FLAME: Federated Learning Across Multi-device Environments

HYUNSUNG CHO*, Carnegie Mellon University, USA
AKHIL MATHUR, Nokia Bell Labs, UK
FAHIM KAWSAR, Nokia Bell Labs, UK

Federated Learning (FL) enables distributed training of machine learning models while keeping personal data on user devices private. While we witness increasing applications of FL in the area of mobile sensing, such as human-activity recognition, FL has not been studied in the context of a *multi-device environment* (MDE), wherein each user owns multiple data-producing devices. With the proliferation of mobile and wearable devices, MDEs are increasingly becoming popular in ubicomp settings, therefore necessitating the study of FL in them. FL in MDEs is characterized by high non-IID-ness across clients, complicated by the presence of both user and device heterogeneities. Further, ensuring efficient utilization of system resources on FL clients in a MDE remains an important challenge. In this paper, we propose FLAME, a user-centered FL training approach to counter statistical and system heterogeneity in MDEs, and bring consistency in inference performance across devices. FLAME features (i) user-centered FL training utilizing the time alignment across devices from the same user; (ii) accuracy- and efficiency-aware device selection; and (iii) model personalization to devices. We also present an FL evaluation testbed with realistic energy drain and network bandwidth profiles, and a novel class-based data partitioning scheme to extend existing HAR datasets to a federated setup. Our experiment results on three multi-device HAR datasets show that FLAME outperforms various baselines by 4.8-33.8% higher F_1 score, 1.02-2.86× greater energy efficiency, and up to 2.02× speedup in convergence to target accuracy through fair distribution of the FL workload.

CCS Concepts: • **Human-centered computing → Ubiquitous and mobile computing systems and tools**; • **Computing methodologies → Cooperation and coordination**.

Additional Key Words and Phrases: Human Activity Recognition, Federated Learning

1 INTRODUCTION

Federated Learning (FL) [51, 58] enables collaborative training of machine learning models in networks of remote devices (or clients), while keeping users' personal data on the devices private. At a high level, FL involves repeating three steps: (i) updating the parameters of a shared prediction model locally on each remote client, (ii) sending the local parameter updates to a central server for aggregation, and (iii) receiving the aggregated prediction model back on the remote client for the next round of local updates. While many of the early use-cases of FL were related to natural language processing [21], visual recognition [24, 49], and speech recognition [22] tasks, we are now also witnessing its applications in the area of human-activity recognition [17].

Prior FL works have made a common assumption that each remote user owns a *single data-producing device*, which then participates as a client in the federated training process. However, there is nowadays a trend of people simultaneously using *multiple data-producing devices* such as smartphones, smartwatches and smart earbuds to collect data about their physical activities, health, or context. It is even predicted that by 2025, each person will own 9.3 connected devices on average [61]. From a sensing perspective, multi-device environments offer exciting opportunities to develop accurate and generalizable models by leveraging the similarities and differences across devices. Although there have been prior works on training machine learning models for multi-device environments [28, 52, 70], they were primarily based on centralized training and required sharing of raw data between devices. Applying federated learning to these setting could be a potential privacy-preserving solution, however to the best of our knowledge no prior works have investigated federated learning in these settings.

*This work was done while the author was an intern at Nokia Bell Labs Cambridge.

Authors' addresses: Hyunsung Cho, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA, hyunsung@cs.cmu.edu; Akhil Mathur, Nokia Bell Labs, Cambridge, UK, akhil.mathur@nokia-bell-labs.com; Fahim Kawsar, Nokia Bell Labs, Cambridge, UK, fahim.kawsar@nokia-bell-labs.com.

(a) Single-device FL (b) Multi-device FL

Fig. 1. Architectures of single-device FL and multi-device FL. In single-device FL, each user is assumed to have a single data-producing device, whereas in multi-device FL, multiple devices observe the user's activity simultaneously.

The presence of multiple devices on a user presents the following research challenges and opportunities for federated learning:

- A multi-device environment can have N users, each owning multiple ($K \geq 1$) data-producing devices that simultaneously collect sensor data (see Figure 1b). Conventional FL systems would consider all these $N \cdot K$ devices as independent FL clients; however, this approach ignores the natural affinities in the data collected by the devices of the same user. An alternative is to consolidate the data from all K devices of each user on an edge node, treat it as a unified dataset that represent each user's behavior, and perform FL on N consolidated datasets. This approach, however, has a practical drawback as it requires sharing of raw datasets across devices, which has a significant communication cost and may raise privacy concerns. As such, we need a FL approach which respects the privacy of each device's data and at the same time considers the natural affinity between the devices of the same user. We elaborate on this challenge in §3 and propose a user-centered FL training paradigm to address it.
- Countering the statistical heterogeneity across users during training is an active area of research in FL [26, 27, 33, 80]. The multi-device problem setting presents a unique challenge that there exists statistical heterogeneity not just across user datasets, but also within a user's local dataset. This 'local' statistical heterogeneity is caused by the differences in data distributions of the various devices owned by the user. For example, motion sensors placed at different positions of the body will capture the user's motion differently, thereby making the device datasets heterogeneous. In §3, we quantify the impact of both these types of heterogeneity in multi-device FL systems and propose a heterogeneity-aware client selection approach to mitigate it.
- In addition to the statistical heterogeneity, there also exists system heterogeneity across devices, e.g., variations in computational capabilities, battery power, network communication speeds. These variations are present both across users and across devices of the same user. As system efficiency is a core success metric for a FL system, we need to strike a balance between model accuracy, convergence speed, and resource consumption on devices. In §4.2, we detail a strategy which combines both statistical and systems utility of each device in a unified metric to drive the client selection in FL.
- Due to the high statistical heterogeneity in the multi-device setting, it is likely that a single global model will not generalize to all the devices, and it may even result in uneven accuracies across devices of the same user. We elaborate on this challenge in §3.3 and present a weight-regularized federated personalization approach to train accurate models tailored to each device.

The main contributions of this paper are as follows:

Extending FL to Multi-device Environments. We present Federated Learning Across Multi-device Environments (FLAME), a unified solution to solve the aforementioned challenges for FL in multi-device environments.

FLAME employs a user-centered FL training approach in combination with a device selection scheme that balances accuracy, convergence time, and energy efficiency of FL. FLAME further utilizes model personalization to counter statistical heterogeneity and inconsistency in inference performance across devices.

A Realistic Testbed for FL. Prior works on federated training of HAR models (e.g., [77]) have assumed that each FL client owns a large amount of labeled data, in the order of 3000-5000 seconds. Instead, we setup a realistic FL testbed with a large number of clients, each holding a small amount of labeled data. This is achieved through a novel class-based partitioning scheme that divides existing HAR datasets over a large number of users, and makes them suitable for realistic federated evaluations. In addition, our testbed includes latency and energy consumption profiles for nine embedded-scale hardware, which allows for obtaining a realistic estimate of the resource consumption of federated HAR algorithms. We plan to open-source the source code of our partitioning algorithm as well as the federated HAR datasets for reproducibility.

Extensive Empirical Analysis. We compare FLAME against various FL baselines in terms of inference performance, training time, and energy consumption on three multi-device HAR datasets. These datasets contain inertial sensor data for human activities ranging from locomotion tasks to activities of daily living. For a deeper analysis of FLAME, we also present sensitivity and ablation studies on it. Our results highlight the superior inference performance, energy-efficiency, and convergence rate of FLAME as compared to the baselines.

2 BACKGROUND

In this section, we provide a primer on Federated Learning (FL) and explain the factors that influence the performance of FL systems.

2.1 Key Elements of a Federated Learning System

A federated learning system consists of a central server and N remote clients, often containing labeled data samples. The central server randomly initializes a deep neural network model M^0 and sends it to a subset of the clients $C \subseteq N$ for local training. Each client $c \in C$ updates the parameters of the model M^0 independently through supervised learning, by optimizing a loss function such as categorical cross-entropy on its local dataset. In practice, the local training is done by iterating over the labeled dataset E times, where E denotes the number of *local epochs*. After the local training finishes, we obtain a trained model M_c^0 on each client c. These local models are sent to the central server, where the model parameters are averaged using federating averaging algorithms such as FedAvg [8] to obtain the new global model M^1. This entire process of local training and federated averaging is repeated for R rounds to obtain the final global model M^R.

There are three key evaluation metrics for a FL system: firstly, we would like the model M^R to have high **test accuracy** on each client. Secondly, we would like the training to be **time-efficient**, both in terms of the number of rounds taken to convergence and the overall wall-clock time of training. Finally, we would like the training to be **energy-efficient**, in that it should minimize the battery drain due to local training on remote clients.

Below we describe two important factors that influence the performance of an FL system.

- **IID-ness of client datasets.** The *test accuracy* of a model learned using FL is adversely impacted if the datasets on the remote clients are not independent and identically distributed (IID) [80, 81]. The fundamental reason for this performance degradation is that when the client datasets are non-IID or heterogeneous, the local models trained on the clients may diverge, despite having the same initial parameters. This parameter divergence in local models, in turn, makes the global model obtained after federated averaging sub-optimal, thereby worsening its test accuracy.
- **Clients selected in each round.** In each round of FL, the server samples $C \subseteq N$ clients for local training. The choice of sampled clients can impact FL performance in three ways: (a) the data heterogeneity across

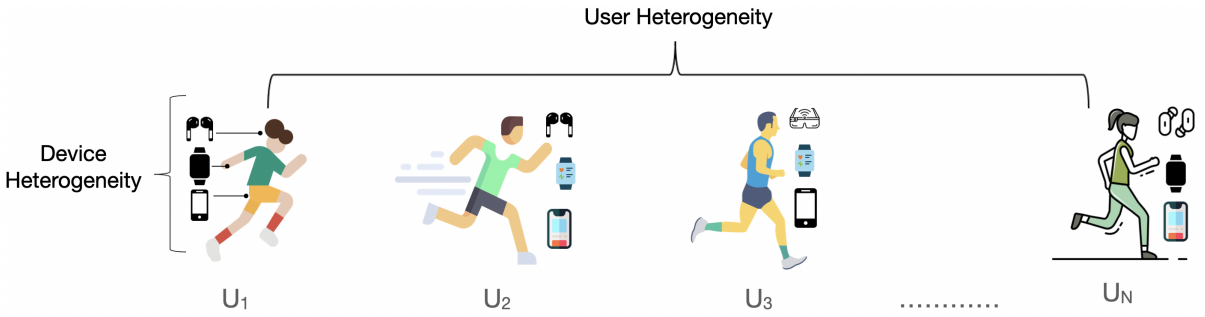


Fig. 2. Illustration of user and device heterogeneities in a multi-device FL system. $U_1 \cdots U_N$ are different users, each wearing multiple heterogeneous data-producing devices.

sampled clients can influence the global model's *accuracy*, (b) the communication bandwidth and computational resources of the sampled clients have a direct influence on how fast the local model is trained and communicated back to the global server. This in turn will impact the *training time* of FL, and (c) excessive local training on a client can lead to severe *battery drain*, thus leading to failure cases in selected clients and user experience issues such as devices running out of battery power.

Addressing the challenges of non-IID-ness or statistical heterogeneity [27, 43, 80] and achieving higher system efficiency through better client selection [11, 38, 44] are active topics of research in FL.

