Towards Energy-Aware Federated Learning on Battery-Powered Clients

Amna Arouj[*]
Queen Mary University of London
United Kingdom

Ahmed M. Abdelmoniem[†]
Queen Mary University of London
United Kingdom

Abstract

Federated learning (FL) is a newly emerged branch of AI that facilitates edge devices to collaboratively train a global machine learning model without centralizing data and with privacy by default. However, despite the remarkable advancement, this paradigm comes with various challenges. Specifically, in large-scale deployments, client heterogeneity is the norm which impacts training quality such as accuracy, fairness, and time. Moreover, energy consumption across these battery-constrained devices is largely unexplored and a limitation for wide-adoption of FL. To address this issue, we develop EAFL, an energy-aware FL selection method that considers energy consumption to maximize the participation of heterogeneous target devices. *EAFL* is a power-aware training algorithm that cherry-picks clients with higher battery levels in conjunction with its ability to maximize the system efficiency. Our design jointly minimizes the time-to-accuracy and maximizes the remaining on-device battery levels. *EAFL* improves the testing model accuracy by up to 85% and decreases the drop-out of clients by up to 2.45×.[1]

Keywords
Federated Learning, Heterogeneity, Energy, Efficiency

ACM Reference Format:
Amna Arouj and Ahmed M. Abdelmoniem. 2022. Towards Energy-Aware Federated Learning on Battery-Powered Clients. In *1st ACM Workshop on Data Privacy and Federated Learning Technologies for Mobile Edge Network (FedEdge '22), October 17, 2022, Sydney, NSW, Australia.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3556557.3557952

[*]Work done during an internship at Queen Mary University of London.
[†]Corresponding author (ahmed.sayed@qmul.ac.uk)
[1]Code and scripts are available at https://github.com/SAYED-Sys-Lab/EAFL

1 Introduction

Due to the advancements in technology and the wireless industry's growth, a wealth of data is born at the edge every day. We are connecting more devices and crunching data faster than ever before and by 2025 the number of smartphones and wearable devices will reach 7.33 and 1 billion, respectively [13]. To draw useful information from this geographically distributed data, a prominent paradigm known as federated learning (FL) has emerged that allows the end devices to learn a shared ML model while preserving the user's data privacy. Ideally, a cluster of users run stochastic gradient descent (SGD) locally and aggregate their updated models via the server to obtain a new global model.

Lately, the advancements in the powerful mobile System-on-Chips (SoCs) further foster the adoption of FL for collaborative learning on edge devices such as smartphones, smart-wear, IoT devices, etc. These devices are now equipped with high-performance central processing units (CPUs) and graphics processing units (GPUs) to operate intensive computations for AI/ML-based applications. Due to the technical abilities and implementation ease, FL has recently seen wins in several applications like power keyboard predictions(Gboard), vocal/face classifiers (Face ID and Siri), virtual assistants, smart cities and augmented reality [19].

Although on-device inference comes with promising deployment in exciting applications, improved latency, work offline, privacy advantages, and better battery life, however, implementing FL on mobile devices brings severe challenges, of which energy consumption is a primary concern. Common characteristics of these devices are that they have limited energy, storage, and computing resources; thus, because of these constraints, optimizing the energy efficiency of the ML inference while fulfilling the Quality-of-Service (QoS) requirements is essential for these services.

