UNIVERSITY OF CAMBRIDGE

# Bidirectional Hierarchical Federated Learning

*PhD Proposal*

Alexandru-Andrei Iacob

Homerton

First year report submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy

# Contents

# Chapter 1

# Introduction

Federated Learning (FL) is a distributed Machine Learning (ML) paradigm allowing multiple clients to train a shared collaborative model without communicating private data. It was introduced by McMahan et al. [65] as a means of reducing communication costs and lessening the privacy concerns of storing sensitive data in a centralised location, following the principle of data minimisation outlined in the White House [90] privacy report. These properties have led to FL applications with large cohorts of small edge devices, e.g., mobile keyboard prediction [27] for Android phones, and settings with larger entities subject to privacy requirements, e.g., hospitals [79]. Kairouz et al. [41] distinguish them as cross-device and cross-silo FL.

The growth in the preponderance of Federated Learning since the publication of McMahan et al. [65] can be ascribed to two primary trends. First, an increase in the privacy requirements of consumers and legal frameworks has put pressure on technology companies. This pressure drove interest in privacy-preserving ML at major corporations such as Google [65, 27, 24, 40], Microsoft [86, 17], Meta [34, 69], and Apple [72]. Second, ML has extended to domains with strict privacy requirements such as healthcare [79, 76, 71], Human Activity Recognition (HAR) [81, 70] or collaborations between competing corporations [96, 62]. Moreover, the emergence of Large Language Models (LLMs) [6] has made accessing private language corpora advantageous, leading to the development of Federated Natural Language Processing (FedNLP) [56]. Similarly, the release of openly available LLM pre-trained weights [85] allows collaboration between entities with low computational resources using FL frameworks [4, 46, 28].

While the field has enjoyed abundant scientific and industry attention, the privacy and communication benefit it provides cause significant challenges in efficiently scaling and evolving federated systems. Crucially, training a single global model is unsuitable when unusual clients require partial or complete personalisation of the model to their local data distribution.

In its standard form, FL operates directly on clients using a centralised server to distribute model parameters and then aggregate them after client training; this process is repeated for multiple rounds. However, data in FL is subject to attributes such as client geographic location, sensor hardware, and behaviour. Due to these factors, the federated distribution violates the

Independent and Identically Distributed (IID) assumption. Such *data heterogeneity* [41, sec. 3.1] is interwoven with *systems heterogeneity* [41, sec. 7.2] since clients have different computational abilities and network speeds. Additionally, the communication costs of transmitting model parameters between servers and clients are non-trivial.

Efficiency and scalability have been at the centre of FL research since Hard et al. [27] applied FL to mobile keyboard prediction at Google. Building on top of Hard et al. [27], Bonawitz et al. [8] showed that FL could be used to train models over tens of millions of smartphones. However, despite the optimistic billion-device forecasts of Bonawitz et al. [8], several limitations to the efficiency of FL emerged. These limitations are threefold: (a) synchronous FL can only effectively use hundreds of devices every round, (b) federated training is considerably slower than centralised training, (c) user devices are unreliable, leading to dropout and stragglers. These limitations received further attention in the empirical evaluation of Charles et al. [10].

Charles et al. [10] show that the performance of FL does not scale as expected when the number of clients trained every round increases despite previous theoretical work [43] indicating the contrary. Their experimental results show that the primary limitation of increasing cohort size under Non-IID settings is the miss-alignment of client models, indicated by a near-zero cosine similarity between updates. This miss-alignment limits the impact of a round, causes diminishing returns to increasing cohort size, and results in an inability to learn efficiently from client data in parallel. Thus, its scalability is limited by the ability to learn from clients on a per-sample basis. Asynchronous Federated Learning systems [94, 69, 12, 19], such as Meta's PAPAYA [34], show promise in improving efficiency and scalability; however, they introduce the new issues of staleness and high update variance.

Evolving FL systems is also a major challenge. The datasets of clients forming a federated network are generally not static. Clients may delete data immediately after generation, periodically, or ad-hoc based on memory needs or owner requests. Furthermore, the characteristics of newly added data can change gradually or immediately. For example, seasonal transitions shift captured images slowly, while changing locations leads to discrete changes. This problem is known as dataset shift [41, sec. 3.1] and represents *intra-client* heterogeneity rather than the common *inter-client* heterogeneity. Even works which maintain persistent local models [51, 3, 16, 26] assume that this model is only used within FL rounds, obfuscating such shifts.

**Structure** Given the challenges of FL and shortcomings of previous work, detailed in Chapter 2, the proposed research for my PhD aims to *provide flexible personalisation for highly efficient and scalable FL systems* by exploring the research questions outlined in Section 3.1. To achieve this goal, Chapter 3 propose a new family of FL algorithms called Bidirectional Hierarchical Federated Learning (B-HFL) based on Algorithm 1. Building on these foundations, I outline the research directions of my PhD that B-HFL enables in Section 3.4. Finally, I summarise previously completed work in Chapter 4 and present a detailed timeline for the PhD in Chapter 5.

**Notice** The following subsection provides a *summary* of B-HFL and the contributions it enables.

## 1.1 Summary of Proposed Research

Bidirectional Hierarchical FL (B-HFL) would address the aforementioned personalisation, efficiency, and evolution challenges by constructing hierarchical federated network structures that allow bidirectional and potentially cyclical dataflow where each leaf is a client and each internal node is a server. As a result, levels in the tree closer to the leaves are more personalised to the specific client population of a subtree, while those closer to the root provide more generalisable models. Furthermore, since B-HFL treats nodes homogeneously, every intermediary node can operate independently like a standard FL server, using synchronous or asynchronous execution of its sub-nodes depending on constraints.

The proposed research builds upon the work done by Iacob et al. [35] and Iacob et al. [36] on personalised and hierarchical FL. The proposed system communicates data as shown in Algorithm 1 and Fig. 3.1. Crucially, model parameters can flow bidirectionally, and nodes can apply partial updates from their parents via aggregation. Furthermore, each node can weigh children and parent parameters differently while using methods such as the adaptive server optimisers [75], Iterative Moving Averaging (IMA) [101] of model parameters, model-interpolation [16, 26], or training-based methods [51, 45, 99, 98]. Adaptive algorithms and IMA are particularly relevant as they allow each node in the tree to distinguish itself based on its previous state without necessitating additional parameter tuning. Furthermore, since each edge server controls few clients, the diminishing effects of increasing cohort sizes are avoided. Finally, when cohorts are meaningfully clustered, this structure may allow an increase in the sample efficiency of the system as each cluster decides how to optimise the generalisation-personalisation trade-off [2, 58]. The potential contributions to the field of Federated Learning include:

1. A family of efficient algorithms with minute control of personalisation and generalisation, capable of achieving communication efficiency in hierarchical networks.

2. The investigation of three techniques enabled by B-HFL: (a) allowing leaf nodes to maintain persistent local models training asynchronously to tackle dataset shift, (b) making any node in the tree capable of training with a proxy dataset to inject general information, (c) constructing vertical connections in the tree, similar to residual connections [29], to allow customisable dataflow without changing the underlying communication infrastructure.

3. Extensive empirical evaluations considering scenarios with or without meaningful client clusters in language and image/speech recognition tasks leading to intended publication at ICLR or MLSys. This publication will be followed up by a work intended for MobiCom investigating asynchronous training on resource-constrained devices with dataset shift using the Raspberry Pi FL cluster at Cambridge ML Systems.

# Chapter 2

# Background and Related Work

The standard FL objective can be modelled as seen in Eq. (2.1)

$$\min_{\theta} F(\theta) = \sum_{c \in C} p_c F_c(\theta) \ , \tag{2.1}$$

where $F$ is the federated objective, $C$ is the client set, $\theta$ is the model, and $F_c$ is the loss of client $c$ weighted by their fraction of the total number of examples $p_c$. This formulation assumes that a single global model is being trained without regard for the distribution of its performance across client datasets. Federated Averaging (FedAvg) [65] trains the global model locally, for each round $t$ it sums the update $\theta_t^c - \theta_t$ from client $c$ weighted by $p_c$ with the previous model $\theta_t$ using learning rate $\eta$, as seen in Eq. (2.2)

$$\theta_{t+1} = \theta_t + \eta \left( \sum_{c \in C} p_c \left( \theta_t^c - \theta_t \right) \right) \ . \tag{2.2}$$

The inability to colocate client data and the need to construct rough mixtures of model parameters as a compromise represent the leading causes of FL-specific challenges.