3 FEDERATED LEARNING IN MULTI-DEVICE ENVIRONMENTS

Building on the primer presented in §2.1, we now contextualize FL in a multi-device environment and aim to highlight the challenges and research opportunities for FL in this scenario. First, we define and state our assumptions about a multi-device environment.

Definition 3.1. (**Multi-Device Environment (MDE)).** In this setup, there exist *multiple* sensor devices that capture a physical phenomenon, e.g., a user's locomotion activity, simultaneously. In the context of human-activity recognition, an example of MDE is when a person wears multiple inertial sensing devices on their body [23] as shown in Figure 1b. These multiple devices observe the user's activity or context *simultaneously* and record sensor data in a *time-aligned manner* [28, 70]. In contrast, conventional federated learning setups assume that each user (or client) has a single data-producing device as shown in Figure 1a.

Below we present three key challenges for federated learning in the MDE setup.

3.1 Device and User Heterogeneity Lead to Higher Non-IID-ness

The MDE setting presents a challenging case of non-IID-ness in client datasets because of the presence of two types of data heterogeneities as illustrated in Figure 2:

- *User heterogeneity* occurs due to differences in personal characteristics of the users, such as different running styles and gait variations [69, 74]. As federated learning, by definition, aims to leverage the data from multiple users to learn a prediction model, this form of heterogeneity is expected across FL clients.
- *Device heterogeneity.* In addition to differences across users, the MDE setup also exhibits heterogeneity due to the multiple devices owned by the users. This heterogeneity comes from the differences in hardware and software components of the multiple devices worn by the user as well as their positions on the user's body. For example, devices with inertial sensors can be placed on the wrist (smartwatch), in the ear (smart earbud), or

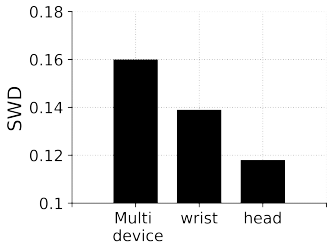


Fig. 3. Comparison of mean Sliced Wasserstein Distance (SWD) in a multi-device FL setup with two single-device FL setups. A higher mean SWD indicates higher non-IID-ness of the FL setup. In the multi-device setup, each user has multiple devices that act as FL clients. The single-device setup exemplifies a conventional FL setup wherein each user has just one device (i.e., wrist-worn IMU or head-worn IMU).

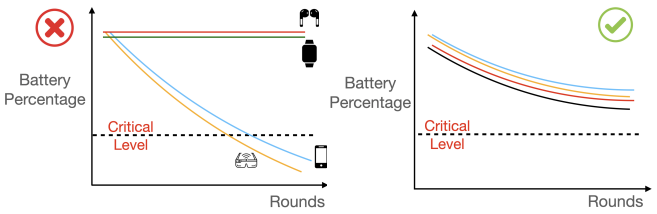


Fig. 4. A conceptual illustration to depict the potential impact on local devices if client selection does not balance energy and accuracy. We show the state of four devices owned by a user over the FL training process. (a) Only two of the four devices are ever selected for training as they result in higher model accuracy. This leads to a situation that both these devices get over-sampled, resulting in a significant energy drain due to local training and their battery levels go below a critical level required for other device operations. (b) A better approach which balances the energy consumption of each device during the training process and ensures that none of the devices go below the critical level.

inside a trouser pocket (smartphone). Prior works have shown that device heterogeneity can lead to distribution shifts in HAR data [10, 67]. Critically, *device heterogeneity* can be present even within a user's own data when a user wears multiple sensor-enabled devices while performing an activity.

To illustrate the effect of these heterogeneities on the non-IID-ness of a FL setup, we present a quantification experiment on the RealWorld HAR dataset [68]. This dataset consists of physical activity data collected from 3-axis accelerometer and 3-axis gyroscope sensors by N(=15) users, while they are wearing K(=7) IMU-enabled devices placed at thigh, waist, chest, head, shin, upperarm, and forearm. Specifically, we compare two scenarios: (a) when each of the K devices owned by the N users are considered as separate FL clients, and (b) the ideal case when each of the N users only have a single device (such as a wrist-mounted IMU or a head-mounted IMU). While (b) is a conventional FL setup with users owning a single data-producing device, the scenario (a) exemplifies the MDE setup with multiple users owning multiple devices.

We use Sliced Wasserstein Distance (SWD) [7, 57] as a metric for this quantification. SWD is a metric used to compute the distance between two multi-dimensional data distributions; a higher SWD implies a larger heterogeneity between the distributions. We compute the pairwise SWD for every pair of FL clients[1] and report their mean. A higher mean SWD of the entire setup would indicate a higher degree of non-IID-ness across clients. Figure 3 illustrates our findings. We observe that the multi-device setup has a mean SWD which is 33% and 14.2% higher than the SWDs of single device setups with head- and wrist-mounted IMUs respectively. This finding confirms that the presence of both user and device heterogeneities in the MDE setup leads to a higher degree of non-IID-ness across clients, which in turn, could degrade the performance of FL algorithms.

3.2 Balancing Accuracy, Energy and Convergence Time

In addition to model accuracy, convergence time and energy consumption are two important evaluation metrics for FL systems. The *convergence time* of FL depends on the computational capabilities of each device and the

[1]The SWD is computed over the 6-dimensional accelerometer and gyroscope datasets on each client. We segment the data in windows of 3 seconds based on prior work [10, 47]; this yields a $X_i \times 6$ dimension dataset on each client, where X_i denotes the number of windows on client i.

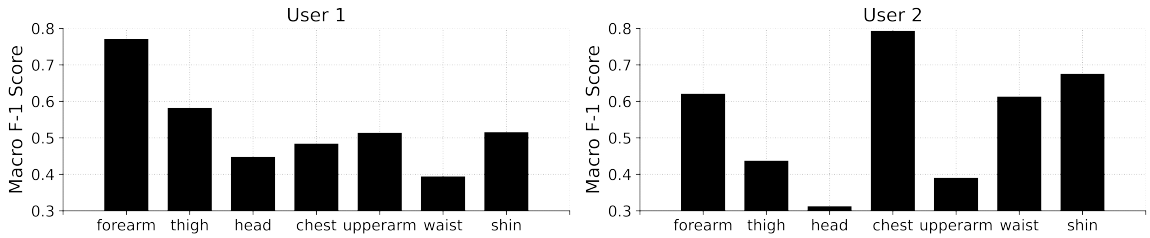


Fig. 5. Macro F_1 scores obtained using the global model trained for 100 FL rounds. We observe a high variance in model performance across the devices owned by the same user. Please refer to §5.4 for more details about the experiment setup.

communication bandwidth between the device and the server, e.g., if a device has a slow processor or a low network bandwidth, it will take more time to train the model and communicate its parameters to the server, which in turn would increase the FL convergence time. Similarly, the *energy consumption* of FL is the sum of energy consumed on each of the client devices during training. Each client's energy cost depends on the power efficiency of its processor (e.g., CPU, GPU) as well as the time taken to complete the local training over all FL rounds. During each round of federated training, we need to ensure that clients are selected in a way that achieves a good balance between model accuracy, convergence time and energy efficiency. Instead, if we optimize for only one of these metrics (e.g., accuracy) during client selection, we could run into undesirable cases as depicted in Figure 4.

3.3 Uneven Accuracy Across Devices and Users

The output of FL is typically a single global model which is obtained by averaging the local models from all the participating clients. However, in a MDE setup with various types of statistical heterogeneity present in the data, the global model may not be optimal for each participating user and device. We illustrate this challenge in Figure 5 where we plot the test accuracies obtained using the global model on devices owned by two randomly sampled users from the RealWorld HAR dataset [68]. The global model in this case is trained for 100 rounds of FL on the RealWorld dataset. We observe a high variance in the macro F_1 score obtained for different devices owned by the same user. For example, the chest-worn IMU device of user 2 achieves a test F_1 score of 0.8 while the head-worn IMU device only has a 0.31 F_1 score using the global FL model. Clearly, such variations in prediction outcomes across devices is not ideal for user experience as they may cause confusion for the user on which outcome to trust.

Takeaways: The MDE setup amplifies the challenges for FL by introducing higher non-IID-ness across clients; necessitating a careful balance between accuracy, convergence time and energy consumption; and causing significant variance in performance of different devices of the same user.

4 FLAME: FEDERATED LEARNING ACROSS MULTI-DEVICE ENVIRONMENTS

FLAME is our end-to-end FL solution comprising of a novel client selection scheme that minimizes data heterogeneity across clients in each round, and strikes a balance between inference accuracy, energy efficiency, and training time of FL. In addition, FLAME consists of a personalization module which brings consistency in the

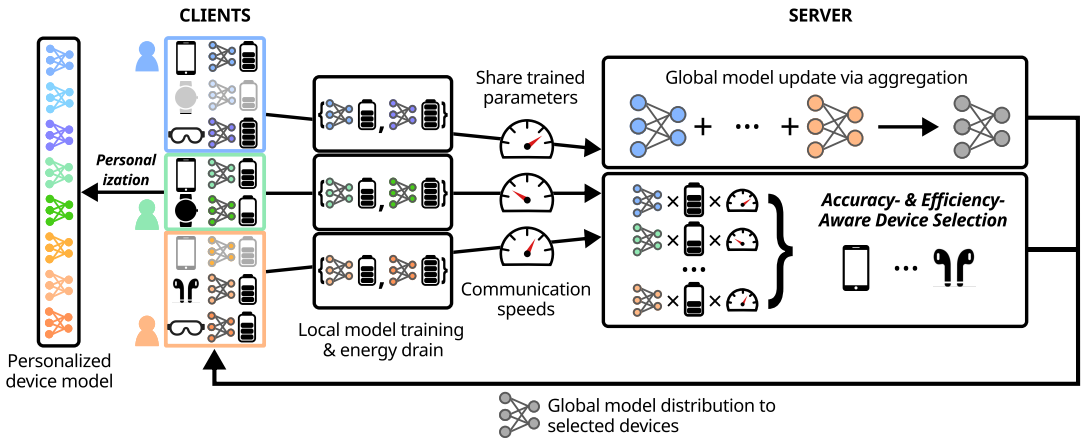


Fig. 6. System architecture and workflow of FLAME. The server distributes the global model to selected client devices. Clients perform training on their local dataset and upload model updates along with their unified utility metric (see §4.2) to the server. After the upload, clients personalize their device models using the up-to-date global model weights. Meanwhile, the server updates the global model by aggregating local updates from selected clients. It also uses the utility metrics reported by the client to perform accuracy- and efficiency-aware device selection and samples clients for the next round.

inference performance of multiple devices and users. Figure 6 presents an overview of FLAME and below we describe its main components.

4.1 User-Centered FL Training

In Figure 3, we showed that the multi-device setup has a high degree of non-IID-ness due to the presence of both user and device heterogeneities. There are two extreme choices to reduce this non-IID-ness:

(a) *Remove Device Heterogeneity*: we can select the same device from each user during a round of FL and ensure that there is no device heterogeneity across clients. For example, during Round 1, we can train the global model only on smartphone data from all users; during Round 2, we can train only on smartwatch data, and so on. This ensures that during each round of FL, only user-related heterogeneity is the source of non-IID-ness across clients.