Although existing FL solutions have shown significant progress in overcoming the challenges encountered in the FL design space, most of them, however, given a pool of participants, focus either on optimizing *statistical model efficiency* (i.e., improving training accuracy with lesser training rounds) [22, 28] or *system efficiency* (i.e., shorter training rounds) [24] while other mechanisms have been proposed to guarantee privacy and robustness [12]. As adopted in Oort, the designer makes all efforts to optimize the system efficiency,

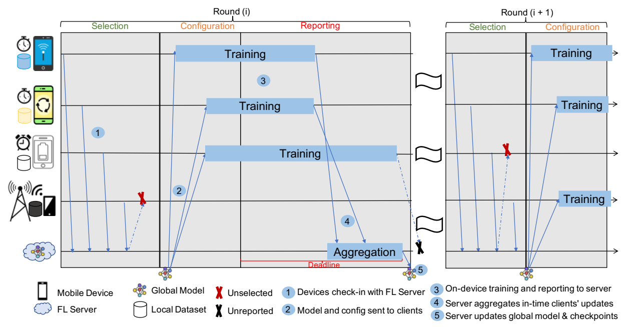


Figure 1: The various phases in training rounds of FL.

ignoring the diversity of the clients' data [21]. This level of unfairness results in a less robust model toward data heterogeneity and, therefore, results in
Executing intensive on-device computation for long periods can quickly drain the battery or burn the device, leading to client dropouts [21]. Hence, we deem *a power-aware FL training is an open problem.* In this work, we introduce *EAFL*, a novel, user-friendly training algorithm that picks high remaining power learners to increase the participation level in FL training and reduce the energy impact on user's devices. Our results show that *EAFL* delivers significant accuracy benefits over the state-of-the-art while reducing drop-outs due to battery drainage, hence preserving the user experience.
In this work, we make the following contributions:

(1) We consider a more practical FL scenario on battery-powered devices where heavy computations during FL training may lead to performance degradation while accounting for the heterogeneity in the system.
(2) Based on studying various energy-consumption models of real mobile devices, we present a power-aware design that intelligently improves the FL performance in battery-powered scenarios through reduced client dropouts and increased participation levels.
(3) We show, via experiments on real FL benchmark in battery-powered scenario, that *EAFL* produces models of high quality compared to state-of-art solutions.

2 Background and Motivation

We start with a quick overview of the FL system design, followed by highlighting the key limitations in the existing solutions that motivate our work.

2.1 Federated learning

Federated learning, an emerging networking paradigm, runs machine learning on decentralized data. It takes advantage of the available computing resources across a massive pool of edge devices. More specifically, all clients collectively train on one global learning model and perform regular model parameter updates by continuous interactions.
Two main entities are in the core of FL design —*Clients*, the data owners (e.g. smartphones, tablets, laptops, autovehicles, etc.) and — *Aggregator*, the model owner (e.g. server). Training of the global model takes place by specifying the number of epochs **E** (rounds) until the model converges. Other parameters like minibatch size **B**, and participants per round **K** are also specified and are determined by the FL-based services [8, 19]. As depicted in Fig. 1, at the start of each training round, the aggregator selects **K** participants among **N** available devices depending on given criteria. *(Step 1)* Server broadcasts the model's current version and other necessary hyper-parameters to the selected devices. *(Step 2)* Each participant trains the model by performing **E** local optimization steps with a batch size of **B**. *(Step 3)* Learners send their computed model updates back to the server.*(Step 4)*The server collects the model updates from the participants for aggregation and checkpoints it. *(Step 5)* The stages are repeated until the desired accuracy is achieved.

2.2 Motivation

In this part, we try to motivate our work.
FL considerations in mobile environment: Existing FL solutions have simply failed to notice the consequences resulting in the interplay of client devices and training speed (e.g., using significant local steps to save communication) [24] leads to intensive on-device computation for extended periods, resulting in the sudden drain of the battery or even burning of the device hence leading to unavailability of the client. However, the possibility of redeeming these disadvantages by implementing a *power-aware training algorithm* has been incredibly overlooked [21].
Furthermore, very less work is presented in the *energy efficiency* optimization domain. Most prior work supposes that FL training is activated once smartphones are plugged into a power-point because of the significant energy consumption during FL training [29, 34]. Unfortunately, this has restricted the practical implementation of FL, resulting in incorrect model accuracy and long convergence times. Energy-efficient federated learning can facilitate on-device training with better accuracy, model quality, and user experience. Through *EAFL*, we aim to present solutions to simultaneously achieve system and energy conservation, embracing the opportunities presented by earlier work.
System heterogeneity and client drop-outs: Edge and IoT devices differ in underlying architectures (CPU, design, or memory). These devices function on different battery levels and use different communication mediums (e.g., Wi-Fi or 4G/5G). Hence, each device differs in computational storage and communication capacities. These differences in system configurations introduce problems to FL schemes especially when entry-level edge devices with low battery power and bandwidth are involved. This is because these devices are more susceptible to dropouts during training when they run out of battery at any time.