## 2.1 Heterogeneity

Non-IID data has been shown to impact both practical accuracies [100, 31] and theoretical convergence bounds [52]. It is thus worth detailing some forms of heterogeneity that Kairouz et al. [41] identify. The most commonly addressed form is quantity skew caused by clients having different amounts of data available. Standard FL algorithms effectively address Quantity skew via a simple reweighing (Eq. (2.2)). The other frequently-considered type of heterogeneity is label-distribution skew which is quantity skew per class. While these forms of heterogeneity have been most investigated, situations where features and labels are not related in the same manner across clients are more pathological and may require some form of clustering or personalisation

to tackle. In the worst-case scenario, each client may represent an entirely different task, as in Multi-Task Learning, with potentially little overlap in their solution space.

**System (hardware) heterogeneity** Devices within the federated network may differ regarding computational ability, storage, network speed, and reliability. They may also differ from themselves at a different point in time as their battery power, network connection, or operational mode vary. Importantly, variations in data-generating hardware, such as sensors, are linked to data heterogeneity. However, system heterogeneity and device unreliability harm the FL process independently of data. For example, slower hardware may result in straggling clients which elongate rounds in synchronous FL [8, 48] or operate on stale parameters in asynchronous FL [93, 34]. In addition, network or device unreliability creates client dropout, which requires oversampling clients [8] and harms the effectiveness of maintaining local state across rounds.

**Dataset Shift and Continual Learning** Allowing ML models to participate in lifelong learning effectively is the goal of continual learning [15]; however, applying continual learning to the FL context is problematic for two primary reasons. First, the optimisation objective (Eq. (2.1)) intends to find a compromise model across all clients and cannot precisely fit all their data. Consequently, if the dataset of one client shifts independently of the whole network, the federated model will find it hard to adapt. Second, continual learning techniques such as Elastic-weight Consolidation [45], PackNet [63], and Learning without Forgetting [54] are designed for task-incremental settings where class labels are known, small amounts of previous data may still be available for specialised use cases [45], or there may even be different output heads for each task. The privacy requirements of FL make such solutions difficult at the level of the federated network without the addition of persistent local storage.

## 2.2 Privacy

While privacy in FL is not currently the intended primary research direction for the near-term of my PhD, it is one of the primary concerns of the field. As such, any approaches which attempt to tackle the main challenges of FL must do so while accounting for their privacy implications.

Since FL keeps training data locally stored on the client, it offers more privacy than standard ML approaches. However, previous works [22, 5, 73, 102] have shown that models trained in a federated fashion may allow for a partial or complete reconstruction of their training data. From the perspective of the proposed research, privacy serves as a test for the feasibility of a particular method to be applied in an FL context. For example, methods which require detailed knowledge of the data of each client [100, 67, 91, 37] may be rejected as impractical. Similarly, the ability of a system to support FL privacy techniques like Secure Aggregation (SecAgg) [7, 39, 80] or Differential Privacy (DP) [20, 89, 66, 62, 40] is relevant.

Secure aggregation is a form of Secure Multi-party Computation and was introduced to FL by Bonawitz et al. [7]. It operates by having all clients generate and share secrets. The clients then mask their models using random noise so that the server can construct the actual average of client updates without knowing the parameters of any individual client. Such techniques require multiple clients to participate in aggregation concurrently and have $\mathcal{O}(C^2)$ communication cost, where $C$ is the number of clients whose models are being aggregated. This excludes, for example, fully asynchronous federated learning as proposed by Xie et al. [93].

Differential Privacy in FL [66] is formally defined as shown in Eq. (2.3)

$$\Pr[M(d) \in S] \leq e^{\epsilon}\Pr[M(d') \in S] + \delta \,, \tag{2.3}$$

where $M$ is a probabilistic model, $S$ is the output set of that model, $d$ is the dataset used to train the model and $d'$ is an adjacent dataset. Two datasets are adjacent in FL if they can be formed by adding or subtracting the local dataset of one client. Finally, $(\epsilon, \delta)$ bound the similarity of outputs between two models trained with or without a specific client. Since DP has an inherent privacy-accuracy trade-off, the most relevant factor for its usability is whether an FL system can be scaled to operate over sufficiently large populations of clients. Larger populations allow productive training while offering a low $\epsilon$ by limiting the contribution of individual clients.

## 2.3 Federated Learning Efficiency

It is now worth expanding on the trends that Charles et al. [10] discovered. Those that limit the efficiency of FL in Non-IID settings where clients perform multiple SGD steps are of particular interest. Three significant effects can be observed. First, highly heterogeneous clients may cause sudden reductions in accuracy when their models are aggregated. Second, larger cohorts bring diminishing improvements in final accuracy and speed of convergence. Third, larger cohorts decrease data efficiency as more examples are needed for every accuracy gain. More recently, Zhou et al. [101] have shown that while the federated model successfully keeps the client models in a common basin and achieves the lowest loss, it can drift from the optimum across rounds due to cohort heterogeneity. Furthermore, they also show that the efficacy of large cohorts in reducing the variance of aggregate updates is limited by the degree of data heterogeneity present.

These behaviours are approximately analogous to the well-known efficiency and generalisation limitations of large-batch training in centralised ML [42]. Charles et al. [10] find that data efficiency issues are caused by decreasing pseudo-gradient norms with increased cohort sizes and by the near-orthogonality of client updates following multiple steps of local training. The authors also find that adaptive optimisers fare better as cohort sizes grow due to scale invariance, making them particularly attractive aggregation algorithms.

### 2.3.1 Adaptive Federated Optimisation

Of particular relevance to the proposed research are Federated Averaging with Server Momentum (FedAvgM) [32] and the more general Federated Adaptive Optimisation (FedOPT) [75]. They extend the concepts of momentum and adaptive optimisation [18, 44, 77] to Federated Learning on the *server-side* by treating client updates as pseudo-gradients and maintaining information across rounds on server-side accumulators. This structure allows such strategies to minimise the impact of individual rounds by averaging their pseudo-gradients and derived quantities with those of previous rounds. Since the outcome of individual rounds is highly variable based on the combination of clients selected, such techniques offer a more consistent optimisation trajectory.

Specifically, following the account provided by Reddi et al. [75] as shown in Eq. (2.4)

$$\Delta_t = \frac{1}{|C|} \sum_{c \in C} (\theta_t^c - \theta_t) \tag{2.4a}$$

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1)\Delta_t \tag{2.4b}$$

$$v_t = \beta_2 v_t + (1 - \beta_2)\Delta_t^2 \tag{2.4c}$$

$$\theta_{t+1} = \theta_t + \eta \frac{m_t}{\sqrt{v_t} + \tau} \tag{2.4d}$$

for a given round $t$ and federated model $\theta_t$ each client $c$ in the selected set $C$ trains the model locally to construct a personalised version $\theta_t^c$. The pseudo-gradient $\Delta_t$ is computed by averaging the differences between these personalised and federated models as shown in Eq. (2.4a). All operations on tensors are element-wise, including the division between tensors.

The first-moment accumulator $m_t$ can then be constructed as the weighted average of the previous accumulator $m_t$ and $\Delta_t$ using weight $\beta_1$ as shown in Eq. (2.4b). Thus, the pseudo-gradient of the current round is smoothed by those of the previous rounds decayed using $\beta_1$. Similarly, for the version of FedOpt based on Adam [44] the second-moment accumulator $v_t$ keeps track of the second power of the pseudo-gradient denoted by $\Delta_t^2$ as shown in Eq. (2.4c). These two accumulators are then used to compute the updated model for the next round $\theta_{t+1}$ using the server learning rate $\eta$ as shown in Eq. (2.4d). The term $\sqrt{v_t}$ normalises model parameters, making the algorithm scale-invariant to the pseudo-gradient. Finally, $\tau$ controls adaptivity.

FedOPT presents several promising properties in the context of hierarchical FL. First, Reddi et al. [75] show it is highly resilient to the exact choice of hyperparameters, including learning rate, compared to standard FedAvg and FedAvgM. Second, their scale-invariance partially addresses the issues observed by Charles et al. [10] regarding the small pseudo-gradients caused by the near-orthogonality of client updates. Third, they provide a means of automatically differentiating multiple servers based on accumulator state without hyperparameter tuning.