(b) *Remove User Heterogeneity*: On the other extreme, one can consider removing user heterogeneity from each round of FL by training on each user sequentially, e.g., in Round 1, we train the global model on data from the devices of only user 1, and so on.

Both these extreme choices, however, are undesirable. In (a), the parameters of the global model can oscillate significantly between rounds due to the distribution shifts induced by the different devices used in each round and lead to poor convergence behavior. Similarly, (b) induces distribution shifts due to a drastic change in user characteristics between rounds, which could lead to poor convergence as shown in prior work [12]. In addition, (b) significantly limits the number of clients participating in each round, slowing down the training process.

Rather than completely eliminating either of these heterogeneity during training, we adopt a practical solution of minimizing their impact on the non-IID-ness of the setup. To motivate our solution, we first extend the experiment shown in Figure 3 and compare the impact of device vs. user heterogeneity on the non-IID-ness of a multi-device FL setup. Specifically, we compute the Sliced Wasserstein Distance (SWD) across clients in two scenarios: (a) When only *User Heterogeneity* is present across clients, and (b) When only *Device Heterogeneity* is present across

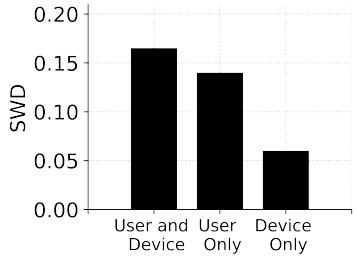


Fig. 7. Comparison of the impact of user heterogeneity vs. device heterogeneity on a multi-device FL setup. Same experiment setup as Figure 3 is used. To compute the mean user heterogeneity, we first fix a device (e.g., thigh-worn IMU) and calculate the mean pairwise SWD across all users for this device. This process is repeated for every device in the dataset to obtain the mean SWD due to user variations. To compute device heterogeneity, we fix a user (e.g., User 1) and calculate the mean pairwise SWD across all devices of this user. This process is repeated for every user in the dataset to obtain the mean SWD due to device variations.

clients. Our findings in Figure 7 show that the presence of user heterogeneity results in a higher SWD than device heterogeneity, and hence, they contribute more to the non-IID-ness of a FL setup. On the contrary, device heterogeneity alone causes less non-IID-ness because the devices in an MDE collect sensor data in a synchronized and time-aligned manner.

Based on this empirical insight, we propose a user-centered scheme for selecting FL clients in each round. Let C be the total clients we wish to sample in each round, N is the number of users in the MDE setup, K is the number of devices owned by each user, and $\rho \in [1, K]$ is a parameter that balances user and device heterogeneity in the setup. In each round of FL, we select $\frac{C}{\rho} \subseteq N$ users. From each selected user, we choose ρ number of time-aligned devices for federated training. As ρ increases, more *time-aligned* devices from each user are selected, while lesser number of unique users are sampled in a round. Both these factors contribute to reducing the overall non-IID-ness of the setup. We also note an interesting empirical finding that emerged from this design decision (see Figure 13a). It is observed that during FL, if we enforce the time-aligned devices of a user to follow the same order of iterating through their local datasets (as opposed to randomly shuffling their data), we can achieve faster convergence and higher overall accuracy.

4.2 Accuracy- and Efficiency-Aware Device Selection

Having proposed a way to reduce the non-IID-ness of the MDE setup, we now need to strike a balance between the various performance metrics of a FL system, namely accuracy, convergence time, and energy efficiency. As explained above, we wish to select C devices, spread over $\frac{C}{\rho}$ users in each round. Below we propose a sampling strategy based on the statistical, system, and time utility of each device.

Statistical utility refers to the utility of a device's data towards improving model performance. Following prior work by Lai et al. [37], we argue that a device which has a higher training loss on its local dataset should have a higher statistical utility. Our statistical utility metric for device i in round r is therefore defined as:

$$stat(i, r) = |X_i| \sqrt{\frac{1}{|X_i|} \sum_{k \in X_i} \mathcal{L}_{r-1}(k)^2}$$

where X_i is the set of locally stored training samples and $\mathcal{L}_{r-1}(k)$ is the training loss of the global model from the previous round on sample k. Intuitively, this utility prioritizes the selection of devices which currently have a

higher training loss over their local dataset, and hence could provide stronger gradients to update the global model.

System utility is a device's utility toward improving system efficiency of FL. In this paper, we use energy drain as a representative metric of system efficiency, although it can be easily substituted with other metrics depending on the system requirements. Our system utility metric for device i in round r is defined as:

$$system(i,r) = \mathbb{1}(drain^i_{r-1} < drain^i_{th}) \cdot log\left(\frac{drain^i_{th}}{drain^i_{r-1}}\right)$$

where $\mathbb{1}$ denotes the indicator function. If device i's accumulated energy drain due to federated training till the previous round ($drain^i_{r-1}$) exceeds a predefined energy drain threshold ($drain^i_{th}$), the device's system utility becomes zero. If not, devices with smaller accumulated energy drain have higher system utility. $drain^i_{th}$ is akin to an energy consumption budget for federated training and it could be predefined either by the system designer or the user. When a device runs out of its energy consumption budget, its system utility becomes 0 and it is no longer selected for training.

Time utility encourages the selection of devices that complete their local training and communicate the model parameters to the server within a predefined time threshold. At the same time, it penalizes the slower devices and discourages their selection. We define time utility as

$$time(i,r) = 1 - \mathbb{1}(t^i_{r-1} > T_{max}).\left(1 - \alpha \cdot \frac{T_{max}}{t^i_{r-1}}\right)$$

where T_{max} is the desired duration of each round and t^i_{r-1} is the time taken by client i to complete the previous round $(r-1)$. Specifically, $t^i_{r-1} = t^i_{dl} + t^i_{ul} + t^i_{train}$ is the sum of the model weights download time (t^i_{dl}), upload time (t^i_{ul}), and the local training time (t^i_{train}) in round $(r-1)$. Our time utility metric penalizes clients whose total round completion time t^i_{r-1} is higher than the threshold T_{max} by a developer-defined factor α, ranging from $[0, 1]$. Lower α will penalize slow clients more. For clients whose round completion time is less than T_{max}, time utility is always 1.

Overall utility. Finally, to strike a balance between accuracy, convergence time, and energy efficiency, we multiply the three utilities to obtain a unified utility metric for device i in round r is as follows:

$$Util(i,r) = stat(i,r) \times system(i,r) \times time(i,r)$$

In FLAME, each device computes its unified utility and reports it to the server. In the next round, the server sorts all the devices by their unified utility value in a descending order, and select the top C devices subject to the constraint that they are distributed across $\frac{C}{\rho}$ users.

4.3 Model Personalization

Finally, to address the challenge of uneven accuracies across devices and users as shown in Figure 5, FLAME personalizes [41] the global model to each device during training. Let w_r denote the weights of the global model in a given round r, which are obtained by averaging the weights from each selected device. In addition to this global model, FLAME also trains a personal model on each device, which is trained using only the local data from the device. As this model is personalized to each device, it can be expected to result in higher inference accuracy as compared to the global model and avoid the challenge of uneven accuracies across devices.

Algorithm 1 FLAME

for $r = 0$ to $R - 1$ **do**
  $C_r \leftarrow$ A subset of clients are sampled ▷ Client selection (§4.2)
  for client $c \in C_r$ in parallel **do**
    for local iteration $e = 0$ to E **do**
      $w_c^r \leftarrow \text{ClientUpdate}(c, w^r)$ ▷ Global model update
      Send w_c^r back to the global server for aggregation
      $v_c^r \leftarrow v_c^r - \eta(\nabla \mathcal{L}_c(v_c^r) + \lambda(v_c^r - w^r))$ ▷ Device model update (§4.3)
    end for
  end for
  $w^{r+1} \leftarrow \sum_{i=1}^{|C_r|} \frac{n_i}{n} w_i^r$ ▷ Server aggregates global model updates
end for

Let v_i denote the weights of the personal model for the device i, which are initialized randomly. Training the personal model follows an update rule below:

$$v_i = v_i - \eta(\nabla L_i(v_i) + \lambda(v_i - w^r)),$$

where η denotes the local learning rate, $\nabla L_i(v_i)$ is the gradient of the loss function on the local dataset, and $(v_i - w^r)$ is a regularization term that ensures that the weights of the personal model do not diverge too much from the global model. Finally, λ is a hyperparameter that balances the local training objective and the regularization term; a higher λ pushes the personal model towards the global model, while a smaller λ encourages higher personalization. Please note that our model personalization approach is inspired by the work of Li et al. [41]; we do not claim novelty on the algorithm or its underlying theoretical foundations. Instead, we argue and empirically validate that the use of model personalization brings significant benefits for FL in a multi-device environment and addresses the challenge shown in Figure 5.

The algorithm for FLAME is summarized in Algorithm 1 and illustrated in Figure 6. For every round r until the final R-th round, the server selects a subset of devices, $\{C_r\}$, following the device selection strategy[2] described in Section 4.2. The server distributes the up-to-date global model weights, w^r to each device $c \in \{C_r\}$. Every selected device updates the global model on its local data for E epochs, generating new global model weights w_c^r. Along with the global model update, each device also updates its personal model following the personalization update rule above, generating v_c^r. The new global model weights w_c^r of each client c are sent back to the global server for aggregation using federated averaging algorithms. The personal model is not shared with the server.

Takeaways: FLAME employs a user-centered FL training approach in combination with a device selection scheme that balances accuracy, convergence time, and energy efficiency of FL. Further, the use of model personalization in FLAME is aimed at reducing the inconsistencies (or variance) in inference performance across devices.

5 EVALUATION

We present a rigorous evaluation of FLAME on three multi-device HAR datasets by comparing the accuracy, time-efficiency and energy-efficiency of our approach against a number of FL baselines. Our key results are:

[2]Only in the first round, devices are sampled randomly as no prior information about their state is available.

- FLAME outperforms various federated learning baslines in terms of inference performance. FLAME achieves higher F_1 score (4.8-33.8%) than its baselines in all three datasets for both personalized and global models.
- FLAME results in less *invalid* devices after 100 rounds of training for all three datasets. Energy-aware device selection algorithm of FLAME leads to more balanced client distribution of the training efforts and reduces the final number of invalid devices by 1.02-2.86×.
- FLAME speeds up convergence to a target accuracy. When we set the target accuracy to the lowest final accuracy among baselines, FLAME sped up the convergence up to 2.02×.
- FLAME achieves the highest inference performance while maintaining system-side benefits over different device sampling strategy baselines and ablation settings.

5.1 Datasets

For our experiments, we use three multi-device datasets for human activity recognition: Opportunity, RealWorld, and PAMAP2. The characteristics of these datasets are described in Table 1.

Table 1. Datasets used in the paper, along with their pre- and post-partitioning statistics.

