17.67	25.62	17.11	**33.79**
0.86 (mild)	23.48	23.98	18.94	22.70	33.60	16.77	**35.24**

Dyn [5], (2) Communication efficient algorithms for reducing the shared model parameters, including FedKD [55] and FetchSGD [40]. (3) Asynchronous FL FedBuff [36]. More experimental details are shown in Appendix Sec. III.

Observation 1: Most methods perform well under mild device/state heterogeneity. As per Fig. 9, the majority of current FL algorithms maintain test accuracy close to the heterogeneous-unaware setting (InterMC-R=1, no device/state heterogeneity) when handling mild heterogeneity (InterMC-R values around 0.9). It is noteworthy that FedDyn and FedKD, which incorporate additional local trainable parameters, may experience a substantial accuracy degradation of up to 10% even at InterMC-R=0.9. This may be attributed to the fact that even mild heterogeneity can lead to the staleness of less active clients' local parameters, which greatly affects final test accuracy.

Observation 2: Increased device and state heterogeneity is a big challenge. Current heterogeneity-aware algorithms are effective in FL with mild device/state heterogeneity but struggle with increased heterogeneity. We identify two primary factors that cause the performance degradation of current FL methods in device and state heterogeneous scenarios. (1) **Increased wall-clock time**: Increased device and state heterogeneity lead to longer wall-clock time, causing missed report deadlines and decreased performance for algorithms that neglect the wall-clock time, such as FedProx, FedDyn and FedBuff. In contrast, algorithms minimizing wall-clock time, such as FetchSGD, show better performance. (2) **Low resource utilization of participating clients:** Despite reduced wall-clock time, FetchSGD still suffers performance drops in moderate-to-high heterogeneity, e.g., 19.5% accuracy drop on InterMC-R=0.6 compared to InterMC=0.9 on OpenImage. This is because FetchSGD always prioritizes active clients with high capacity and availability but disregards updates from stale clients, thus leading to wasted participant clients and subpar performance in highly heterogeneity settings.

In summary, only addressing either increased wall-clock time or low resource utilization individually is insufficient for ensuring robustness and effectiveness in diverse real-world applications. We recommend a comprehensive approach to heterogeneity when studying FL algorithms and encourage the use of our FLHetBench to validate their scalability and stability in real-world settings.

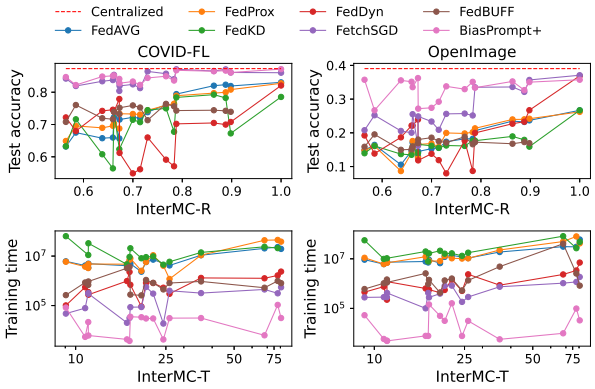

Figure 9. First row: InterMC-R vs. test accuracy for FL algorithms on COVID-FL/OpenImage with deadline-based strategy. Second row: InterMC-T vs. FL training time using readiness-based strategy. InterMC-R=1 denotes no device/state heterogeneity. BiasPrompt+ consistently surpasses competing methods.

III. The superiority of BiasPrompt+. As shown in Fig. 9 and Tab. 2, BiasPrompt+ reduces wall-clock time with less communication cost, increases resource utility, and achieves the best performance across all settings. Nevertheless, BiasPrompt+ still experiences performance declines in highly heterogeneous conditions, emphasizing the need for further research with FLHetBench to develop strategies addressing device/state heterogeneity in various real-world scenarios.

6. Conclusion

In this paper, we study and improve how FL methods perform under real-world device computational constraints and state availabilities. Through a new FLHetBench benchmark that simulates real-world device and state heterogeneity and newly proposed metrics to measure FL performance under these constraints, we identify long wall-clock times and low resource utilization of participating clients as the primary factors contributing to performance degradation. Motivated by these key findings, we propose BiasPrompt+, a novel method that employs staleness-aware aggregation for maximizing resource utility and fast weights to minimize communication costs. While BiasPrompt+ shows better results, we believe this is only a small step towards efficient and robust FL for the real world, and emphasize the importance of FLHetBench in advancing future machine learning methods tackling real-world heterogeneity.

7. Acknowledgments

The work was supported by the National Natural Science Foundation of China (No. 62306253).