### 2.3.2 Asynchronous Federated Learning

Together with the previously mentioned adaptive federated optimisation, asynchronous FL [93, 69, 34, 94, 12] represents another promising means of improving the overall efficiency of FL generally and B-HFL specifically by improving concurrency. In the context of FL, concurrency refers to *the number of clients training simultaneously*.

The federated cohort size trained during a round controls the system's concurrency for standard synchronous FL. Besides the data-efficiency issues discussed in Section 2.3, this round-based design introduces two factors which limit effective concurrency. First, the concurrency of a round decreases as clients finish training. Second, stragglers with slow hardware or large datasets elongate a round, usually addressed through oversampling [8] or a time cut-off.

Asynchronous FL was proposed by Xie et al. [93] as an alternative means of tackling stragglers in FL besides the standard oversampling method introduced by Bonawitz et al. [8]. In its fully asynchronous form, it functions by allowing each client to update the global federated model when they finish training. Thus, it removes client update averaging and allows the system to maintain high concurrency. However, clients may return stale updates at round $t$ generated by training the model from round $\tau$. As such, Xie et al. [93] and future works [69, 34] utilise a staleness function $s(t-\tau)$ in the aggregation to compensate, as seen in Eq. (2.5)

$$\theta_{t+1} = \theta_t + \eta \left( s(t-\tau)\, \theta_\tau^c \right) \ . \tag{2.5}$$

The benefits of the fully asynchronous approach are countermanded by its sensitivity to data heterogeneity and inability to properly utilise cohort-based techniques such as Secure Aggregation [7]. Up to a limit [10, 101, 69, 34], using cohort-based aggregation in synchronous FL imposes a variance-reduction effect which limits the impact of highly heterogeneous clients. Xie et al. [93] must instead adopt $l_2$ regularisation, similarly to FedProx[48], to constrain divergence.

To regain the variance-reduction benefits and cohort-based techniques of synchronous FL, Nguyen et al. [69] introduce FedBuff. FedBuff brings a conceptually minor but practically crucial change to async FL by introducing a buffer of size $K$, which holds updates until it is filled. After it becomes full, the server averages pseudo-gradients weighted by their staleness and updates the model. Unlike the fully synchronous approaches, concurrency remains decoupled from $K$, which only controls how often a new model version is created. Huba et al. [34] build upon FedBuff in deployment at Meta to create PAPAYA and show that it can significantly improve convergence time over synchronous FL by updating the model more frequently with the same concurrency without waiting for stragglers. Huba et al. [34] also show PAPAYA results in a fairer accuracy distribution across clients than oversampling since it incorporates updates from stragglers.

Asynchronous FL can be combined with adaptive optimisation—as done by Huba et al. [34], communication-efficient methods like dropout regularisation [19], and is entirely compatible with the hierarchical FL, as will be shown in Section 2.4.2 and Chapter 3.

## 2.4 Related Work

To tackle the inherent trade-off between optimising for the average global performance versus the performance on the data of a specific client, which can be seen in Eq. (2.1), two overall directions emerged in the literature. The first, exemplified by Fair Federated Learning [49], attempts to modify the importance of a client in the federated objective function to change the final model's effectiveness for that client. The second relaxes the single global model requirement by personalising the federated model [97, 82, 100, 13], maintaining persistent fully-local models alongside it [3, 16, 26, 51], clustering clients based on similarity [64, 23, 58], or building hierarchies [59, 1, 67]. Since the proposed B-HFL family of algorithms falls in the second camp, this section shall detail the most closely related work and present its limitations. Finally, the desired properties of B-HFL and its relation to previous work are summarised in Table 2.1.

### 2.4.1 Personalised Federated Learning

Personalised Federated Learning (PFL) refers to a class of FL algorithms which intends to tackle the Non-IID distribution of client data by creating models which better match the distribution of specific clients or groups of clients. This strategy differs from standard FL, which attempts to create the best compromise model. Thus, PFL approaches lie between creating one global model and personalising on a per-client basis, with clustering approaches offering a compromise.

#### 2.4.1.1 Per-client Personalisation

Fully personalised FL refers to creating one model per client in addition to the global one. The most common means of achieving this is a local adaptation (fine-tuning) of the federated model after training [97, 64, 13, 13]. Local adaptation is potentially combined with techniques such as Knowledge Distillation [99] or Elastic-weight Consolidation [45] for the explicit purpose of combating catastrophic forgetting [21]. However, this two-stage optimisation is challenging to implement in an FL lifecycle where the federated model may need additional training after local adaptation. Furthermore, it provides no middle ground between global and local models, which hurts the ability to integrate new clients, which they may be incapable of fine-tuning.

An alternative approach is represented by Ditto [51] for settings where clients are visited frequently and can maintain state across rounds. Ditto allows clients to maintain a persistent local model and train it alongside the federated one during FL rounds. The two models are connected by incorporating the $l_2$ distance between their weights within the loss function of the local one. Model interpolation [64, 16, 26] and techniques which maintain local personalisation layers [3, 53] also rely on clients being capable of maintaining a persistent state. The model interpolation approaches of Mansour et al. [64] and Deng et al. [16] rely on optimising local models with a mixture parameter adaptively tuned through SGD. Alternatively, Loopless Local Gradient Descent [26] allows clients to probabilistically take steps towards either local training

or partially averaging the federated model into their local one. Finally, split approaches such as those of Arivazhagan et al. [3] and Li et al. [53] train models locally but only average and update the layers up to a cut layer $q$, with the intuition that earlier layers represent more general feature extractors and later layers require more significant personalisation.

However, despite the proven benefits to local performance [3, 16] as well as fairness and robustness [51], maintaining a persistent state still faces the challenges of traditional personalised models in terms of incorporating new clients with the additional cold-start problem of initialising their local state. Moreover, for cross-device settings with low levels of participation, which arise both due to client availability and the small cohort sizes used in practice [8, 10], local client state may become stale across rounds. Finally, neither fine-tuning nor persistent-state techniques address dataset shifts within the client, as they only operate during training or adaptation rounds.

#### 2.4.1.2 Clustering

Clustering clients is a technique that attempts to group participants based on a similarity metric, with two predominant variants being used in FL. First, since directly clustering clients based on their data is unfeasible, standard clustering approaches such as K-means [60] or Hierarchical Agglomerative Clustering [38] need to operate over embeddings representing the local distribution of a client. The natural choice for such an embedding in FL is using locally-trained models (or pseudo-gradients) directly, as done in the one-shot K-means algorithm of Ghosh et al. [23], in Sattler et al. [78], and in the hierarchical clustering algorithm of Briggs et al. [9]. In such solutions, the distance function between models will be the $l_n$ norm of model differences [23, 9] or similarity metric such as the cosine similarity [78, 58, 9] computed over flattened parameters. While locally trained models are widely available and more admissible from a privacy perspective to a raw representation like prototypes [83], using the entire model is computationally expensive and potentially uninformative. As Wang et al. [87] indicates, two models may represent the same concept in different sets of weights. This issue can be addressed using an explicit encoder [33].

Second, because the methods above are computationally expensive and challenging to apply dynamically for low-participation cross-device FL, Ghosh et al. [23] and Mansour et al. [64] concomitantly proposed a loss-based form of clustering for FL. Their algorithms are iterative and operate by maintaining $K$ hypothesis models, which are communicated to clients. Clients then get assigned to the cluster model where they have the lowest training loss, and then the models of clients who are self-selected to a specific cluster are averaged to produce the new cluster models. The procedure then repeats for several rounds or until convergence; at this point, each cluster is separated and runs FL independently. However, despite the dynamic nature of such clustering being well-fit to FL, the increase in communication and computation brought by having each client interact with $K$ models is significant and hard to reconcile with other FL approaches. Furthermore, it is unclear how $K$ should be chosen without access to client data.

A practical FL system for clustering which addresses issues in both approaches, Auxo, was proposed by Liu et al. [58]. Auxo creates an initial set of clusters using K-means. Then, it dynamically adjusts the clusters by splitting them hierarchically if it heuristically detects that the split would reduce heterogeneity in the population. Clients join clusters based on an exploration-exploitation trade-off where the reward is computed based on the distance of the gradient produced by training a client on a specific cluster and the cluster average. This reward can be propagated to unexplored clusters due to the hierarchical structure, with clusters having a distance metric based on the number of steps to their most recent ancestor.