Name	# Devices Per User	Before Partitioning		After Partitioning		Sampling Rate
		# Users	Average # of training samples per device	# Users	Average # of training samples per device	
RealWorld [68]	7	15	1481	149	130	50Hz
Opportunity [60]	5	4	3847	28	356	30Hz
PAMAP2 [59]	3	8	1057	78	87	100Hz

RealWorld [68]: This dataset consists of data from 15 participants performing 8 locomotion activities: jumping, lying, standing, sitting, running, walking, climbing down, and climbing up. While performing the activities, 7 IMU-enabled devices were placed on the user's body at the following positions: head, chest, upper arm, waist, forearm, thigh, and shin. Accelerometer and gyroscope traces were recorded from the devices simultaneously at a sampling rate of 50 Hz.

Opportunity [60]. This dataset consists of data collected from 4 participants performing activities of daily living with 17 on-body sensor devices. For our study, we used five devices deployed on back, left lower arm, right shoe, right upper arm, and left shoe, and we targeted to detect the mode of locomotion: *stand, walk, sit, and lie*. Accelerometer and gyroscope traces were recorded from the devices simultaneously at a sampling rate of 30 Hz.

PAMAP2 [59]. This dataset includes data recorded from 9 subjects performing 18 different activities. As 6 were optional activities, we only used 12 activities among them: ascending stairs, cycling, descending stairs, ironing, lying, nordic walking, rope jumping, running, sitting, standing, vacuum cleaning, and walking. Users were instrumented with IMUs placed at 3 different body positions: head, chest, ankle; and accelerometer and gyroscope data were sampled from the devices simultaneously at a sampling rate of 100 Hz. We chose this dataset because in addition to locomotion activities, PAMAP2 also contains examples of activities of daily living (ADL) such as ironing, vacuum cleaning, and rope jumping.

Based on prior works, we segment the accelerometer and gyroscope data in time windows of 3 seconds for RealWorld and 2 seconds for Opportunity and PAMAP2 datasets without any overlap overlap [10, 47].

5.2 Experiment Testbed

To perform a realistic evaluation of a FL system, we need an experiment testbed that simulates the characteristics of real-world federated learning. One of the key motivations behind FL is that local clients have an *insufficient amount of training data* to learn a good prediction model, and hence they collaborate with other clients to learn a shared prediction model. Unfortunately, some of the prior work on FL with HAR data disregards this assumption; for example, Yu et al. [77] assume between 3000 to 5000 seconds of labeled data on each client for the RealWorld dataset. This raises two concerns: firstly, it is infeasible for users to label several hours of accelerometer and gyroscope data on their devices. Secondly, and perhaps more critically, if each client has a large number of labeled data samples, they may not even need collaborative training algorithms such as FL to train prediction models.

Hence, for a robust evaluation of FL systems, we need to come with ways to partitioning existing HAR datasets to make them appropriate for federated settings. Next, we present a novel data partitioning scheme that can take any multi-device HAR dataset and distribute it over a large number of client devices. *We are in the process of open-sourcing the source code of our partitioning algorithm as well the federated HAR datasets for reproducibility.*

Class-Based Data Partitioning for New User Generation. A realistic FL testbed should have a large number of users each owning a small number of labeled data samples. To this end, our data partitioning scheme generates new users by partitioning the data of existing users in the dataset. This scheme preserves the multi-device nature of the MDE setting and performs *class-based* data partitioning.

Figure 8 demonstrates an example of partitioning a dataset with $N = 4$ users, $L = 4$ classes (walking, running, sitting, standing), and $D = 2$ devices per user (D1, D2). Assume that we wish to partition this dataset across $N' = 16$ users in total (i.e., create 12 new users). We first divide the data samples of each of the existing users into N'/N chunks. The original $N = 4$ users keep one chunk of their data and rest of the chunks are distributed to create 12 ($N' - N$) new users. Each new user is created by mixing different classes from the original users in a distinct combination; for example, the user 5 in Figure 8 contains 'Walking' class from User 1, 'Running' from User 2, 'Sitting' from User 3, and 'Standing' from User 4. Note that the scheme does not partition the device set of an original user in order to preserve the multi-device characteristics of the MDE; a new user has different classes from different users, but each class has all devices of the user. Please refer to Table 1 for details on the partitioned datasets. As an example, the RealWorld dataset has 1481 training samples (74 minutes of data) per device from 15 users before partitioning. After partitioning, it changes to 149 users with 130 training samples (6 mins of data) per device.

Energy Drain Profiles. Our FL testbed also incorporates the energy drain profiles of modern embedded processors, thus allowing us to quantify the energy drain associated with training models on real client devices. To this end, we measure the energy consumption of training an HAR model on 9 different processors as shown in Table 2. These devices are chosen because of their compatibility with Python that enable us to federate TensorFlow model training on them. For energy profiling, we choose the same training configuration of the HAR model (i.e., network architecture, learning rate, local epochs, dataset) that is used in our end-to-end FL experiments (see §5.4). We run the training for five rounds and measure the mean training time per round and the mean energy consumption on each processor separately. In total, this results in 9 realistic device profiles for each dataset. Finally, in our large-scale FL evaluation, each client in the FL setup (obtained after data partitioning) is randomly assigned one of the 9 device profiles. When a client participates in a training round, it experiences energy drain according to its assigned profile.

We note that a recent work called FedScale [36] took a similar approach of creating an experiment testbed for FL; however they did not include any simulation of realistic energy consumption on edge devices.

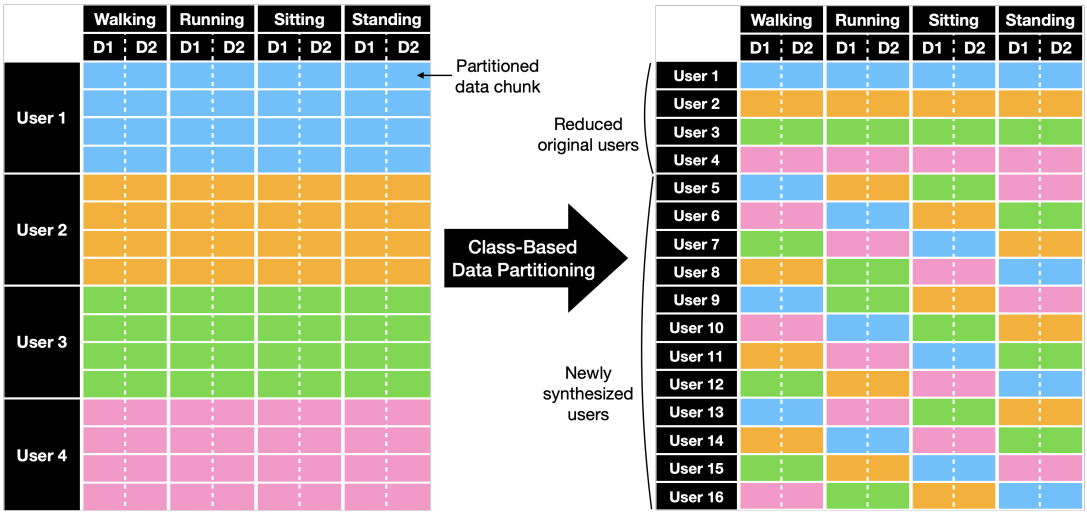


Fig. 8. Class-based data partitioning for a realistic federated learning multi-device user generation. An example of partitioning 4 users, 2 devices per user (D1 and D2), and 4 activity classes (Walking, Running, Sitting, and Standing) generates 12 new users with distinct profiles.

Table 2. Embedded processor profiles with training time and energy consumption per round for training an HAR model on the partitioned RealWorld dataset. Similar profiles are created for the other two datasets.

Name	Processor Type	Release Year	Training time per round (in seconds)	Energy consumption per round (in Joules)
Raspberry Pi 4 (Model B)	CPU	2019	38.18	69.87
Jetson Nano	CPU	2019	50.31	27.3
	GPU	2019	33.10	22.5
Jetson Xavier NX	CPU	2019	23.12	15.5
	GPU	2019	16.11	13.7
Jetson AGX Xavier	CPU	2018	16.0	8.85
	GPU	2018	11.11	7.36
Jetson TX2	CPU	2016	42.79	128.9
	GPU	2016	28.73	87.3

Realistic Network Bandwidths. Finally, to simulate varying network bandwidth across users, our testbed assigns a realistic network bandwidth to each user participating in FL. We collect the data on average download/upload speeds on mobile devices in different countries from SpeedTest [66], and each user in the FL system is randomly assigned a download/upload speed from this dataset. All devices of a user share the same download/upload speed.

5.3 Baselines

We use two federated learning baselines to compare against FLAME: FedAvg [8] and Ditto [41]. Unlike FLAME, FedAvg considers each device in our problem setup as a separate client and ignores the association between different devices of a user. In each round of training, FedAvg randomly samples C devices and averages their models on the server. Ditto [41] is an extension of FedAvg that also includes the training of personalized models. Further, we compare FLAME against Oort [37] which is an extension of FedAvg with heterogeneity-aware device selection policies. As our work operates under the paradigm of supervised FL, we do not compare it against any unsupervised or semi-supervised FL baselines [2, 77].

5.4 Implementation Details and Hyperparameters

For our experiments, we use the DeepConvLSTM model architecture proposed for HAR [64]. Our FL training setup is implemented in Tensorflow using the Flower framework [4]. We use the TF HParams API[3] for hyperparameter tuning and arrived at the following training hyperparameters: {FLAME learning rate = $1e^{-3}$, batch size = 32, optimizer = Adam, rounds = 100, local epochs = 20 for RealWorld and PAMAP2 and 10 for Opportunity}. For other variables in FLAME, we use personalization factor λ = 1.0, device sampling ratio = 0.5, α for time utility = 0.5, and α for Oort's time utility = 2.0.

5.5 Evaluation Metrics

We use a 80-20 train-test split of each dataset; the training set was used for updating the personal and global models and the testing set was used for evaluation. Please note that unlike the more robust k-fold cross validation or leave-one-user-out evaluation, we opted for a simpler train-test split evaluation. This is done due to the high costs (both monetary and environmental) associated with federated learning experiments. For instance, each experiment on the RealWorld dataset requires federated training on 1043 devices (149 users x 7 devices per user) for 20 local epochs and 100 rounds, which in total consumes around 104 GPU-hours on Nvidia V100 GPUs. Hence, we decided not to repeat this experiment for multiple folds to reduce the monetary, and more importantly the environmental costs of training.

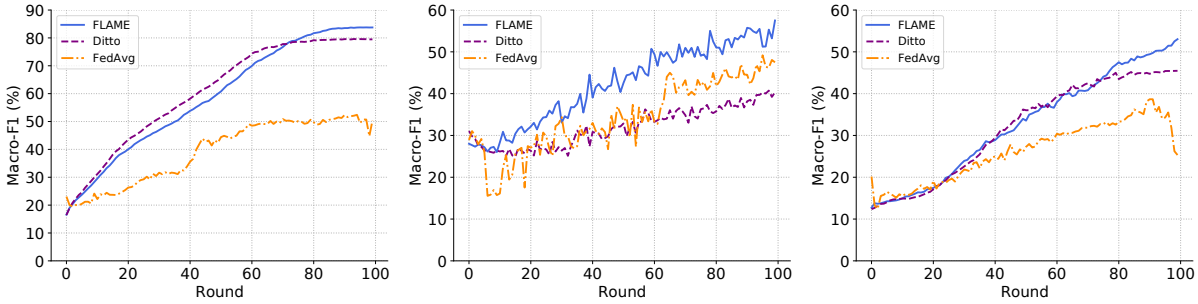
We compare FLAME against the baselines on three metrics: inference performance, energy drain, and convergence speed. For the inference performance, we report the macro F_1 score which is considered a good performance metric for imbalanced datasets [56]. For energy drain, we report the number of devices that become 'invalid' for training in each round. A device becomes 'invalid' when its energy drain grows above a predefined drain threshold. Assuming a 3000mAH battery, we used a conservative drain threshold of 10% of the battery capacity.