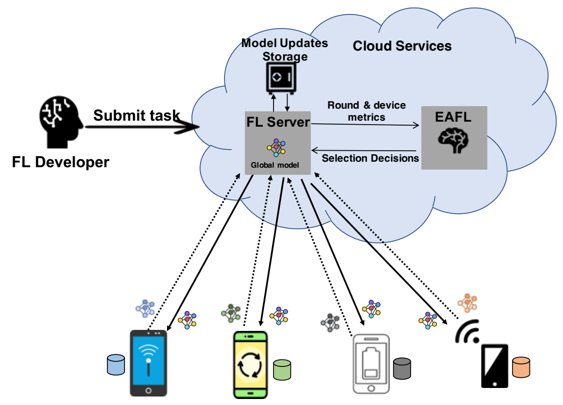


Figure 2: High-level Architectural Design of *EAFL*.

Client drop-out is synonymous with the straggler issue in FL training, in which some clients delay the uploading of the local model [24]. Many solutions exist to mitigate the straggler issue [31, 37]. A straggler-resilient design is proposed in [31] that adaptively selects clients by incorporating the statistical characteristics of the client's data. However, client drop-out is a more severe case because dropout clients cannot upload their model in the current round and are likely to remain unavailable for some period of time; thus, existing schemes to mitigate the straggler issue do not work [35].

For the aforementioned reasons, system heterogeneity introduces unexplored challenges in FL. We introduce the notion of power-aware training which mitigates client drop-out and improves FL performance. Our design goal is to trade-off remaining power with time which reduces the clients' dropouts and hence improves the model quality.

3 EAFL Design Overview

EAFL's goal is to jointly reconcile the demand for power-aware FL training and optimization of system/statistical efficiency over heterogeneous battery-powered edge devices. To illustrate and study the heterogeneity impact on model quality and client drop-outs, we follow the experimental design approach [14] and raise the following questions:

(1) To what extent do client dropouts due to battery-constraints impact the convergence of FL?
(2) What is the trade-off between model quality and energy efficiency of the system?

3.1 Architecture

Fig. 2 shows the high-level architecture of *EAFL* and the interactions between *EAFL*, the FL developer and server and the edge devices. The developer submits the task to the FL coordinator, and then the coordinator registers each client's profile (e.g., battery level, workload, RAM, etc.) and forwards the characteristics to the server running *EAFL*. Based on the feedback from the earlier rounds and client profiles, *EAFL* associates a utility with each client and selects the group of participants for the upcoming training round according to the selection mechanism. Next, the coordinator distributes the relevant model profiles to each selected participant, who computes results on their local data individually. The coordinator then collects the updates from the participants to aggregate. Furthermore, the steps from selection to aggregation are repeated until the desired accuracy is achieved.

In *EAFL*, the algorithm identifies and favours participants with higher utilities. The reward function calculates the utility of each client which consists of two parts. One part is a function for jointly measuring the system and statistical utility and the other part is a function of the remaining battery level of the client's device. We test our scheme under different scenarios by giving different weights to each function in the utility definition. We elaborate more on client utility in Section 4. In our scheme, we preferentially select clients with higher power values by controlling the weights in the utility function. We use different models for the power consumption based on devices' system configuration which accounts for idle and normal usage states of the devices.