All the available clustering algorithms, including Auxo, fail to obtain the desired trade-off between generalisation and personation because they do not continuously share information between clusters. In the case of Auxo, because of the hierarchical cluster splitting, ancestors provide the initialisation weights for new clusters but become utterly disconnected afterwards. As such, creating more clusters results in them having access to increasingly fewer clients, which always hurts generalisation and may harm the performance of the cluster clients directly if they do not possess enough aggregate samples to train a high-quality model. Thus, the decision is only reasonable when the reductions in data heterogeneity are sufficient to compensate for the smaller population. One exception is the weight-sharing iterative clustering algorithm proposed by Ghosh et al. [23], which serves as a middle point between a two-layer hierarchical solution and multi-task learning with personalisation layers; however, it still suffers from the drawback of having to train $K$ versions of the model on each client. Finally, clustering algorithms are not meant to provide a single global model or intermediary models besides cluster models, even for applications where it would be beneficial to have both a good default and customised experiences.

Clusters may also exist naturally based on characteristics like location or language, which become relevant if the clients and servers controlling them are geographically correlated.

### 2.4.2 Hierarchical Federated Learning

The most relevant subfield of FL for the proposed research is Hierarchical Federated Learning (HFL) introduced by Liu et al. [59]. Their proposed HierFAVG algorithm was developed primarily to handle the communication challenges of traditional cloud-based FL. In order to obtain scales of millions of participating clients [27, 8], FL systems relied on cloud infrastructure to connect devices over a wide geographic area and thus incurred additional latency. This trade-off was considered worthwhile since the larger populations were necessary for convergence, and edge servers, while capable of fast client communication, could not draw on a sufficient data pool. Liu et al. [59] argue that a two-level structure resolves the tensions between edge servers close to the clients and cloud servers. Abad et al. [1] propose an identical algorithm for heterogeneous cellular networks where edge servers are small cell base stations, and a central macro base station replaces the cloud server. Similarly to Liu et al. [59], Abad et al. [1] focus on reducing communication costs and go further in this direction by utilising update sparsification

techniques [57, 84]. To further improve communication efficiency, Luo et al. [61] propose a resource allocation framework which assigns clients to edge servers to optimise costs.

Previous works in HFL show a series of limitations. The HierFAVG algorithm directly extends FedAvg [65] by allowing the cloud server to treat edge servers as clients. However, because Liu et al. [59], Abad et al. [1], and Luo et al. [61] only consider communication efficiency, they do not allow the edge servers to maintain greater personalisation and instead replace their model entirely during cloud-aggregation. Furthermore, their system does not consider asynchronicity, proxy training, or multi-level hierarchies. The work of Wang et al. [88], RFL-HA, combines hierarchical aggregation and clustering in a mixed scenario of peer-to-peer and client-server FL. In this setting, performant clients take on the role of edge servers and perform aggregation before transmitting their models to the cloud for asynchronous aggregation. However, their clustering procedure is meant to optimise communication efficiency first and foremost. It thus does not exploit the personalisation advantages of combining clustering and hierarchical FL.

Mhaisen et al. [67] do consider scenarios where the data distribution of edge servers is taken into account and propose optimal user-edge assignment. Specifically, they allow edge servers to contain clients with a Non-IID distribution and use the heterogeneity-resilient FedSGD [65] algorithm to counteract its effects. To obtain communication efficiency without sacrificing convergence at the cloud server, they attempt to maintain an IID distribution across edge servers and apply FedAvg at the cloud server level. While promising, their work requires complete knowledge of the distribution of each client in order to realise edge-server assignment. Furthermore, it assumes that edge servers have sufficiently low communication latency to efficiently train with FedSGD despite the original work of McMahan et al. [65] showing FedSGD to be up to two orders of magnitude slower than FedAvg in terms of convergence speed.

The current approaches to hierarchical Federated Learning cannot adequately tackle data heterogeneity because their sole objective is constructing a single consensus model to which the entire federated network is meant to converge. This is opposite to the concern of previously discussed clustering systems incapable of effectively sharing information across clusters. To address this, it is necessary to simultaneously tackle the construction of both generalisable and more personalised models while flexibly tuning the generalisation-personalisation trade-off.

Table 2.1: Gap analysis showing B-HFL's intended properties and overlap with related work.

| Related Work | Hierarchical | Per-client Personalisation | Allows Persistent State | Group Models | Meaningful Groups | Allows Async | Scalable | Private |
|---|---|---|---|---|---|---|---|---|
| Local Adaptation [97, 64, 13] | | ✓ | | | | | ✓ | ✓ |
| Persistent Models/Layers [51, 16, 3] | | ✓ | ✓ | | | | | ✓ |
| Standard Clustering [23, 9, 78, 64] | | | | ✓ | ✓ | | | ✓ |
| Auxo [58] | | | | ✓ | ✓ | | ✓ | ✓ |
| HierFAVG [59, 1, 61] | ✓ | | | ✓ | | | ✓ | ✓ |
| Optimal User-edge Assignment [67] | ✓ | | | ✓ | ✓ | | | |
| RFL-HA [88] | ✓ | | | ✓ | | ✓ | ✓ | ✓ |
| Asynchronous FL [93] | | | | | | ✓ | | ✓ |
| FedBuff [69, 34] | | | | | | ✓ | ✓ | ✓ |
| **Bidirectional Hierarchical FL** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

# Chapter 3

# Proposed Research

Given the shortcomings of traditional hierarchical FL systems, this chapter proposes Bidirectional Hierarchical Federated Learning (B-HFL), an alternative family of methods that optimises data and communication efficiency while allowing flexible degrees of personalisation. Section 3.4 then presents the future research directions of the PhD enabled by B-HFL.

## 3.1 Research Questions

The proposed research aims to answer the following research questions during the PhD:

1. Can multiple servers with small cohorts *outperform large-cohort single-server FL by avoiding the data efficiency problems identified by Charles et al. [10]?*
2. If the need for convergence to a global model is removed, *can hierarchical FL effectively address the trade-off between generalisation and personalisation with Non-IID data?*
3. If all nodes are treated homogeneously, and servers are allowed to train on proxy data, *can the regularisation strength be more effectively controlled with the hierarchical structure?*
4. Can the network structure be used to *enhance or design aggregation strategies without major harm to communication efficiency?*
5. Do persistent client models with continual asynchronous learning allow nodes to *tackle dataset shift and obtain a greater degree of personalisation?*

All of these questions are intertwined with the hierarchical structure itself, and, unlike previous work in hierarchical FL, they are all predicated on the abandonment of a single global model as the explicit goal of FL. Consequently, they can all be reduced to the following question: *Can a hierarchical FL structure allow us to better utilise node data on both clients and servers while maintaining the communication efficiency which made FL practical in the first place?*

As shown in Table 2.1 and discussed in Section 2.4, previous approaches in the field are not flexible enough to simultaneously allow trade-offs in terms of personalisation, sample efficiency, communication efficiency and asynchronous training or execution. As such, research in the following proposed family of algorithms would significantly contribute to the field.

## 3.2 System Design

Bidirectional Hierarchical FL organises communication between servers and controls the dissemination of training parameters through the following design choices:

1. While previous methods such as HierFAVG [59, 1] entirely replace the edge-server and client models after global aggregation takes place, B-HFL performs partial aggregation between a child node and their parent. This allows children to maintain their local weights while incorporating global information. It proceeds in two phases:

   (a) **Leaf-to-root aggregation** After clients finish training, their information is propagated up the tree. Each internal node has a parameter $T_n$, which determines after how many rounds it sends its updates to the parent. This value is equivalent to local client epochs during SGD and may be the same for a tree level or independent per node.

   (b) **Root-to-leaf aggregation** After a node has received and aggregated the training result from some or all of its children, it propagates its parameters down their subtree. The propagation cost is proportional to the depth; however, communication between internal nodes can be assumed to be faster than between clients and edge servers.

2. All nodes may be allowed to execute synchronously or asynchronously concerning other nodes on the same level if necessary during leaf-to-root aggregation. For leaves (clients), this is equivalent to traditional asynchronous FL [94]. For an internal node, the same federated asynchronous strategies [69, 34] can be applied when receiving models from children, with client training replaced by executing the subtree rooted at the child node.

3. Internal nodes within the hierarchical structure can train on proxy datasets to regularise training as done by Guha et al. [25], Zhao et al. [100]. Proxy training is especially relevant for language modelling as large public corpora are available. In order to avoid operating on stale parameters, the natural point for such training is after leaf-to-root aggregation and before root-to-leaf aggregation. However, the latency incurred from such training may be too large. In that case, parameters may train asynchronously while the subtree executes.