5.6 Results

In this section, we present our results comparing FLAME against the baselines in a number of experiment settings.

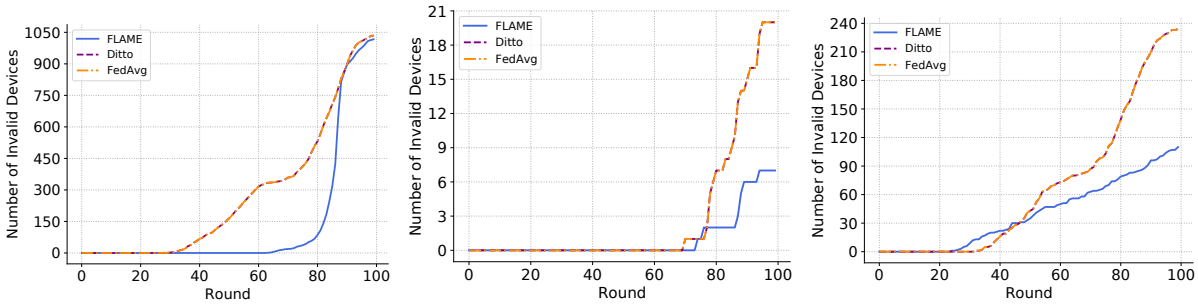
FLAME improves the test accuracy of FL models. Figure 9 shows the plot for macro F_1 score obtained per round, and Table 3 summarizes the final F_1 score for personalized device models and global models averaged over all the devices in the dataset. In all three datasets, FLAME achieved the highest F_1 score over Ditto and FedAvg after 100 rounds of federated training. Interestingly, we observe that in RealWorld and PAMAP2 datasets, FLAME and Ditto show similar inference performance at start; however, once the number of invalid devices start increasing in Ditto (see Figure 10), its performance saturates due to a smaller pool of available devices to train on. This shows how energy-aware device selection could benefit not only the energy consumption but also the prediction performance of FL algorithms.

[3] https://www.tensorflow.org/tensorboard/hyperparameter_tuning_with_hparams



(a) RealWorld dataset (b) Opportunity dataset (c) PAMAP2 dataset

Fig. 9. Round-to-F1 score plot.



(a) RealWorld dataset (b) Opportunity dataset (c) PAMAP2 dataset

Fig. 10. Number of invalid devices over rounds.

FLAME distributes the FL workload across devices in an energy-aware manner, resulting in the least number of invalid devices after training. Figure 10 show the number of invalid devices over rounds. It is important to note that due to local model training, devices will inevitably run out of the energy budgets assigned to them, more so if the budgets are conservative. Hence, an ideal FL algorithm should: a) result in lesser number of invalid devices after training, and b) distribute the workload equitably across devices such that the onset of invalid devices is delayed during training. From Figure 10, we can observe that FLAME has the least number of invalid devices for all three datasets, and it succeeds in delaying the onset of invalid devices for RealWorld and Opportunity datasets.

FLAME lowers the variance in F_1 score across devices of the same user. As shown in Figure 5, FL models often do not generalize well to different devices, and may result in uneven accuracies across devices of the same user. Figure 11 shows the variance in the macro F_1 score of personalized and global models across devices of the same user. We first compute the variance in macro F_1 across all devices of each user. The average of these variances over all users is reported in Figure 11. We observe that both device and global models trained with FLAME lower the across-device variance, compared to Ditto and FedAvg, thereby leading to more equitable performance across devices.

Table 3. End-to-end performance of FLAME compared to Ditto and FedAvg baselines. The final model F_1 score of FedAvg's global model is used as the target F_1 score for speedup calculation. The best performance in each column is marked in bold. Device model results for FedAvg is marked 'N/A' as FedAvg does not train a device model. Ditto does not reach the target F_1 score with its device model in Opportunity dataset, hence its speedup is marked '-'.

Dataset	Algorithm	Macro-F_1 Score		Target F_1 score	Speedup	
		Device	Global		Device	Global
RealWorld	FLAME	**83.8%**	**52.6%**	50.4%	1.89×	**1.00**×
	Ditto	79.5%	51.7%		**2.41**×	0.85×
	FedAvg	N/A	50.4%		N/A	1.00×
Opportunity	FLAME	**57.5%**	**48.8%**	47.5%	**1.61**×	**2.02**×
	Ditto	40.3%	48.8%		-	1.05×
	FedAvg	N/A	47.5%		N/A	1.00×
PAMAP2	FLAME	**53.0%**	**39.7%**	25.3%	**1.27**×	**1.05**×
	Ditto	45.5%	22.7%		1.20×	1.02×
	FedAvg	N/A	25.3%		N/A	1.00×

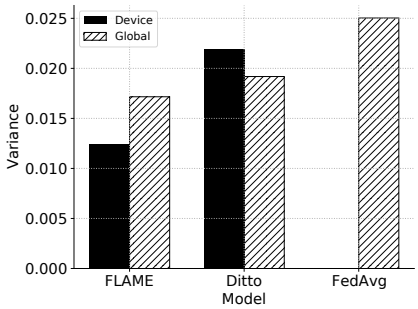


Fig. 11. Variance of device and global model inference F_1 score across each user's devices, averaged over users.

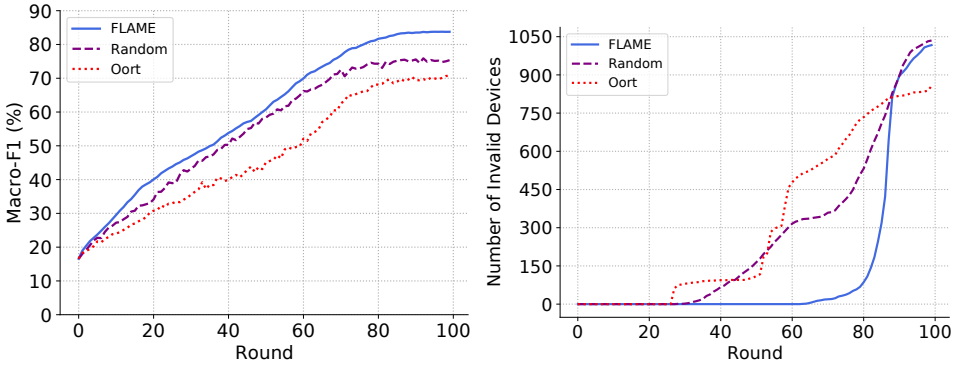
Convergence time of FLAME. Table 3 shows the convergence time speedup for FLAME. As the two baselines do not achieve the same accuracy as FLAME, we use the final accuracy of the least accurate baseline (FedAvg) to measure the convergence speedup. In other words, we measure the time taken by FLAME and Ditto to achieve the same accuracy as FedAvg. Our results show that FLAME achieves a slight speedup in global model accuracy over the baselines for RealWorld and PAMAP2, while it sees a 2.02× speedup for Opportunity. The speedups for the device models are higher as they converge faster due to personalization.

5.7 Analysis of FLAME

In this section, we present ablation studies and a detailed analysis on the constituent algorithms of FLAME.

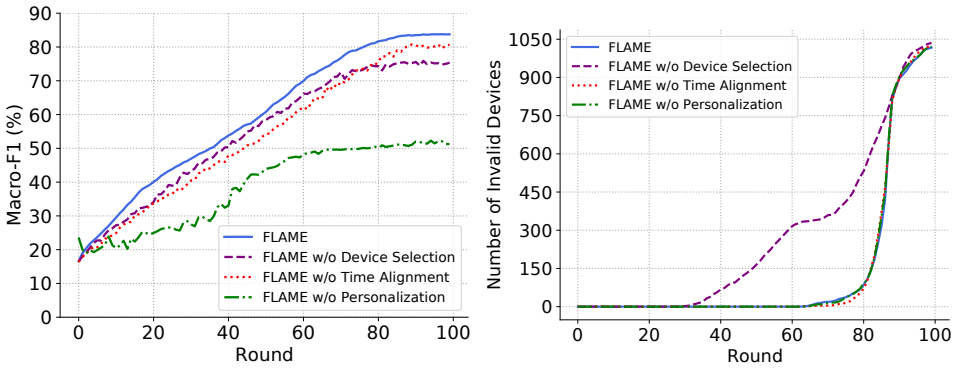
FLAME outperforms baseline device selection strategies. We compare FLAME's unified device selection strategy (Section 4.2) against the following baselines:

- **Random**: devices are randomly selected for training.
- **Oort**: device selection based on Oort [37] which combines statistical utility and a different time utility.



(a) Device Model Round-to-Accuracy (b) Invalid Devices

Fig. 12. Comparison between different device sampling strategies using the RealWorld dataset.

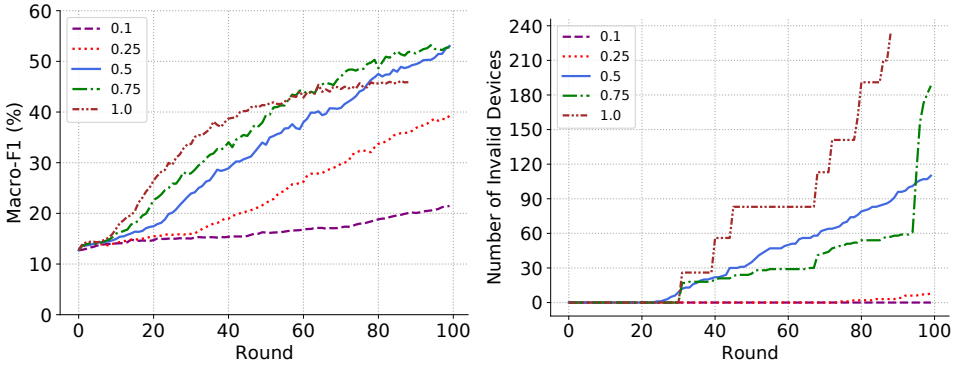


(a) Round-to-accuracy (b) Invalid Devices

Fig. 13. Ablation study using the RealWorld dataset.

Figure 12 illustrates how the personalized model F_1 score and number of invalid devices changes over rounds for all sampling strategies. FLAME achieves the highest inference F_1 score using the personalized model when compared to the other device sampling strategies. Oort has an overall less number of invalid devices; however, the onset of invalid devices starts early in round 27, which hampers its convergence. FLAME on the contrary manages to distribute the training workload across devices, and the onset of invalid devices only starts around round 70.

Ablation Study. Figure 13 shows our ablation study results, comparing FLAME with ablations of FLAME without our device selection strategy (i.e., random sampling), without enforcing time alignment across devices of the same user, and without personalization. We observe that FLAME achieves the highest F_1 score over the ablation baselines while achieving comparable system performance.