4 Federated Model Training

In this section, we explain how *EAFL* quantifies the client utility, how it selects high-utility clients as training unwinds, and then discuss the energy consumption models.

4.1 Clients utility definition

The reward function comprises two parts, seeking to reconcile the demand for time-to-accuracy and energy conservation. We modify the current reward function proposed in Oort [21] by replacing it for client i as:

$$reward = f \times Util(i) + (1 - f) \times power(i), \tag{1}$$

where, $f \in [0, 1]$ and Eq. (1) will naturally give high-priority to the high-power clients as $f \longrightarrow 0$. For example, to prioritize the clients with high-battery levels we use:

$$power(i) = cur\_battery\_level(i) - battery\_used(i)$$

Trade-off between system and statistical efficiency: the first part of the reward function of Eq. (1) influences the time-to-accuracy metric of the FL process. It relies on two factors: (i) **system efficiency**: completion time of each training round, and (ii) **statistical efficiency**: the number of rounds completed to reach the desired accuracy. Both forms of efficiencies are considered jointly for better time-to-accuracy performance. Oort [21] finds a sweet spot in the trade-off by attaching a utility with every client that optimizes each form of efficiency. In Oort, a high statistical utility may lead to longer rounds especially if the client becomes a bottleneck for the aggregation round; moreover, on the other hand, a high system utility may reduce each round's duration and can lead to more rounds as the fastest clients are exclusively picked who become over-represented. The utility of each

Table 1: Comm. energy consumption (y) given duration (x)

	Download	Upload
WIFI	y = 18.09x + 0.17	y = 21.24x - 2.68
3G	y = 20.59x - 1.09	y = 15.31x + 2.67

client i is formulated by a utility calculated after each training round and is given by [21]:

$$Util(i) = |B_i|\sqrt{\frac{1}{|B_i|}\sum_{kEB_i} Loss(k^2)} \times (\frac{T}{t_i})^{1(T<t_i)\times a} \quad (2)$$

where, in Eq. (2), T is the duration of each round, t_i is the time taken by the client to process the training, and $1(x)$ is an indicator function that is 1 if x is true and 0 otherwise. More details are given in [21].

4.2 Energy consumption model

The energy consumption in the second part of Eq. (1) comes from the local computations executed during training and the wireless transmissions of the model updates.

Computation: The energy consumed by a selected device for a local iteration can be determined by taking the product of execution time and the run-time power such as $E_{comp} = P \times t$ where t is the time spent in the training on CPU/GPU and P are the power consumption at average usage during the training. Here, we assume that the mobile devices are equipped with GPUs and hence we inherit the GPU power model as in [10]. The run-time power in the busy state is calculated for each category of edge devices (high-end, mid-end or low-end devices) and is dependent on the device-specific parameters (detail of the device categories and specifications is mentioned in Section 5).

Communication: The mobile devices participating in a particular round of the FL training transmit their model aggregates to the server via wireless transmissions, corresponding to the communication energy consumption. We follow a simple linear energy model proposed in [18]. Table 1 shows the energy consumption functions for the elapsed time (communication latency) while using WiFi or 3G communication technologies to upload and download data. These functions compute the percentage of battery consumed by smartphones (y) when using the respective mediums for x hours. Their measurements were conducted on HTC Desire HD smartphone running Android OS version 2.3 [18].

5 Evaluation

We evaluate *EAFL* against Oort and Random selection.

Experimental Setup In the experiments, we simulate an FL benchmark for speech recognition task that uses Google Speech dataset [36] and ResNet model [17]. Learners are assigned real-world devices and network capability profiles from the AI Benchmark [7] and MobiPerf Trace [23], respectively. This is an event-driven simulation with time calculated based on the completion time of the learners. To train the model, we use a cluster of GPU servers and interleave the time between individual learners. We assign 4 GPUs per experiment and use Pytorch as the training backend and YoGi as the aggregation algorithm [30]. The hyper-parameters are set to 0.05, 500, and 20 for learning rate, number of epochs, and batch size, respectively. All clients are always available but in each round the target number of learners to be selected is 10. For the selected devices, we calculate energy consumption during training using the power model Eq. (1); however, for unselected devices, we reduce the energy consumed for being in a combination of idle or busy states.