Thus, the objective function of FL from Eq. (2.1) is modified for B-HFL as described in Eq. (3.1)

$$\min_{\theta} F_q(\theta) = \alpha_q f_q(\theta) + \beta_q F_{D_q}(\theta) + \gamma_q F_{A_q}(\theta) \tag{3.1a}$$

$$F_{D_q}(\theta) = \sum_{d \in D_q} p_d F_d(\theta) \tag{3.1b}$$

$$F_{A_q}(\theta) = \sum_{a \in A_q} p_a F_a(\theta) \tag{3.1c}$$

$$f_q(\theta) = \frac{1}{|\Omega_|} \sum_{j \in \Omega_q} f_q^j(w) \tag{3.1d}$$

where each node $q$ in the tree attempts to find the model $\theta$ which minimizes its local objective $f_q$, that of its descendants $F_{D_q}$, and ancestors $F_{A_q}$ using weights $\alpha_q, \beta_q, \gamma_q$. The objective of the descendants and ancestors are recursively described, while the local objective $f_q$ is defined by

the performance of the model $\theta$ on the local node dataset $\Omega_q$. In the case of a leaf node, only its local objective and that of the ancestors matter, while for the root, only its local objective and that of the descendants matter. If an internal node lacks a proxy dataset, only $F_{D_q}$ and $F_{A_q}$ are optimised. All leaf nodes are expected to have local datasets.

Expressly, parameters aggregated from the leaf nodes (clients) up through the tree are fine-tuned to relevant local data. In contrast, parameters transmitted from parents to children are averaged over more numerous populations. When servers cover meaningfully clustered clients, these populations may be less related (e.g., covering multiple languages). Furthermore, if internal nodes are allowed to train on proxy datasets, they inject additional training into the federated models and provide regularisation for the entire tree. In traditional FL approaches, training on the server directly controlling the clients can impose overly strong regularisation; however, in B-HFL, higher nodes in the tree already represent a global picture and have limited impact at the leaves as their influence gets diluted through multiple intermediary nodes. Finally, allowing each client to maintain a persistent model across rounds and aggregate with their parents rather than entirely replacing their model makes them identical to any other node except for not having children. Keeping persistent models and repeatedly re-aggregating them is roughly analogous to the Iterative Moving Averaging applied by Zhou et al. [101].

Since not all nodes in the tree are required to be capable of training, it is worth distinguishing models which have been optimised via additional learning rather than mere aggregation. Specifically, training data being available may enable more efficient learning-based aggregation methods such as mutual learning [99], $l_2$-based regularisation [51] or model interpolation with adaptive weights [16, 64]. Additionally, updates constructed via training directly may offer a better optimisation signal, similarly to methods trying to build diverse ensembles [47] or diverse models for parameter averaging [74]. Thus, B-HFL proposes adding dataflows directly between training nodes (e.g., clients and the root) while using the underlying hierarchical communication structure, like the "residual" connection in ResNet [29]. For example, the system could allow the $K$ client updates of each server with the highest absolute value to pass all the way to the root, where they may be merged via either training or adaptive optimisation with independent accumulator states. This sort of vertical connection provides highly dynamic and potentially cyclic dataflow. Another avenue worth exploring is allowing nodes, especially clients, to train asynchronously using their persistent model. This would permit clients to account for local dataset shifts using well-known techniques from the Continual Learning literature [15, 54, 45].

Algorithm 1 describes B-HFL recursively starting from the system's root. It assumes that the TRAIN and node aggregation NODEOPT procedures are provided. All variables are indexed per node and assumed to be provided by the implementation. The "residual" connections are adjacency lists between nodes and their ancestors/descendants in $\mathrm{AncRes}/\mathrm{DescRes}$.

**Algorithm 1** Recursive algorithm for a generic version of B-HFL. Each node $q \in Q$ has an associated persistent model $W_q$, number of rounds $T_q$, child nodes $C_q$, leaf-to-root learning rate $\eta^{\uparrow}$, root-to-leaf learning rate $\eta^{\downarrow}$. "Residual" edges are kept between nodes and their ancestors/descendants in $\mathrm{AncRes}/\mathrm{DescRes}$ with models accumulated in the lists $R^{\uparrow}$ and $R^{\downarrow}$.

1: **Require** $Q, W, T, C, \eta^{\uparrow}, \eta^{\downarrow}, \eta^{l}, \Omega$ ▷ lists indexed over all the nodes in Q
2: **Require** $R^{\uparrow}, R^{\downarrow}$ ▷ list of lists of models that a node $q$ receives from children/ancestors
3: **Require** $\mathrm{AncRes}, \mathrm{DescRes}$ ▷ list of "residual" connections to descendants/ancestors
4: **Require** TRAIN, NODEOPT, SELECTRESIDUALS
5: **procedure** EXECUTENODE($\phi, q$)
6: **if** $q = \emptyset$ **then return** $\emptyset$ ▷ error checking
7: $\theta_0 \leftarrow W_q$ ▷ handle root
8: **if** $\phi \neq \emptyset$ **then**
9: $\theta_0 \leftarrow \text{NODEOPT}(q, W_0, [\phi], R_q^{\downarrow}, q, \eta_q^{\downarrow})$ ▷ aggregate parent $[\phi]$ and "residuals"
10: **for each** round $t \leftarrow 0, \ldots, T_q - 1$ **do**
11: **for each** node $d \in \mathrm{DescRes}_q$ **do**
12: $R_d^{\downarrow} \leftarrow [\theta_t]$
13: $S \leftarrow$Sample a subset from $q$'s set of children $C_q$
14: **for each** node $c \in S$ **do**
15: $\theta_t^c \leftarrow \text{EXECUTENODE}(\theta_t, c)$ ▷ sync/async
16: **for each** node $a \in \mathrm{AncRes}_q$ **do**
17: $R_a^{\uparrow} \leftarrow \text{SELECTRESIDUALS}(q, [\theta_t^c \ \forall c \in S])$
18: $\theta_t' \leftarrow \text{NODEOPT}(q, \theta_t, [\theta_t^c \ \forall c \in S], R_q^{\uparrow}, \eta_q^{\uparrow})$ ▷ aggregate children and "residuals"
19: $\theta_{t+1} = \text{TRAIN}(\theta_t', \Omega_q, \eta_q^l)$ ▷ train (sync/async) parameters on node data
20: $W_q \leftarrow \theta_{T_q}$ ▷ update persistent node model
21: **return** $\theta_{T_q}$
22: EXECUTENODE($\phi = \emptyset, q = root$)

In its natural language form, Algorithm 1 operates as follows:

1. For the root, use the persistent model as the initial federated model $\theta_0$. [Line 7]
2. **Root-to-leaf aggregation:** UseNODEOPT to aggregate the persistent node model with the parent model $\phi$ and those in "residual" connections from ancestors $R_q^{\downarrow}$ using $\eta_q^{\downarrow}$. [Line 9]
3. Begin executing federated rounds. [Line 10]
4. Add ancestor model $\theta_t$ to $R_d^{\downarrow}$ for descendants with "residual" connections. [Line 11 to 12]
5. Sample node subset $S$ for execution. For edge servers, $|S|$ would equal the client cohort size. For non-edge servers, $S = C_q$ while for a leaf node (client) $S = \emptyset$. [Line 13]
6. Recursively execute nodes in the subtree of selected children, sending $\theta_t$. [Line 14 to 15]

7. Select and send children models $\theta_t^c$ to $R_a^{\uparrow}$ for ancestors with "residual" connections. [Line 16 to 17]

8. **Leaf-to-root aggregation:** UseNODEOPT to aggregate $\theta_t$ with the children models $[\theta_t^c \ \forall c \in S]$ and those in "residual" connections from descendants $R_q^{\uparrow}$ using $\eta^{\uparrow}$. [Line 18]

9. Train $\theta_t$ on the potentially empty dataset $\Omega_q$ using local learning rate $\eta_q^l$. *This is where edge clients and servers with proxy datasets would execute training.* [Line 19]

10. After federated training, update the persistent model $W_q$ with the most recent federated model $\theta_{T_q}$ and then return $\theta_{T_q}$.[Line 20 to 21]

The synchronicity of TRAIN is defined concerning the execution of child nodes. If training is synchronous, it must complete before child nodes begin execution. If async, the model sent to a child would be $\theta_t'$ prior to training, and the post-training $\theta_{t+1}$ would be used during leaf-to-root aggregation if completed. When async training is used, it must be accounted for during the aggregation procedure with a potential staleness factor.