(a) Device Model Round-to-Accuracy (b) Number of Invalid Devices

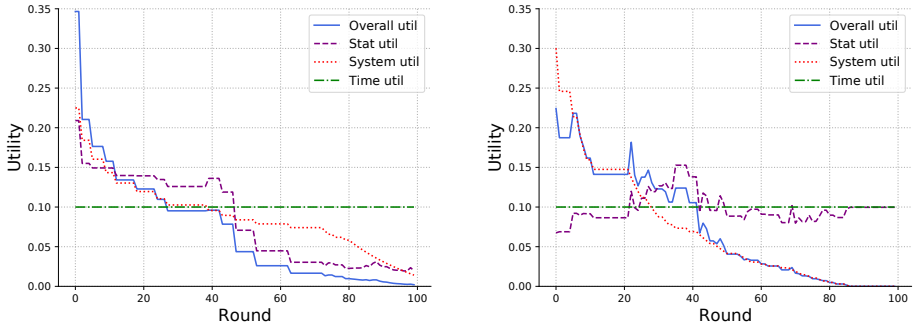
Fig. 14. Varying the number of devices selected in each round using the PAMAP2 dataset.

Varying the number of clients selected in each round. In our main experiments, we used a client sampling ratio of 0.5, that is 50% of the total devices were sampled in each round of FL. We now present an experiment on the PAMAP2 dataset measuring the impact of varying the number of clients sampled in each round, from 0.1, 0.25, 0.5, 0.75, 1.0 fraction of total clients. Figure 14 shows the per-round F_1 score for personalized models and the number of invalid devices during training. We observe that sampling more clients (fractions: 0.5, 0.75, or 1.0) leads to faster convergence to higher F_1 score than sample fractions of 0.25 or 0.1. At the same time, we see diminishing returns with oversampling; for example, sample fraction of 0.5 performs better than 0.75 and 1.0. Such diminishing returns associated with oversampling have been shown in prior works as well [12]. Another explanation for this result is that sampling higher fraction of clients in each round depletes the energy of participating devices faster, leading to more invalid devices and hurting the overall F_1 score.

Illustration of the Device Selection strategy. Figure 15 illustrates how the three utility components (Statistical, System, Time) and the overall utility change over 100 rounds for two devices: *Device 1*, where our selection strategy manages to balance statistical and systems utility over time and *Device 2*, where it falters to some extent. The plot is normalized to depict all utility components in the same plot.

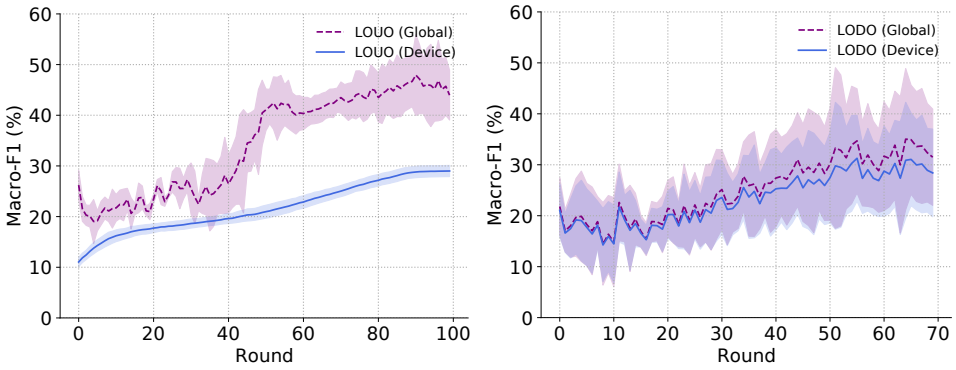
Device 1 starts with a high statistical and systems utility and is frequently sampled by our strategy until round 23. This can be verified by the constant reduction in the system utility for the device. After round 23, the statistical utility of Device 1 becomes low (i.e., it does not provide enough useful gradients to the latest global model) and its system utility also drops. As such, the device is sampled less frequently for training. This ensures that the energy budget of the device is not exhausted early, and the device manages to contribute to FL until the last round. On the contrary, Device 2 is sampled frequently throughout the training process due to its high overall utility caused by a high systems utility (until round 15) and high statistical utility (round 20 onward). As a result, this device exhausts its energy budget around round 85.

Generalizability of the learned classifiers. Finally, we evaluate the generalizability of the learned global and device models on new users and new devices of the same user through leave-one-user-out (LOUO) and leave-one-device-out (LODO) experiments, on the RealWorld dataset. For the LOUO experiments, we randomly select three users (S1, S2, and S11) to be left out. In each LOUO experiment, one of the three users is excluded from training, and the global and device models trained with all other users are tested on this held-out user. In



(a) Device 1 (b) Device 2

Fig. 15. Illustration of utility variation over rounds for two devices in the RealWorld dataset.



(a) Bootstrapping new users (b) Bootstrapping new devices

Fig. 16. Macro-F_1 scores of learned device and global models tested on unseen users and unseen devices. This depicts better generalizability of global models than device models.

the LODO experiments, one of the seven devices in the RealWorld dataset is held-out and the models are trained on the remaining devices.

Figure 16a and 16b report the mean and standard deviation of F_1 scores over 100 rounds for the three LOUO and seven LODO experiments. In both cases, we observe that the global models achieve higher test accuracies for unseen users and unseen devices, which suggests that the global FL models generalize better than the personal models. This result highlights the need for maintaining both personal and global models during FL; while personal models offer higher inference F_1 score on each device, global models could be used to bootstrap HAR inference for new users and new devices that may appear in the future. Eventually, we expect that the new users or devices will also participate in FL training and learn a personalized model using FLAME.

6 DISCUSSION AND LIMITATIONS

In this section, we elaborate on the limitations of FLAME and discuss future work on this topic.

Generalizability of FLAME. While the evaluation of FLAME was done on HAR tasks based on inertial sensing data, the core ideas of FLAME can be applied to train machine learning models in other types of multi-device systems, such as multiple microphones collecting a user's speech in a smart home or multiple cameras assessing driving behavior at a traffic junction. Further, our FL testbed is highly adaptable and can be easily re-purposed to partition and scale any small-scale sensor datasets for FL evaluations.

The Use of Energy and Latency Profiles. We created energy and inference latency profiles of various embedded devices, and assigned those profiles to the FL clients in our evaluation. This methodology of simulating the energy drain and latency behavior of clients can assist FL researchers in quantifying the resource consumption of their algorithms in a quasi-realistic setup. That said, currently we have only profiled nine embedded processors specifically for HAR models. In order to scale this approach, more fine-grained profiles should be developed that can quantify the resource consumption on client devices based on the number of mathematical operations needed during local model training.

Model Optimization and Averaging Algorithms. In this paper, we did not explore the use of model optimization, which is an effective solution to speed up and reduce the resource consumption of local training [9, 20, 29]. Moreover, we only focused on FedAvg as the averaging algorithm for local parameters; the choice of averaging algorithm is an active area of research in FL and newer approaches are being proposed to deal with non-IID-ness during averaging [44, 46]. Although an in-depth study of averaging algorithms or model optimization techniques is out of the scope of this paper, we believe they can be easily incorporated with FLAME in future works.

The Reliance on Labeled Data. In its current form, FLAME operates under the paradigm of supervised federated learning [51]. We assumed that each device will have a small yet sufficient number of labeled data instances to drive federated training as well as model personalization. This assumption comes along with the well-known drawbacks of supervised learning on sensor data, e.g., the challenges associated with data labeling. We note that unsupervised [79] and semi-supervised FL [3, 62, 77] are active research areas in ML and future works can investigate them in the context of multi-device systems.

Similarity-Based Client Clustering. An alternative approach to reducing non-IID-ness in sampled clients is to perform similarity-based clustering on their data and sample clients with higher similarity. We did not adopt this approach because in order to get a robust measure of distribution similarity, each client needs to share either raw data or features extracted from the data with the server. This violates the privacy of the personal data on each device. However, recent works have shown that one can get an approximation of client similarity by computing statistical measures such as the KL-Divergence on model parameters [16, 63]. These approaches are more private, and could be extended to multi-device settings in future work.

Data Privacy. FLAME does not share any raw data or features across devices, or between the devices and the server during FL. Hence, it provides some level of privacy protection. However, recent works in FL have shown that model parameters exchanged during training could leak the raw data [5]. This remains an open challenge with FLAME as well. Techniques such as differential privacy [73] are being employed in FL literature to address these challenges, and they could be applied to FLAME in future work.

7 RELATED WORK

Overview of Research Challenges in Federated Learning. Despite being a relatively new area in machine learning, FL has seen tremendous interest from both machine learning and systems community [43]. FL can be

categorized into two types depending on the scale and nature of federation: *cross-silo* and *cross-device* [30]. In *cross-silo* settings, large organizations collaborate to train a model, often using data center infrastructure and large local datasets. The more constrained scenario is that of *cross-device* FL wherein clients are generally thousands of mobile or wearable devices with limited computational capabilities and relatively small amount of local data. Our work focuses on *cross-device* FL. Recent works on this topic have investigated the issues of statistical and systems heterogeneity [26, 27, 33, 44, 80]; achieving scalability, privacy, and fairness with respect to participating clients [6, 11, 42, 45, 65]; and improving communication efficiency [1, 35, 38, 58]. Our work builds upon this line of research, albeit in the context of multi-device FL systems. We tackle the statistical and system heterogeneity in multi-device FL using a user-centered training approach which promotes the participation of time-aligned devices in each round of FL. Further, we build upon the literature on client selection in FL [13, 14, 16, 37, 37, 53] and propose a unified client selection metric to balance model accuracy, training speed, and energy efficiency in multi-device FL.

Personalization in Federated Learning. To address statistical heterogeneity and fairness challenges across FL clients, researchers have studied the personalization of local models to client characteristics. Mocha [65], Ditto [42], IFCA [19], and ClusterFL [63] leverage multi-task learning to model structural relationships in distributed data and use the similarities for personalization. GraFeHTy [62] applies a Graph Convolution Network with graph representations of activity inter-relatedness. APFL [15] and MAPPER [50] perform personalization through mixing the local and global models via model interpolation. FedDL [71] and FjORD [25] dynamically adapt model layers and model size for heterogeneous devices to participate in FL. Some studies further employ self-supervised or semi-supervised learning methods to utilize unlabeled sensor data for personalized federated learning [3, 62, 77]. Meta-HAR [40] adopts Model-Agnostic Meta Learning to boost the representation ability of the shared embedding network and uses personalization for adaptation on top of it. Although these works tackle heterogeneous FL environments through personalization and representation learning, all of them share an underlying assumption that the raw data of multiple devices owned by a user can be freely exchanged among each other. In other words, these FL pipelines treat each user as a client node of federated learning, who has access to raw sensor data from all the different devices owned by them. However, this inter-device raw data communication incurs not only significant communication cost but also has privacy concerns. In contrast, FLAME does not share any raw data or features even between the devices of the same user. Instead, it leverages the time-aligned nature of data collection in the MDE setup to reduce the statistical heterogeneity in multi-device FL.