References

[1] The M-Lab MobiPerf Data Set. https://measurementlab.net/tests/mobiperf. 4

[2] Google Open Images Dataset. https://storage.googleapis.com/openimages/web/index.html. 6

[3] Ahmed M Abdelmoniem, Chen-Yu Ho, Pantelis Papageorgiou, and Marco Canini. Empirical analysis of federated learning in heterogeneous environments. In *Proceedings of the 2nd European Workshop on Machine Learning and Systems*, pages 1–9, 2022. 1, 2

[4] Ahmed M Abdelmoniem, Chen-Yu Ho, Pantelis Papageorgiou, and Marco Canini. A comprehensive empirical study of heterogeneity in federated learning. *IEEE Internet of Things Journal*, 2023. 2

[5] Durmus Alp Emre Acar, Yue Zhao, Ramon Matas Navarro, Matthew Mattina, Paul N. Whatmough, and Venkatesh Saligrama. Federated learning based on dynamic regularization. In *ICLR*. OpenReview.net, 2021. 1, 2, 7, 8, 5

[6] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konečnỳ, Stefano Mazzocchi, Brendan McMahan, et al. Towards federated learning at scale: System design. *Proceedings of machine learning and systems*, 1:374–388, 2019. 1

[7] Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečnỳ, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018. 2

[8] Sebastian Caldas, Jakub Konečny, H Brendan McMahan, and Ameet Talwalkar. Expanding the reach of federated learning by reducing client resource requirements. *arXiv preprint arXiv:1812.07210*, 2018. 1

[9] Daoyuan Chen, Dawei Gao, Weirui Kuang, Yaliang Li, and Bolin Ding. pfl-bench: A comprehensive benchmark for personalized federated learning. *NeurIPS*, 35:9344–9360, 2022. 2

[10] Daoyuan Chen, Dawei Gao, Yuexiang Xie, Xuchen Pan, Zitao Li, Yaliang Li, Bolin Ding, and Jingren Zhou. FS-REAL: towards real-world cross-device federated learning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023, Long Beach, CA, USA, August 6-10, 2023*, pages 3829–3841. ACM, 2023. 1, 2, 3

[11] Israel Cohen, Yiteng Huang, Jingdong Chen, Jacob Benesty, Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. Pearson correlation coefficient. *Noise reduction in speech processing*, pages 1–4, 2009. 6

[12] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. In *ICML*, pages 2089–2099. PMLR, 2021. 2

[13] Rong Dai, Li Shen, Fengxiang He, Xinmei Tian, and Dacheng Tao. Dispfl: Towards communication-efficient personalized federated learning via decentralized sparse training. *arXiv preprint arXiv:2206.00187*, 2022. 2

[14] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. Ieee, 2009. 6, 4

[15] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 6

[16] Chen Dun, Mirian Hipolito, Chris Jermaine, Dimitrios Dimitriadis, and Anastasios Kyrillidis. Efficient and light-weight federated learning via asynchronous distributed dropout. In *International Conference on Artificial Intelligence and Statistics*, pages 6630–6660. PMLR, 2023. 2

[17] Dilan Görür and Carl Edward Rasmussen. Dirichlet process gaussian mixture models: Choice of the base distribution. *Journal of Computer Science and Technology*, 25(4): 653–664, 2010. 4

[18] Jenny Hamer, Mehryar Mohri, and Ananda Theertha Suresh. Fedboost: A communication-efficient algorithm for federated learning. In *ICML*, pages 3973–3983. PMLR, 2020. 1

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 7

[20] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019. 4, 6

[21] Dzmitry Huba, John Nguyen, Kshitiz Malik, Ruiyu Zhu, Mike Rabbat, Ashkan Yousefpour, Carole-Jean Wu, Hongyuan Zhan, Pavel Ustinov, Harish Srinivas, et al. Papaya: Practical, private, and scalable federated learning. *Proceedings of Machine Learning and Systems*, 4:814–832, 2022. 2, 1

[22] Andrey Ignatov, Radu Timofte, William Chou, Ke Wang, Max Wu, Tim Hartley, and Luc Van Gool. Ai benchmark: Running deep neural networks on android smartphones. In *ECCV Workshops*, pages 0–0, 2018. 4

[23] Berivan Isik, Francesco Pase, Deniz Gunduz, Tsachy Weissman, and Zorzi Michele. Sparse random networks for communication-efficient federated learning. In *ICLR*, 2023. 2