Table 2: Mobile device specification

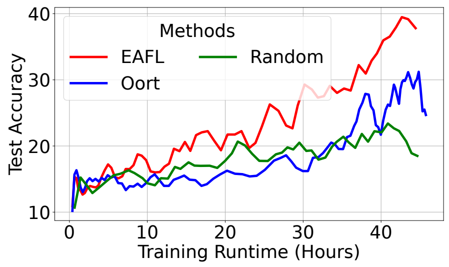
Device	Average Power (W)	Perf/W	Memory	Battery Capacity
Huawei Mate 10 (Kirin 970) (High-end)	6.33	5.94 fps/W	4GB(RAM)	4000mAh
Nexus 6P (Snapdragon 810 v2.1) (Mid-range)	5.44	4.03 fps/W	3GB(RAM)	3450mAh
Huawei P9 (Kirin 955) (Low-end)	2.98	3.55 fps/W	3GB(RAM)	3000mAh

Data Partitioning: The client to data mappings used in Oort [21] is close to an IID distribution, so we introduce a more realistic non-IID distribution. The learners are assigned data samples from a random 10% of the labels (4 out of 35) while the data points per learner are sampled uniformly.

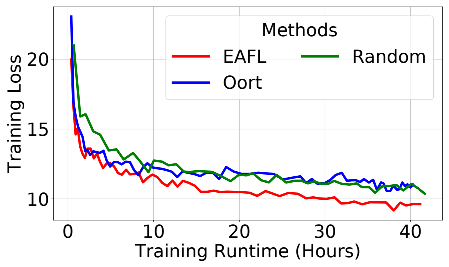
Device Profiles: We evaluate each device's communication and computation profiles using the real device measurements from AI benchmark [7] and MobiPerf [23] benchmark. However, for energy consumption, there is no such information widely available hence we cluster these profiles into three main performance categories (high, mid, and low) and map each device to one of them. We select three smartphones to represent the high, mid and low-end categories. Their average power consumption measurements from GFXBench [16] along with device specifications are shown in Table 2.

Experimental Results: To evaluate the performance of *EAFL*, we compare it with Oort and a random sampler (Random). We run the experiments in non-iid settings. We use the value of $f = 0.25$ in Eq. (1) to calculate the reward function for *EAFL* to give more weight to the high-power clients (i.e., higher remaining battery). This allows more clients to stay and reduces dropouts due to running out of battery.

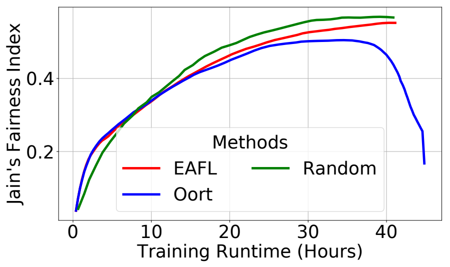
As shown Fig. 4a, compared to other methods, Oort shows a significant increase in battery run-outs as the training progresses, highlighting the impact of not being energy-aware unlike *EAFL* and random with low drop-outs. *EAFL*, to conserve energy, it trades-off time and selects high-power clients and hence sees fewer drop-outs over the training. Random has low dropouts as it selects clients uniformly at random because it distributes the consumption over more clients, however, its consumption supersedes *EAFL* after 40 hours of training. Fig. 4b shows Random results in significantly higher round duration compared to the other methods. We note the per-round duration for Oort and *EAFL* is almost the same, but further in time, the duration increases to compensate



(a) Test accuracy vs time

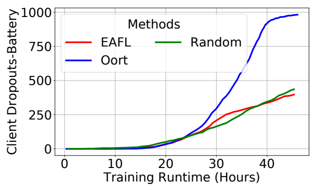


(b) Training loss vs time

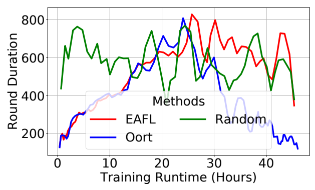


(c) Jain's fairness index vs time

Figure 3: Performance of Oort and *EAFL* in terms of (a) test accuracy, (b) train loss, (c) Jain's fairness index in a Non-IID case.