Applications of Federated Learning in HAR. While many of the early use-cases of FL were related to natural language processing [21], visual recognition [24, 49], and speech recognition [22] tasks, we are now also witnessing its applications in the area of human-activity recognition [3, 18, 48, 62, 63, 77]. We extend this line of work on FL in HAR, albeit in the context of multi-device environments. The evaluation of our proposed approach is done on HAR datasets containing locomotion states and activities of daily living.

Multi-device environments and multi-device HAR. Multi-device environments offer exciting opportunities to develop accurate and generalizable sensing models by leveraging the similarities and differences across devices. This is primarily because multiple devices (e.g., a smartphone, a smartwatch, a smart glass) capture the same physical phenomenon (e.g., a user's motion activity) from different perspectives. By intelligently combining these multi-perspective datasets, we can potentially develop more robust sensory inference models. Previous research also investigates ways to improve the runtime system performance through sensor selection [31, 32, 34, 39, 78] and sensor fusion [54, 55, 72, 75, 76] in multi-device environments. However, multi-device environments suffer from the well-known challenge of labeled data scarcity, in that there is often insufficient labeled sensor data for each user to be able to train a robust inference model on it. To overcome this challenge, the research community has focused on collecting multi-device datasets from multiple users and training a sensory prediction model on

the aggregated dataset [60, 70]. This methodology is called *centralized training*, because the data from different users is aggregated in a central repository, and a prediction model is trained on the aggregated dataset using machine learning techniques. The obvious downside of centralized training is that it requires sharing of raw data traces from users, thereby compromising user privacy. It is also important to note that even though raw sensor data (e.g., accelerometer traces) may not seem privacy-sensitive at first, it could be used to infer various private aspects of a user's life. As an alternative to centralized training, we explore *federated learning* [51], a more privacy-preserving approach to collaborative training for multi-device HAR.

8 CONCLUSION

Federated learning is a crucial technique to bring intelligence to pervasive mobile and edge devices while preserving privacy of users. Many challenges yet remain in applying federated learning to real-world ubiquitous computing, e.g., high non-IID-ness, heterogeneity and scarcity of labeled training data, especially with the surge of personal data-generating devices. This paper crystallizes challenges in multi-device environments by analyzing the impact of device, user, and combined heterogeneities on a human activity recognition task. Our proposed solution, FLAME, counters statistical and system heterogeneities in MDEs, featuring accuracy- and efficiency-aware device selection strategy and model personalization. Evaluation in our realistic FL testbed shows improvement in inference performance over various baselines with higher F_1 scores and lower variance across devices, while enhancing training efficiency with reduction in invalid devices and speedup in convergence to target accuracy. Our exploration in FL for multi-device environments takes one step towards accurate, efficient, and consistent deployment of privacy-preserving machine intelligence in ubiquitous sensing applications.

REFERENCES

[1] Durmus Alp Emre Acar, Yue Zhao, Ramon Matas, Matthew Mattina, Paul Whatmough, and Venkatesh Saligrama. 2021. Federated Learning Based on Dynamic Regularization. In *Proc. Int. Conf. on Learning Representations (ICLR)*.

[2] Claudio Bettini, Gabriele Civitarese, and Riccardo Presotto. 2021. Personalized semi-supervised federated learning for human activity recognition. *arXiv preprint arXiv:2104.08094* (2021).

[3] Claudio Bettini, Gabriele Civitarese, and Riccardo Presotto. 2021. Personalized Semi-Supervised Federated Learning for Human Activity Recognition. *ACM Transactions on Intelligent Systems and Technology* 37, 4 (2021), 1–19. arXiv:2104.08094 http://arxiv.org/abs/2104.08094

[4] Daniel J Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Titouan Parcollet, Pedro PB de Gusmão, and Nicholas D Lane. 2020. Flower: A friendly federated learning research framework. *arXiv preprint arXiv:2007.14390* (2020).

[5] Franziska Boenisch, Adam Dziedzic, Roei Schuster, Ali Shahin Shamsabadi, Ilia Shumailov, and Nicolas Papernot. 2021. When the Curious Abandon Honesty: Federated Learning Is Not Private. *arXiv preprint arXiv:2112.02918* (2021).

[6] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. 2017. Practical Secure Aggregation for Privacy-Preserving Machine Learning. In *Proc. ACM SIGSAC Conf. on Computer and Communications Security (CCS)*.

[7] Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. 2015. Sliced and radon wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision* 51, 1 (2015), 22–45.

[8] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017* (feb 2017). arXiv:1602.05629 http://arxiv.org/abs/1602.05629

[9] Sebastian Caldas, Jakub Konečny, H. Brendan McMahan, and Ameet Talwalkar. 2019. Expanding the Reach of Federated Learning by Reducing Client Resource Requirements. arXiv:1812.07210 [cs.LG]

[10] Youngjae Chang, Akhil Mathur, Anton Isopoussu, Junehwa Song, and Fahim Kawsar. 2020. A systematic study of unsupervised domain adaptation for robust human-activity recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 1 (2020), 1–30.

[11] Zachary Charles, Zachary Garrett, Zhouyuan Huo, Sergei Shmulyian, and Virginia Smith. 2021. On Large-Cohort Training for Federated Learning. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*.

[12] Zachary Charles, Zachary Garrett, Zhouyuan Huo, Sergei Shmulyian, and Virginia Smith. 2021. On large-cohort training for federated learning. *Advances in Neural Information Processing Systems* 34 (2021).

[13] Yae Jee Cho, Samarth Gupta, Gauri Joshi, and Osman Yagan. 2020. Bandit-based communication-efficient client selection strategies for federated learning. *arXiv* (2020), 1–4. arXiv:2012.08009

[14] Yae Jee Cho, Jianyu Wang, and Gauri Joshi. 2020. Client selection in federated learning: Convergence analysis and power-of-choice selection strategies. *arXiv preprint arXiv:2010.01243* (2020).

[15] Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. 2020. Adaptive personalized federated learning. *arXiv preprint arXiv:2003.13461* (2020).

[16] Don Kurian Dennis, Tian Li, and Virginia Smith. 2021. Heterogeneity for the win: One-shot federated clustering. In *International Conference on Machine Learning*. PMLR, 2611–2620.

[17] Aiden Doherty, Dan Jackson, Nils Hammerla, Thomas Plötz, Patrick Olivier, Malcolm H Granat, Tom White, Vincent T Van Hees, Michael I Trenell, Christoper G Owen, et al. 2017. Large scale population assessment of physical activity using wrist worn accelerometers: The UK Biobank Study. *PloS one* 12, 2 (2017), e0169649.

[18] Jie Feng, Can Rong, Funing Sun, Diansheng Guo, and Yong Li. 2020. PMF: A privacy-preserving human mobility prediction framework via federated learning. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 1 (2020), 1–21.

[19] Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. 2020. An efficient framework for clustered federated learning. *arXiv preprint arXiv:2006.04088* (2020).

[20] Pengchao Han, Shiqiang Wang, and Kin K Leung. 2020. Adaptive gradient sparsification for efficient federated learning: An online learning approach. *arXiv preprint arXiv:2001.04756* (2020).

[21] Andrew Hard, Chloé M Kiddon, Daniel Ramage, Francoise Beaufays, Hubert Eichner, Kanishka Rao, Rajiv Mathews, and Sean Augenstein. 2018. Federated Learning for Mobile Keyboard Prediction. https://arxiv.org/abs/1811.03604

[22] Andrew Hard, Kurt Partridge, Cameron Nguyen, Niranjan Subrahmanya, Aishanee Shah, Pai Zhu, Ignacio Lopez Moreno, and Rajiv Mathews. 2020. Training Keyword Spotting Models on Non-IID Data with Federated Learning. arXiv:2005.10406 [eess.AS]

[23] Khalid Hasan, Kamanashis Biswas, Khandakar Ahmed, Nazmus S Nafi, and Md Saiful Islam. 2019. A comprehensive review of wireless body area network. *Journal of Network and Computer Applications* 143 (2019), 178–198.

[24] Chaoyang He, Alay Dilipbhai Shah, Zhenheng Tang, Di Fan1Adarshan Naiynar Sivashunmugam, Keerti Bhogaraju, Mita Shimpi, Li Shen, Xiaowen Chu, Mahdi Soltanolkotabi, and Salman Avestimehr. 2021. FedCV: A Federated Learning Framework for Diverse Computer Vision Tasks. arXiv:2111.11066 [cs.CV]

[25] Samuel Horvath, Stefanos Laskaridis, Mario Almeida, Ilias Leontiadis, Stylianos I Venieris, and Nicholas D Lane. 2021. FjORD: Fair and Accurate Federated Learning under heterogeneous targets with Ordered Dropout. *arXiv preprint arXiv:2102.13451* (2021).

[26] Kevin Hsieh, Amar Phanishayee, Onur Mutlu, and Philip B. Gibbons. 2020. The Non-IID Data Quagmire of Decentralized Machine Learning. In *Proc. Int. Conf. on Machine Learning (ICML)*.

[27] Harry Hsu, Hang Qi, and Matthew Brown. 2019. Measuring the Effects of Non-Identical Data Distribution for Federated Visual Classification. *arXiv* abs/1909.06335 (2019).

[28] Yash Jain, Chi Ian Tang, Chulhong Min, Fahim Kawsar, and Akhil Mathur. 2022. ColloSSL: Collaborative Self-Supervised Learning for Human Activity Recognition. arXiv:2202.00758 [cs.LG]

[29] Yuang Jiang, Shiqiang Wang, Victor Valls, Bong Jun Ko, Wei-Han Lee, Kin K Leung, and Leandros Tassiulas. 2019. Model pruning enables efficient federated learning on edge devices. *arXiv preprint arXiv:1909.12326* (2019).

[30] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Hubert Eichner, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konecný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Hang Qi, Daniel Ramage, Ramesh Raskar, Mariana Raykova, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. 2021. Advances and Open Problems in Federated Learning. *Foundations and Trends® in Machine Learning* 14, 1–2 (2021), 1–210.

[31] Seungwoo Kang, Jinwon Lee, Hyukjae Jang, Hyonik Lee, Youngki Lee, Souneil Park, Taiwoo Park, and Junehwa Song. 2008. SeeMon: Scalable and Energy-Efficient Context Monitoring Framework for Sensor-Rich Mobile Environments. In *Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services* (Breckenridge, CO, USA) *(MobiSys '08)*. Association for Computing Machinery, New York, NY, USA, 267–280. https://doi.org/10.1145/1378600.1378630

[32] Seungwoo Kang, Youngki Lee, Chulhong Min, Younghyun Ju, Taiwoo Park, Jinwon Lee, Yunseok Rhee, and Junehwa Song. 2010. Orchestrator: An active resource orchestration framework for mobile context monitoring in sensor-rich mobile environments. In *2010 ieee international conference on pervasive computing and communications (percom)*. IEEE, 135–144.

[33] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank J. Reddi, Sebastian U. Stich, and Ananda Theertha Suresh. 2020. SCAFFOLD: Stochastic Controlled Averaging for Federated Learning. In *Proc. Int. Conf. on Machine Learning (ICML)*.

[34] Matthew Keally, Gang Zhou, Guoliang Xing, Jianxin Wu, and Andrew Pyles. 2011. Pbn: towards practical activity recognition using smartphone-based body sensor networks. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*. 246–259.