"Residual" connections from descendants to ancestors may send multiple children models (e.g., the $K$ models representing the largest updates) directly or after a "residual" aggregation procedure which merges them. On the other hand, "residual" connections from ancestors to descendants only need to send one model. A relevant example of a NODEOPT procedure is FedOPT (Eq. (2.4)) [75]. FedOPT can be adapted to handle "residual" connections by adding a second accumulator state and averaging the input from the "residuals". Bidirectional Hierarchical FL may bring several potential benefits:

1. It can accommodate nodes with different aggregation methods, learning rates, and dynamic optimiser states for leaf-to-root and root-to-leaf aggregation. Similarly to the number of rounds, aggregation parameters may be independent or set per tree or level.

2. Using small cohorts for edge-servers avoids the issue of decreasing pseudo-gradients norms noticed by Charles et al. [10], as does clustering clients during edge-server assignment.

3. While persistent local models are known to work well in cross-silo FL, this hierarchical structure makes them relevant in cross-device settings by potentially allowing a more significant number of clients to be sampled per round, thus visiting them more than once.

4. Can naturally integrate Secure Aggregation [7, 39] at the level of each edge-server. As first noted by Bonawitz et al. [8], this reduces the additional communication cost of training $C$ clients with Secure Aggregation from $\mathcal{O}(C^2)$ to $\mathcal{O}(C^2/M)$, where $M$ is the number of edge-servers. Secure Aggregation and Differential Privacy [89] only need to be applied at the lowest level of the tree.

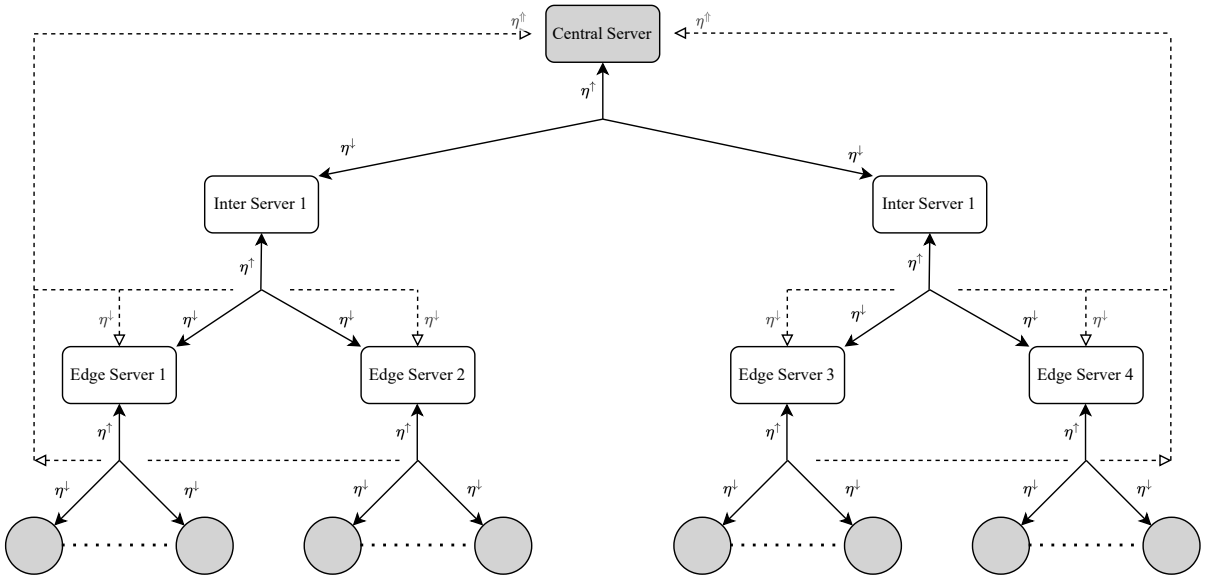



Figure 3.1: Example B-HFL system. Solid lines represent communication links, and dashed lines represent conceptual "residual" connections. Nodes capable of training, such as clients and the central server with proxy data, are in grey. When model parameters propagate up, nodes merge incoming pseudo-gradients and update their model with learning rate $\eta^{\uparrow}$. The same happens when parameters flow from parents to child nodes with learning rate $\eta^{\downarrow}$. Since the dashed lines communicate $0$ to $K$ models, $\eta^{\Uparrow}$ may represent $0$ to $K$ aggregations using a $\eta^{\uparrow}$ learning rate.

## 3.3 Example System

An example of a B-HFL system, which would be the first deliverable for the second year of my PhD, may be seen in Fig. 3.1. The central server controls a proxy dataset used to train after it performs aggregation. Intermediary servers perform only aggregation. Servers send updates to the parent after every round.

Each node, including the clients, runs at-least two stateful FedOPT server optimisers with separate learning rates, one for the leaf-to-root aggregation and one for parent aggregation. Even if the same leaf-to-root learning rate $\eta^{\uparrow}$ and root-to-leaf learning rate $\eta^{\downarrow}$ were to be used for all nodes in the tree or at a given level, the independent server optimiser states would distinguish the aggregation procedure of their node based on historical trends. The central server uses model interpolation with a mixture parameter [16] adapted to its proxy data to merge child updates.

The "residual" connections serve different functions between the leaf-to-root and root-to-leaf stages. For the upward stage, they collect the $K = 1$ client update with the highest absolute value, thus sending one additional model to the central server per edge server. For the downward stage, they allow the edge servers to directly benefit from the central server's training without relying on averaged intermediate models. While this last component is somewhat superfluous in the small hierarchy shown by Fig. 3.1, it may prove relevant for profound structures. For example, in deep hierarchies, parameters that receive extra training at the central server might get averaged several times before reaching the edge servers and thus influencing the leaves.

## 3.4 Research Directions

Algorithm 1 imposes sufficient structure to create a new family of hierarchical FL systems, opening a series of research directions. These directions can be primarily divided into three types: (a) node aggregation procedures filling in the NODEOPT function, (b) "residual" selection procedures filling in the SELECTRESIDUALS function, and (c) clustering algorithms which decide how client nodes are divided across the physical or virtual edge-servers. Both node aggregation and "residual" selection are expected to be set for types of nodes or levels of the tree, as allowing each node to have a separate aggregation procedure would be difficult to manage. Regarding clustering, relations imposed by the physical communication links and purely conceptual ones must be distinguished as the former are unalterable. Importantly, a physical edge server may contain multiple virtual nodes to which clients may be assigned.

**Node Aggregation Procedures** Relevant node aggregation procedures can either be those developed for standard FL or procedures that take advantage of the unique tree structure or available proxy data. FedOPT [75] and Iterative Moving Averaging [101] are examples of standard FL algorithms that offer unique *implicit* benefits for this hierarchical structure because they maintain stateful accumulators or previous models, respectively, which permit every server to be distinguished. The smaller number of children of non-edge servers may also enable costly aggregation procedures. For example, internal nodes having potential proxy data allows them to use model interpolations via either training regularisation [51] or by using adaptive interpolation rates, as proposed by Deng et al. [16], optimised to the proxy data in order to balance the influence of updates from child nodes, parents nodes, and from proxy training. Aggregation procedures can also *explicitly* consider the links between nodes. A simple example of this second type is aggregation considering the distance between an ancestor and a "residual" descendant. More complex procedures may combine the known hierarchical topology with similarity metrics to create a distance matrix between internal node models and perform graph message passing [92], as proposed by Chen et al. [11], to interpolate parameters between all internal nodes.

**"Residual" Selection Procedures** For "residuals" to be beneficial during aggregation, they must contain information not already evident in their parent's model. As mentioned, it is well-known that averaging is an imperfect means of aggregating models trained on Non-IID data [65, 10, 52, 48] as the directions of different pseudo-gradients may conflict. Additionally, maintaining some degree of diversity in ensembles [47] and parameter averages [74] is known to be potentially beneficial. As such, in the case of leaf-to-root "residuals", parameters may be selected based on simple metrics like an $l_n$ norm, their per-sample loss, or their relation to each other (e.g., cosine similarity). If necessary for communication efficiency, an average update from the top-K outlier gradients, according to the metric, may substitute selecting multiple "residuals".