(a) Clients running out of battery



(b) Round duration

Figure 4: Comparison of client dropouts due to running out of Battery between Oort and *EAFL*.

for low statistical efficiency (Oort) or high energy consumption (*EAFL*). Oort experiences a low participation rate and round failures due to significant numbers of dropouts which explains its low-round duration at the end. *EAFL* is able to maintain low client drop-outs due to its energy-aware selection which increases the participation rate in the training. Fig. 3a and Fig. 3b present the achieved accuracy and training loss for all methods, respectively. *EAFL* achieves the best results in both metrics showing the benefits of its selection method (esp. for battery-powered scenarios) which maintains high levels of participation in the training. Fig. 3c shows Jain's fairness index for device selection which measures if users are getting a fair opportunity to participate in the training. The results suggest that *EAFL* enjoys high levels of fairness similar to Random. Oort initially enjoys the same levels of fairness but then due to high dropouts and low participation, its fairness degrades severely.

6 Related Work

Federated Learning (FL): is a new distributed machine learning method which is increasingly becoming popular for its privacy-preservation and low-communication features. This motivated the growing adoption of FL to improve the end-user experience (e.g., the search suggestion quality of virtual keyboards [39]). Moreover, to encourage and speed up experimentation of new ideas, many FL frameworks were recently developed [9, 32]. In FL, training a global model is assigned to a sub-population of decentralized devices such as mobile or IoT devices. These devices possess private data and engage in training the model on their local data [8, 24].

System heterogeneity: One of the major contributors to system performance unpredictability is the heterogeneity inherent in many of the distributed systems. Mainly, in FL context, the heterogeneity of devices' system configurations (e.g., computation, communication, battery, etc) results in unpredictable performance. For instance, the stragglers (i.e., slow workers) can halt the training process for a prolonged duration [2, 22]. Several solutions exist that address this problem through system and algorithmic solutions [2, 22]. Moreover, in FL, the heterogeneity is also a byproduct of other artifacts other than the devices. For example, the learner data distributions, the participants' selection method, and the behaviour of the device's owner are common sources of heterogeneity in FL setting [4, 5, 8].

Energy-conservation: Considering the uncertainties in the mobile environment, several energy management techniques have been proposed [10]. Other works for energy efficiency optimization in the mobile environment use computation offloading techniques. Finally, few works focused on energy-efficient FL training [19]. *EAFL* aims for energy conservation to mitigate client dropouts which is the major contributor to the degraded FL model qualities.

Improvements in FL: In FL, several works aim to improve the time-to-accuracy of training by leveraging techniques such as periodic updates, compression, and layer-wise asynchronous updates [1, 3, 6, 8, 11, 15, 20, 33, 38]. Other proposals aimed at righting the privacy guarantees of FL settings [8, 25, 27]. Moreover, the bias in FL is studied to ensure fair participation in the training process [22, 26].

7 Conclusion

We study a practical federated learning scenario where the participants are battery-powered and collaboratively learn a new global model. We find that the existing state-of-the-art methods, aiming to minimize the time-to-accuracy, fail to achieve satisfactory performance as they ignore the factor of energy consumption on the devices. Therefore, we present *EAFL*, to enable power-aware FL training on battery-powered devices. Our algorithm intelligently selects the participants to maintain low time-to-accuracy while conserving energy. It shows up to 85% improvement in model accuracy and 2.45× decrease in drop-out of clients proving to be a practical FL solution for battery-powered scenarios.