[35] 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 *NeurIPS Workshop on Private Multi-Party Machine Learning*.

[36] Fan Lai, Yinwei Dai, Xiangfeng Zhu, Harsha V Madhyastha, and Mosharaf Chowdhury. 2021. FedScale: Benchmarking model and system performance of federated learning. In *Proceedings of the First Workshop on Systems Challenges in Reliable and Secure Federated Learning*. 1–3.

[37] Fan Lai, Xiangfeng Zhu, Harsha V. Madhyastha, and Mosharaf Chowdhury. 2020. Oort: informed participant selection for scalable federated learning. *arXiv* (2020). arXiv:2010.06081

[38] Fan Lai, Xiangfeng Zhu, Harsha V. Madhyastha, and Mosharaf Chowdhury. 2021. Oort: Efficient Federated Learning via Guided Participant Selection. In *Proc. USENIX Sym. on Operating Systems Design and Implementation (OSDI)*.

[39] Youngki Lee, Chulhong Min, Younghyun Ju, Seungwoo Kang, Yunseok Rhee, and Junehwa Song. 2014. An Active Resource Orchestration Framework for PAN-Scale, Sensor-Rich Environments. *IEEE Transactions on Mobile Computing* 13, 3 (2014), 596–610. https://doi.org/10.1109/TMC.2013.68

[40] Chenglin Li, Di Niu, Bei Jiang, Xiao Zuo, and Jianming Yang. 2021. Meta-har: Federated representation learning for human activity recognition. In *Proceedings of the Web Conference 2021*. 912–922.

[41] Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. 2021. Ditto: Fair and Robust Federated Learning Through Personalization. arXiv:2012.04221 [cs.LG]

[42] Tian Li, Shengyuan Hu, Ahmand Beirami, and Virginia Smith. 2021. Ditto: Fair and Robust Federated Learning Through Personalization. In *Proc. Int. Conf. on Machine Learning (ICML)*.

[43] Tian Li, Ankit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. 2019. Federated Learning: Challenges, Methods, and Future Directions. *arXiv* abs/1908.07873 (2019).

[44] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated Optimization in Heterogeneous Networks. In *Proc. Conf. on Machine Learning and Systems (MLSys)*.

[45] Tian Li, Maziar Sanjabi, Ahmad Beirami, and Virginia Smith. 2020. Fair Resource Allocation in Federated Learning. In *Proc. Int. Conf. on Learning Representations (ICLR)*.

[46] Xiaoxiao Li, Meirui Jiang, Xiaofei Zhang, Michael Kamp, and Qi Dou. 2021. Fedbn: Federated learning on non-iid features via local batch normalization. *arXiv preprint arXiv:2102.07623* (2021).

[47] Jonathan Liono, A Kai Qin, and Flora D Salim. 2016. Optimal time window for temporal segmentation of sensor streams in multi-activity recognition. In *Proceedings of the 13th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*. 10–19.

[48] Bingyan Liu, Yifeng Cai, Ziqi Zhang, Yuanchun Li, Leye Wang, Ding Li, Yao Guo, and Xiangqun Chen. 2021. DistFL: Distribution-aware Federated Learning for Mobile Scenarios. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5, 4 (2021), 1–26.

[49] Yang Liu, Anbu Huang, Yun Luo, He Huang, Youzhi Liu, Yuanyuan Chen, Lican Feng, Tianjian Chen, Han Yu, and Qiang Yang. 2020. FedVision: An Online Visual Object Detection Platform Powered by Federated Learning. arXiv:2001.06202 [cs.LG]

[50] Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. 2020. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619* (2020).

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

[52] Chulhong Min, Alessandro Montanari, Akhil Mathur, and Fahim Kawsar. 2019. A closer look at quality-aware runtime assessment of sensing models in multi-device environments. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*. 271–284.

[53] Takayuki Nishio and Ryo Yonetani. 2019. Client Selection for Federated Learning with Heterogeneous Resources in Mobile Edge. *IEEE International Conference on Communications* 2019-May (may 2019), 1–7. https://doi.org/10.1109/ICC.2019.8761315 arXiv:1804.08333

[54] Francisco Javier Ordóñez and Daniel Roggen. 2016. Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. *Sensors* 16, 1 (2016), 115.

[55] Liangying Peng, Ling Chen, Zhenan Ye, and Yi Zhang. 2018. Aroma: A deep multi-task learning based simple and complex human activity recognition method using wearable sensors. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 2 (2018), 1–16.

[56] Thomas PlÖtz. 2021. Applying Machine Learning for Sensor Data Analysis in Interactive Systems: Common Pitfalls of Pragmatic Use and Ways to Avoid Them. *ACM Computing Surveys (CSUR)* 54, 6 (2021), 1–25.

[57] Julien Rabin, Gabriel Peyré, Julie Delon, and Marc Bernot. 2011. Wasserstein barycenter and its application to texture mixing. In *International Conference on Scale Space and Variational Methods in Computer Vision*. Springer, 435–446.

[58] Sashank J. Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and Hugh Brendan McMahan. 2021. Adaptive Federated Optimization. In *Proc. Int. Conf. on Learning Representations (ICLR)*.

[59] Attila Reiss and Didier Stricker. 2012. Introducing a New Benchmarked Dataset for Activity Monitoring. In *Proceedings of the 2012 16th Annual International Symposium on Wearable Computers (ISWC) (ISWC '12)*. IEEE Computer Society, Washington, DC, USA, 108–109.

https://doi.org/10.1109/ISWC.2012.13
[60] D. Roggen, A. Calatroni, M. Rossi, T. Holleczek, K. Förster, G. Tröster, P. Lukowicz, D. Bannach, G. Pirkl, A. Ferscha, J. Doppler, C. Holzmann, M. Kurz, G. Holl, R. Chavarriaga, H. Sagha, H. Bayati, M. Creatura, and J. d. R. Millàn. 2010. Collecting complex activity datasets in highly rich networked sensor environments. In *2010 Seventh International Conference on Networked Sensing Systems (INSS)*. 233–240. https://doi.org/10.1109/INSS.2010.5573462
[61] Bardia Safaei, Amir Mahdi Hosseini Monazzah, Milad Barzegar Bafroei, and Alireza Ejlali. 2017. Reliability side-effects in Internet of Things application layer protocols. In *2017 2nd International Conference on System Reliability and Safety (ICSRS)*. IEEE, 207–212.
[62] Abhishek Sarkar, Tanmay Sen, and Ashis Kumar Roy. 2021. GraFeHTy: Graph Neural Network using Federated Learning for Human Activity Recognition. In *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 1124–1129.
[63] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. 2019. Clustered Federated Learning: Model-Agnostic Distributed Multi-Task Optimization under Privacy Constraints. *arXiv* abs/1910.01991 (2019).
[64] Satya P. Singh, Madan Kumar Sharma, Aime Lay-Ekuakille, Deepak Gangwar, and Sukrit Gupta. 2021. Deep ConvLSTM With Self-Attention for Human Activity Decoding Using Wearable Sensors. *IEEE Sensors Journal* 21, 6 (Mar 2021), 8575–8582. https://doi.org/10.1109/jsen.2020.3045135
[65] Virginia Smith, Chao-Kai chiang, Maziar Sanjabi, and Ameet Talwalkar. 2017. Federated Multi-Task Learning. In *Proc. Adv. in Neural Information Processing Systems (NeurIPS)*.
[66] SpeedTest. 2021. Speedtest Global Index. https://www.speedtest.net/global-index.
[67] Allan Stisen, Henrik Blunck, Sourav Bhattacharya, Thor Siiger Prentow, Mikkel Baun Kjærgaard, Anind Dey, Tobias Sonne, and Mads Møller Jensen. 2015. Smart devices are different: Assessing and mitigatingmobile sensing heterogeneities for activity recognition. In *Proceedings of the 13th ACM conference on embedded networked sensor systems*. 127–140.
[68] Timo Sztyler and Heiner Stuckenschmidt. 2016. On-body Localization of Wearable Devices: An Investigation of Position-Aware Activity Recognition. In *2016 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE Computer Society, 1–9. https://doi.org/10.1109/PERCOM.2016.7456521 http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=7456521.
[69] Timo Sztyler and Heiner Stuckenschmidt. 2017. Online personalization of cross-subjects based activity recognition models on wearable devices. In *2017 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 180–189.
[70] Timo Sztyler, Heiner Stuckenschmidt, and Wolfgang Petrich. 2017. Position-aware activity recognition with wearable devices. *Pervasive and mobile computing* 38 (2017), 281–295.
[71] Linlin Tu, Xiaomin Ouyang, Jiayu Zhou, Yuze He, and Guoliang Xing. 2021. FedDL: Federated Learning via Dynamic Layer Sharing for Human Activity Recognition. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*. 15–28.
[72] Yonatan Vaizman, Nadir Weibel, and Gert Lanckriet. 2018. Context recognition in-the-wild: Unified model for multi-modal sensors and multi-label classification. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 4 (2018), 1–22.
[73] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H Vincent Poor. 2020. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security* 15 (2020), 3454–3469.
[74] Anjana Wijekoon, Nirmalie Wiratunga, Sadiq Sani, and Kay Cooper. 2020. A knowledge-light approach to personalised and open-ended human activity recognition. *Knowledge-based systems* 192 (2020), 105651.
[75] Shuochao Yao, Shaohan Hu, Yiran Zhao, Aston Zhang, and Tarek Abdelzaher. 2017. Deepsense: A unified deep learning framework for time-series mobile sensing data processing. In *Proceedings of the 26th International Conference on World Wide Web*. 351–360.
[76] Shuochao Yao, Yiran Zhao, Shaohan Hu, and Tarek Abdelzaher. 2018. Qualitydeepsense: Quality-aware deep learning framework for internet of things applications with sensor-temporal attention. In *Proceedings of the 2nd International Workshop on Embedded and Mobile Deep Learning*. 42–47.
[77] Hongzheng Yu, Zekai Chen, Xiao Zhang, Xu Chen, Fuzhen Zhuang, Hui Xiong, and Xiuzhen Cheng. 2021. FedHAR: Semi-Supervised Online Learning for Personalized Federated Human Activity Recognition. *IEEE Transactions on Mobile Computing* (2021).
[78] Piero Zappi, Clemens Lombriser, Thomas Stiefmeier, Elisabetta Farella, Daniel Roggen, Luca Benini, and Gerhard Tröster. 2008. Activity recognition from on-body sensors: accuracy-power trade-off by dynamic sensor selection. In *European Conference on Wireless Sensor Networks*. Springer, 17–33.
[79] Fengda Zhang, Kun Kuang, Zhaoyang You, Tao Shen, Jun Xiao, Yin Zhang, Chao Wu, Yueting Zhuang, and Xiaolin Li. 2020. Federated Unsupervised Representation Learning. *arXiv* abs/2010.08982 (2020).
[80] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. 2018. Federated Learning with Non-IID Data. *arXiv* abs/1806.00582 (2018).
[81] Hangyu Zhu, Jinjin Xu, Shiqing Liu, and Yaochu Jin. 2021. Federated Learning on Non-IID Data: A Survey. arXiv:2106.06843 [cs.LG]