**Clustering Algorithms** Since the connection between cloud and edge servers or multiple layers of intermediary servers is fixed by physical communication links, the underlying communication hierarchy must also remain fixed, making edge reassignments impossible. However, each physical server can create arbitrarily many internal virtual nodes if it adheres to the structure in Algorithm 1. Thus, each physical server is assumed to have the ability to create internal node hierarchies, and edge servers are assumed to have the ability to create clusters for clients within their geographic area. These properties could be exploited for the creation of a B-HFL dynamic clustering system similar to Auxo [58] where edge-servers would split into multiple virtual nodes to create clusters for clients to be assigned to; however, unlike Auxo, these clusters would maintain a common parent node and thus share information.

**Criteria for Success** Given these research directions, it is essential to consider their impact. The proposed research will be successful in the near term if it leads to published work based on the example system in Section 3.3, at ICLR, MLSys, NeurIPS, or an equivalent conference, as detailed in Section 5.1. Within the scope of the whole PhD, B-HFL would be successful if its structure allows for faster convergence to the global optimum by avoiding the diminishing effects of large cohorts, creating a more performant global model, or constructing excellent cluster models through inter-cluster communication. Alternative criteria include achieving similar performance to standard FL algorithms with lower communication costs or obtaining evidence that "residual" connections and bidirectional partial aggregation are independently beneficial.

**Potential Risks** Considering the previously mentioned criteria, potential risks to the methodology include: (a) the continued proliferation of large models trained on public corpora, which are challenging to federate without extensive GPU resources being available to the clients, (b) a lack of adaptivity in the hierarchical structure. For the first risk, if large models continue to proliferate, efficiently training them in a federated fashion via model splitting would need to be added as a primary research direction. For the second, if the proposed B-HFL structure is too rigid to manipulate and would be thus difficult to apply in a practical FL scenario, it would be preferable to refocus the work entirely on inter-cluster communication through a parent node where the depth of the hierarchy never exceeds that of the example system in Fig. 3.1.

# Chapter 4

# Completed Work

The proposed research for my PhD emerged from research on Personalised Federated Learning and Hierarchical Federated Learning I began during my MPhil in Advanced Computer Science and the first year of my PhD. All the mentioned works are available as appendices.

## 4.1 Fairness and Personalisation

Iacob et al. [35] investigated the trade-off between generalisation and personalisation, which is at the heart of the proposed research, from the perspectives of Fair Federated Learning and its interactions with local adaptation (fine-tuning) of the federated model post-training. Since Fair Federated Learning attempts to construct a more uniform accuracy distribution for the federated model over the local test sets of clients, the expectation was to either reduce the need for local adaptation or to provide a better starting point from which to carry it out. The experimental results showed that Fair FL brings no benefits and potential downsides towards later personalisation and led to the proposal of a Personalisation-aware FL algorithm that attempts to anticipate the common regularises used during fine-tuning throughout the FL process.

Personalisation-aware FL functions in two phases; first, it allows standard FL training to progress unimpeded. After near-convergence, it injects standard personalisation regularises such as Knowledge Distillation [30, 99] or Elastic-weight Consolidation [45] into the client loss function, where the reference model is taken to be the federated model from the start of the round, with the intent of pre-empting the loss used during the final local adaptation phase. Such regularisers allow the model to learn from the distributions of highly heterogeneous clients without harming performance on the overall federated network. This enables a better accuracy distribution over clients without harming the average performance, as Fair FL is known to do [49, 50]. While more effective than Fair FL, this regularisation-based approach is still limited by the goal of training a single global model without any intermediary level of personalisation between the federated model and fine-tuned local models.

## 4.2 Hierarchical Multimodal Federated HAR

Iacob et al. [36] evaluated the performance of Federated Human Activity Recognition [81] when trained using multimodal data gathered from different sensor types at increasing levels of privacy. It showed that grouping clients based on the type of sensor that produced their training set effectively mitigated the impacts of privacy being required at a human subject, environment, and sensor level simultaneously. It was a direct precursor to B-HFL as it relied on a two-tiered model structure where each client trained both a group-level model and the global federated model using a mutual learning approach [99]. This work was later extended to consider the adaptability of such two-tiered systems to the addition of a new sensor type (group) into the federation; the extension was submitted to the MobiUK symposium. Mutual learning was chosen to relate the group-level and global models since it allowed divergent architectures that only shared the output layer. However, despite its success, this training method requires clients to have a high amount of data and local epochs to train both models. The expensive nature of the procedure prompted a move towards the more flexible methods, such as adaptive partial model averaging [75], considered in the presently proposed research (Section 3.4).

## 4.3 Simulation Efficiency

Both previous works were implemented in the Flower [4] FL framework; however, the scale of experimentation required for fully validating B-HFL would be unfeasible on the publicly available simulation engine in the case of cross-device scenarios. Flower's previous Virtual Client Engine (VCE) used the Ray [68] distributed execution engine for simulation, which is designed for a few long-running ML tasks rather than the numerous and short-running client training of FL. Furthermore, Ray does not have a means of forcefully freeing GPU memory allocated by an ML framework like PyTorch without killing a process. This resulted in slow training times, instability and frequent disk spillage.

To correct such issues, I have contributed to research on a new engine that doubles Flower simulations' throughput by intelligent ML-based client placement on GPUs. Since clients in FL are heterogeneous regarding workloads, pre-processing pipelines and dataset size, this required more than mere load-balancing based on sample count. The system functions by actively placing clients to workers on specific GPUs based on a log-linear model which estimates client training time on a given piece of hardware based on historical data. The workers then perform local averaging in order to minimise communication. This system results in up to $400\%$ improvements in FL training time compared to other FL frameworks like FedScale [46], original Flower [4], and Flute [17]. The paper "High-throughput Simulation of Federated Learning via Resource-Aware Client Placement" will be submitted to MLSys.

# Chapter 5

# Plan and Timeline

The presented family of Bidirectional Hierarchical Federated Learning algorithms will be developed during the PhD period and will form part of the final PhD thesis. In addition, before the final thesis, it offers opportunities for conference publications that significantly contribute to Federated Learning. Given the novelty of FL in general and hierarchical FL in particular, there is ample room for further developments in the structure of B-HFL as the fields mature.

## 5.1 Second-year Plan

The summer period of the end of my first year of the PhD will be dedicated to implementing the example version of B-HFL in the Flower [4] FL framework affiliated with my research group. The framework is currently tuned to standard FL settings and would require heavy API modifications to execute and simulate hierarchical FL effectively. However, the previous work on group-level models for Federated Human Activity Recognition of Iacob et al. [36] and the effective FL simulation engine I contributed to can be the basis for implementing and streamlining the process.

The autumn Michaelmas Term of my second year will have as a main objective the publication of a conference paper based on the example system proposed in Section 3.3. ICLR and MLSys would be appropriate venues. Given the growing importance of LLMs, and the trade-offs recently discovered by Agarwal et al. [2] in terms of their generalisation and personalisation abilities with or without pre-trained weights, they represent a natural application for the proposed hierarchical system. Moreover, multilingual text prediction provides a naturally clustered FL application corresponding to real-world scenarios where countries have independent servers for FL and must collaborate at a continental and global level. The study would use a large multilingual BERT model [14] together with two multilingual datasets [e.g., 55, 95] for training. One dataset will be partitioned by language, and the other will be kept as a proxy dataset at the central server, as in Fig. 3.1. The study's goals would be to compare the final accuracy of each model at every level of the hierarchy on the client test sets and the centralised test set created from the proxy

dataset. The expectation would be for the model performance on the data of a specific client to be proportional to their proximity to that client in the tree. Alternatively, for the proxy test set and the union of all client test sets, accuracy should be proportional to the proximity to the central server. In addition, ablation studies on the "residual" connections, adaptive optimisation, or persistent local models will also be performed together with efficiency comparisons between synchronous and asynchronous node execution at different levels of the tree. Finally, if time allows, the paper could include other naturally-clustered tasks, such as speech recognition for multilingual data.