References

[1] Ahmed M. Abdelmoniem and Marco Canini. 2021. DC2: Delay-aware Compression Control for Distributed Machine Learning. In *IEEE INFOCOM.*

[2] Ahmed M. Abdelmoniem and Marco Canini. 2021. Towards Mitigating Device Heterogeneity in Federated Learning via Adaptive Model Quantization. In *ACM EuroMLSys.*

[3] Ahmed M. Abdelmoniem, Ahmed Elzanaty, Mohamed-Slim Alouini, and Marco Canini. 2021. An Efficient Statistical-based Gradient Compression Technique for Distributed Training Systems. In *MLSys.*

[4] Ahmed M. Abdelmoniem, Chen-Yu Ho, Pantelis Papageorgiou, Muhammad Bilal, and Marco Canini. 2021. On the Impact of Device and Behavioral Heterogeneity in FederatedLearning. *arXiv 2102.07500* (2021).

[5] Ahmed M. Abdelmoniem, Chen-Yu Ho, Pantelis Papageorgiou, and Marco Canini. 2022. Empirical Analysis of Federated Learning in Heterogeneous Environments. In *ACM EuroMLSys.*

[6] Ahmed M. Abdelmoniem, Atal Narayan Sahu, Marco Canini, and Suhaib A. Fahmy. 2021. Resource-Efficient Federated Learning. *arXiv 2111.01108* (2021).

[7] AI Benchmark. 2021. Performance Ranking. https://ai-benchmark.com/ranking.html

[8] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konečný, Stefano Mazzocchi, H. Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. 2019. Towards Federated Learning at Scale: System Design. In *MLSys.*

[9] Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečný, H. Brendan McMahan, Virginia Smith, and Ameet Talwalkar. 2018. LEAF: A Benchmark for Federated Settings. *arXiv 1812.01097* (2018).

[10] Ning Ding and Y. Charlie Hu. 2017. GfxDoctor: A Holistic Graphics Energy Profiler for Mobile Devices. In *Proceedings of the European Conference on Computer Systems (EuroSys).*

[11] Aritra Dutta, El Houcine Bergou, Ahmed M. Abdelmoniem, Chen-Yu Ho, Atal Narayan Sahu, Marco Canini, and Panos Kalnis. 2020. On the Discrepancy between the Theoretical Analysis and Practical Implementations of Compressed Communication for Distributed Deep Learning. In *The AAAI Conference on Artificial Intelligence.*

[12] Keith Bonawitz et al. 2017. Practical Secure Aggregation for Privacy-Preserving Machine Learning. In *In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS).*

[13] Chip 1 Exchange. 2022. The Wave of Wearables.

[14] Ronald Fisher. 1971. *The Design of Experiments* (9 ed.). Macmillan.

[15] Rishikesh R. Gajjala, Shashwat Banchhor, Ahmed M. Abdelmoniem, Aritra Dutta, Marco Canini, and Panos Kalnis. 2020. Huffman Coding Based Encoding Techniques for Fast Distributed Deep Learning. In *Workshop on Distributed Machine Learning - CoNext (DistributedML).*

[16] GFXBench. 2022. GFXBench 5.0. https://gfxbench.com/result.jsp.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *CVPR.*

[18] Goran Kalic, Iva Bojic, and Mario Kusek. 2012. Energy consumption in android phones when using wireless communication technologies. In *2012 Proceedings of the 35th International Convention MIPRO.*

[19] Young Geun Kim and Carole-Jean Wu. 2021. AutoFL: Enabling Heterogeneity-Aware Energy Efficient Federated Learning. *ArXiv 2107.08147* (2021).