[24] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *ECCV*, 2022. 5

[25] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021. 2, 7, 8

[26] Jakub Konečnỳ, H Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016. 1

[27] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016. 1

[28] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 7

[29] Fan Lai, Yinwei Dai, Sanjay S. Singapuram, Jiachen Liu, Xiangfeng Zhu, Harsha V. Madhyastha, and Mosharaf Chowdhury. FedScale: Benchmarking model and system perfor-

mance of federated learning at scale. In *ICML*, 2022. 1, 2, 3, 6, 4

[30] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE signal processing magazine*, 37(3):50–60, 2020. 1

[31] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020. 2, 7, 8, 5

[32] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. In *ICLR*. OpenReview.net, 2020. 2, 7, 8

[33] Paul Pu Liang, Terrance Liu, Liu Ziyin, Nicholas B Allen, Randy P Auerbach, David Brent, Ruslan Salakhutdinov, and Louis-Philippe Morency. Think locally, act globally: Federated learning with local and global representations. *arXiv preprint arXiv:2001.01523*, 2020. 2

[34] Tao Lin, Lingjing Kong, Sebastian U Stich, and Martin Jaggi. Ensemble distillation for robust model fusion in federated learning. In *NeurIPS*, 2020. 1

[35] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017. 6, 7

[36] John Nguyen, Kshitiz Malik, Hongyuan Zhan, Ashkan Yousefpour, Mike Rabbat, Mani Malek, and Dzmitry Huba. Federated learning with buffered asynchronous aggregation. In *International Conference on Artificial Intelligence and Statistics, AISTATS 2022, 28-30 March 2022, Virtual Event*, pages 3581–3607. PMLR, 2022. 2, 8, 1, 5

[37] Liangqiong Qu, Niranjan Balachandar, and Daniel L Rubin. An experimental study of data heterogeneity in federated learning methods for medical imaging. *arXiv preprint arXiv:2107.08371*, 2021. 1

[38] Liangqiong Qu, Niranjan Balachandar, Miao Zhang, and Daniel Rubin. Handling data heterogeneity with generative replay in collaborative learning for medical imaging. *Medical Image Analysis*, 78:102424, 2022. 1

[39] Liangqiong Qu, Yuyin Zhou, Paul Pu Liang, Yingda Xia, Feifei Wang, Ehsan Adeli, Li Fei-Fei, and Daniel Rubin. Rethinking architecture design for tackling data heterogeneity in federated learning. In *CVPR*, pages 10061–10071, 2022. 1, 3, 6

[40] Daniel Rothchild, Ashwinee Panda, Enayat Ullah, Nikita Ivkin, Ion Stoica, Vladimir Braverman, Joseph Gonzalez, and Raman Arora. Fetchsgd: Communication-efficient federated learning with sketching. In *ICML*, pages 8253–8265. PMLR, 2020. 2, 7, 8, 5

[41] Felix Sattler, Arturo Marban, Roman Rischke, and Wojciech Samek. Communication-efficient federated distillation. *arXiv preprint arXiv:2012.00632*, 2020. 2

[42] David W Scott. *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons, 2015. 3

[43] Mehreen Tahir and Muhammad Intizar Ali. On the performance of federated learning algorithms for iot. *IoT*, 3(2): 273–284, 2022. 2

[44] Tensorflow Team. Tensorflow Federated. https://github.com/tensorflow/federated, 2021. 2

[45] TensorFlow Team. TensorFlow Lite. https://www.tensorflow.org/lite, 2023. Retrieved on: 30.7.2023. 4

[46] Tijmen Tieleman and Geoffrey Hinton. Using fast weights to improve persistent contrastive divergence. In *ICML*, pages 1033–1040, 2009. 5

[47] David Michael Titterington, Adrian FM Smith, and Udi E Makov. Statistical analysis of finite mixture distributions. *(No Title)*, 1985. 3

[48] UW UW. Introduction to the dirichlet distribution and related processes. 2010. 5

[49] Ewen Wang, Ajay Kannan, Yuefeng Liang, Boyi Chen, and Mosharaf Chowdhury. FLINT: A platform for federated learning integration. *CoRR*, abs/2302.12862, 2023. 1, 2

[50] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. In *ICLR*, 2020. 1

[51] Irene Wang, Prashant J Nair, and Divya Mahajan. Fluid: Mitigating stragglers in federated learning using invariant dropout. *arXiv preprint arXiv:2307.02623*, 2023. 7, 8