[20] Jakub Konečný, H. Brendan McMahan, Felix X. Yu, Peter Richtarik, Ananda Theertha Suresh, and Dave Bacon. 2016. Federated Learning: Strategies for Improving Communication Efficiency. In *Workshop on Private Multi-Party Machine Learning - NeurIPS.*

[21] Fan Lai, Xiangfeng Zhu, Harsha V. Madhyastha, and Mosharaf Chowdhury. 2021. Efficient Federated Learning via Guided Participant Selection. In *USENIX OSDI.*

[22] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated Optimization in Heterogeneous Networks. In *MLSys.*

[23] M-Lab. 2021. MobiPerf: an open source application for measuring network performance on mobile platforms. https://www.measurementlab.net/tests/mobiperf/

[24] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *AISTATS.*

[25] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. 2019. Exploiting Unintended Feature Leakage in Collaborative Learning. In *IEEE Symposium on Security and Privacy (SP).*

[26] Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. 2019. Agnostic Federated Learning. In *ICML.*

[27] Milad Nasr, Reza Shokri, and Amir Houmansadr. 2019. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *IEEE Symposium on Security and Privacy (SP).*

[28] Xingchao Peng, Zijun Huang, Yizhe Zhu, and Kate Saenko. 2020. Federated Adversarial Domain Adaptation. In *International Conference on Learning Representations (ICLR).*

[29] Xinchi Qiu, Titouan Parcollet, Daniel J. Beutel, Taner Topal, Akhil Mathur, and Nicholas D. Lane. 2020. A first look into the carbon footprint of federated learning. *ArXiv 2010.06537* (2020).

[30] Swaroop Ramaswamy, Om Thakkar, Rajiv Mathews, Galen Andrew, H. Brendan McMahan, and Françoise Beaufays. 2020. Training Production Language Models without Memorizing User Data. *arXiv 2009.10031* (2020).

[31] Amirhossein Reisizadeh, Isidoros Tziotis, Hamed Hassani, Aryan Mokhtari, and Ramtin Pedarsani. 2021. Straggler-Resilient Federated Learning: Leveraging the Interplay Between Statistical Accuracy and System Heterogeneity. In *International Workshop on Federated Learning - ICML.*

[32] Theo Ryffel, Andrew Trask, Morten Dahl, Bobby Wagner, Jason Mancuso, Daniel Rueckert, and Jonathan Passerat-Palmbach. 2018. A generic framework for privacy preserving deep learning. *arXiv 1811.04017* (2018).

[33] Atal Sahu, Aritra Dutta, Ahmed M. Abdelmoniem, Trambak Banerjee, Marco Canini, and Panos Kalnis. 2021. Rethinking gradient sparsification as total error minimization. In *NeurIPS.*

[34] C. Wang, Y. Yang, and P. Zhou. 2021. Towards Efficient Scheduling of Federated Mobile Devices Under Computational and Statistical Heterogeneity. *IEEE Transactions on Parallel & Distributed Systems* 32, 02 (2021), 394–410.

[35] Heqiang Wang and Jie Xu. 2021. Friends to Help: Saving Federated Learning from Client Dropout. *ArXiv 2205.13222* (2021).

[36] P. Warden. 2018. Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition. *ArXiv 1804.03209* (2018).

[37] Cong Xie, Sanmi Koyejo, and Indranil Gupta. 2020. Asynchronous Federated Optimization. In *Annual Workshop on Optimization for Machine Learning - NeurIPS.*

[38] Hang Xu, Chen-Yu Ho, Ahmed M. Abdelmoniem, Aritra Dutta, El Houcine Bergou, Konstantinos Karatsenidis, Marco Canini, and Panos Kalnis. 2021. GRACE: A Compressed Communication Framework for Distributed Machine Learning. In *IEEE ICDCS.*

[39] Timothy Yang, Galen Andrew, Hubert Eichner, Haicheng Sun, Wei Li, Nicholas Kong, Daniel Ramage, and Françoise Beaufays. 2018. Applied Federated Learning: Improving Google Keyboard Query Suggestions. *arXiv 1812.02903* (2018).