[52] Jianyu Wang, Zachary Charles, Zheng Xu, Gauri Joshi, H. Brendan McMahan, Blaise Agüera y Arcas, Maruan Al-Shedivat, Galen Andrew, Salman Avestimehr, Katharine Daly, Deepesh Data, Suhas N. Diggavi, Hubert Eichner, Advait Gadhikar, Zachary Garrett, Antonious M. Girgis, Filip Hanzely, Andrew Hard, Chaoyang He, Samuel Horváth, Zhouyuan Huo, Alex Ingerman, Martin Jaggi, Tara Javidi, Peter Kairouz, Satyen Kale, Sai Praneeth Karimireddy, Jakub Konečný, Sanmi Koyejo, Tian Li, Luyang Liu, Mehryar Mohri, Hang Qi, Sashank J. Reddi, Peter Richtárik, Karan Singhal, Virginia Smith, Mahdi Soltanolkotabi, Weikang Song, Ananda Theertha Suresh, Sebastian U. Stich, Ameet Talwalkar, Hongyi Wang, Blake E. Woodworth, Shanshan Wu, Felix X. Yu, Honglin Yuan, Manzil Zaheer, Mi Zhang, Tong Zhang, Chunxiang Zheng, Chen Zhu, and Wennan Zhu. A field guide to federated optimization. *CoRR*, abs/2107.06917, 2021. 1

[53] Yujia Wang, Lu Lin, and Jinghui Chen. Communication-efficient adaptive federated learning. In *ICML*, pages 22802–22838. PMLR, 2022. 2

[54] Herbert Woisetschläger, Alexander Isenko, Ruben Mayer, and Hans-Arno Jacobsen. Fledge: Benchmarking federated machine learning applications in edge computing systems. *arXiv preprint arXiv:2306.05172*, 2023. 2

[55] Chuhan Wu, Fangzhao Wu, Lingjuan Lyu, Yongfeng Huang, and Xing Xie. Communication-efficient federated learning via knowledge distillation. *Nature communications*, 13(1): 2032, 2022. 2, 8, 5

[56] Shanshan Wu, Tian Li, Zachary Charles, Yu Xiao, Ziyu Liu, Zheng Xu, and Virginia Smith. Motley: Benchmarking heterogeneity and personalization in federated learning. *arXiv preprint arXiv:2206.09262*, 2022. 2

[57] Chenhao Xu, Youyang Qu, Yong Xiang, and Longxiang Gao. Asynchronous federated learning on heterogeneous devices: A survey. *Computer Science Review*, 50:100595, 2023. 2

[58] Zirui Xu, Fuxun Yu, Jinjun Xiong, and Xiang Chen. Helios: Heterogeneity-aware federated learning with dynamically balanced collaboration. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pages 997–1002. IEEE, 2021. 7, 8

[59] Rui Yan, Liangqiong Qu, Qingyue Wei, Shih-Cheng Huang, Liyue Shen, Daniel Rubin, Lei Xing, and Yuyin Zhou. Label-efficient self-supervised federated learning for tackling data heterogeneity in medical imaging. *IEEE Transactions on Medical Imaging*, 2023. 6

[60] Chengxu Yang, Qipeng Wang, Mengwei Xu, Zhenpeng Chen, Kaigui Bian, Yunxin Liu, and Xuanzhe Liu. Characterizing impacts of heterogeneity in federated learning upon large-scale smartphone data. In *Proceedings of the Web Conference 2021*, pages 935–946, 2021. 1, 2, 5, 4

[61] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019. Publisher: ACM New York, NY, USA. 1

[62] Xin Yao, Tianchi Huang, Chenglei Wu, Rui-Xiao Zhang, and Lifeng Sun. Federated learning with additional mechanisms on clients to reduce communication costs. *arXiv preprint arXiv:1908.05891*, 2019. 1

[63] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *NeurIPS*, 33:5824–5836, 2020. 5

[64] Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Nghia Hoang, and Yasaman Khazaeni. Bayesian nonparametric federated learning of neural networks. In *ICML*, pages 7252–7261. PMLR, 2019. 1

[65] Miao Zhang, Liangqiong Qu, Praveer Singh, Jayashree Kalpathy-Cramer, and Daniel L. Rubin. Splitavg: A heterogeneity-aware federated deep learning method for medical imaging. *IEEE Journal of Biomedical and Health Informatics*, 26(9):4635–4644, 2022. 1

[66] Tuo Zhang, Lei Gao, Sunwoo Lee, Mi Zhang, and Salman Avestimehr. Timelyfl: Heterogeneity-aware asynchronous federated learning with adaptive partial training. In *CVPR Workshops*, pages 5064–5073, 2023. 8

[67] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018. 1

[68] Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-free knowledge distillation for heterogeneous federated learning. In *ICML*, pages 12878–12889. PMLR, 2021. 1