Following the publication of this work, a natural extension during Lent and Easter terms would be to tackle a setting where clients continuously generate and delete data with limited local storage. The example system would be extended to allow asynchronous training on all nodes, including the leaves, which run parallel to the actual FL component. Each client would generate a data stream while having a fixed internal memory to operate on during training. Real resource constraints and asynchronicity can be modelled using the Raspberry Pi FL cluster at Cambridge ML Systems. This work would be intended for MobiCom or another systems conference. The summer term would include buffer time to account for any new projects spawned from previous ones. This would then be followed up by a further literature review on clustering and model aggregation, an inspection of the code of Auxo [58], and the implementation of early prototypes for automatic clustering in B-HFL.

## 5.2 Third-year Plan

During Michaelmas, I would finalise the thesis's literature review and implement an automatic hierarchical clustering system in the vein of Auxo. If successful, this would be intended for a potential publication at ICLR or a later conference.

The beginning of the Lent term will represent the beginning of the thesis write-up considering all work done up to that point. Lent term would then involve exploring different "residual" selection criteria and complex aggregation procedures based on the hierarchical structure of the model, assuming that the B-HFL system contains natural or automatically constructed clusters. The findings would be written up for a conference or workshop if successful.

Easter term would be devoted to finishing any work leftover from Michaelmas or Lent. The rest of the time will be devoted to constructing a complete thesis outline and continuing the write-up started during the Lent term. The summer term would be dedicated to finishing the thesis.

# References


[1] Mehdi Salehi Heydar Abad, Emre Ozfatura, Deniz Gündüz, and Özgür Erçetin. Hierarchical federated learning ACROSS heterogeneous cellular networks. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*, pages 8866–8870. IEEE, 2020. doi: 10.1109/ICASSP40776.2020.9054634. URL https://doi.org/10.1109/ICASSP40776.2020.9054634. Cited on pages 14, 16, 17, and 20.

[2] Ankur Agarwal, Mehdi Rezagholizadeh, and Prasanna Parthasarathi. Practical takes on federated learning with pretrained language models. In Andreas Vlachos and Isabelle Augenstein, editors, *Findings of the Association for Computational Linguistics: EACL 2023, Dubrovnik, Croatia, May 2-6, 2023*, pages 454–471. Association for Computational Linguistics, 2023. URL https://aclanthology.org/2023.findings-eacl.34. Cited on pages 7 and 29.

[3] Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. Federated learning with personalization layers. *CoRR*, abs/1912.00818, 2019. URL http://arxiv.org/abs/1912.00818. Cited on pages 6, 14, 15, and 17.

[4] Daniel J. Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Titouan Parcollet, and Nicholas D. Lane. Flower: A friendly federated learning research framework. *CoRR*, abs/2007.14390, 2020. URL https://arxiv.org/abs/2007.14390. Cited on pages 5, 28, and 29.

[5] Abhishek Bhowmick, John C. Duchi, Julien Freudiger, Gaurav Kapoor, and Ryan Rogers. Protection against reconstruction and its applications in private federated learning. *CoRR*, abs/1812.00984, 2018. URL http://arxiv.org/abs/1812.00984. Cited on page 10.

[6] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ B. Altman, Simran Arora, and et al. On the opportunities and risks of foundation models. *CoRR*, abs/2108.07258, 2021. URL https://arxiv.org/abs/2108.07258. Cited on page 5.

[7] Kallista A. Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth.

Practical secure aggregation for federated learning on user-held data. *CoRR*, abs/1611.04482, 2016. URL https://www.usenix.org/conference/osdi18/presentation/nishihara. Cited on pages 10, 11, 13, and 23.

[8] Kallista A. Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloé Kiddon, Jakub Konečný, Stefano Mazzocchi, Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. Towards federated learning at scale: System design. In Ameet Talwalkar, Virginia Smith, and Matei Zaharia, editors, *Proceedings of Machine Learning and Systems 2019, MLSys 2019, Stanford, CA, USA, March 31 - April 2, 2019*. mlsys.org, 2019. URL https://proceedings.mlsys.org/book/271.pdf. Cited on pages 6, 10, 13, 15, 16, and 23.

[9] Christopher Briggs, Zhong Fan, and Peter Andras. Federated learning with hierarchical clustering of local updates to improve training on non-iid data. In *2020 International Joint Conference on Neural Networks, IJCNN 2020, Glasgow, United Kingdom, July 19-24, 2020*, pages 1–9. IEEE, 2020. doi: 10.1109/IJCNN48605.2020.9207469. URL https://doi.org/10.1109/IJCNN48605.2020.9207469. Cited on pages 15 and 17.

[10] Zachary Charles, Zachary Garrett, Zhouyuan Huo, Sergei Shmulyian, and Virginia Smith. On large-cohort training for federated learning. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 20461–20475, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/ab9ebd57177b5106ad7879f0896685d4-Abstract.html. Cited on pages 6, 11, 12, 13, 15, 19, 23, and 25.

[11] Fengwen Chen, Guodong Long, Zonghan Wu, Tianyi Zhou, and Jing Jiang. Personalized federated learning with a graph. In Luc De Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 2575–2582. ijcai.org, 2022. doi: 10.24963/ijcai.2022/357. URL https://doi.org/10.24963/ijcai.2022/357. Cited on page 25.

[12] Yujing Chen, Yue Ning, Martin Slawski, and Huzefa Rangwala. Asynchronous online federated learning for edge devices with non-iid data. In Xintao Wu, Chris Jermaine, Li Xiong, Xiaohua Hu, Olivera Kotevska, Siyuan Lu, Weija Xu, Srinivas Aluru, Chengxiang Zhai, Eyhab Al-Masri, Zhiyuan Chen, and Jeff Saltz, editors, *2020 IEEE International Conference on Big Data (IEEE BigData 2020), Atlanta, GA, USA, December 10-13, 2020*, pages 15–24. IEEE, 2020. doi: 10.1109/BigData50022.2020.9378161. URL

https://doi.org/10.1109/BigData50022.2020.9378161. Cited on pages 6 and 13.

[13] Gary Cheng, Karan N. Chadha, and John C. Duchi. Fine-tuning is fine in federated learning. *CoRR*, abs/2108.07313, 2021. URL https://arxiv.org/abs/2108.07313. Cited on pages 14 and 17.

[14] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8440–8451. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.acl-main.747. URL https://doi.org/10.18653/v1/2020.acl-main.747. Cited on page 29.

[15] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7):3366–3385, 2022. doi: 10.1109/TPAMI.2021.3057446. Cited on pages 10 and 21.

[16] Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Adaptive personalized federated learning. *CoRR*, abs/2003.13461, 2020. URL https://arxiv.org/abs/2003.13461. Cited on pages 6, 7, 14, 15, 17, 21, 24, and 25.

[17] Dimitrios Dimitriadis, Mirian Hipolito Garcia, Daniel Madrigal, Andre Manoel, and Robert Sim. Flute: A scalable, extensible framework for high-performance federated learning simulations, March 2022. Cited on pages 5 and 28.

[18] John C. Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12:2121–2159, 2011. doi: 10.5555/1953048.2021068. URL https://dl.acm.org/doi/10.5555/1953048.2021068. Cited on page 12.

[19] Chen Dun, Mirian Hipolito Garcia, Chris Jermaine, Dimitrios Dimitriadis, and Anastasios Kyrillidis. Efficient and light-weight federated learning via asynchronous distributed dropout. In Francisco J. R. Ruiz, Jennifer G. Dy, and Jan-Willem van de Meent, editors, *International Conference on Artificial Intelligence and Statistics, 25-27 April 2023, Palau de Congressos, Valencia, Spain*, volume 206 of *Proceedings of Machine Learning Research*, pages 6630–6660. PMLR, 2023. URL https://proceedings.mlr.press/v206/dun23a.html. Cited on pages 6 and 13.

[20] Cynthia Dwork. Differential privacy. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II*, volume 4052 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2006. doi: 10.1007/11787006_1. URL https://doi.org/10.1007/11787006_1. Cited on page 10.

[21] Robert French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3:128–135, 05 1999. doi: 10.1016/S1364-6613(99)01294-2. Cited on page 14.

[22] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. Inverting gradients - how easy is it to break privacy in federated learning? In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/c4ede56bbd98819ae6112b20ac6bf145-Abstract.html. Cited on page 10.

[23] Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An efficient framework for clustered federated learning. *IEEE Trans. Inf. Theory*, 68(12):8076–8091, 2022. doi: 10.1109/TIT.2022.3192506. URL https://doi.org/10.1109/TIT.2022.3192506. Cited on pages 14, 15, 16, and 17.

[24] Google. Tensorflow federated, 2019. URL https://www.tensorflow.org/federated. Cited on page 5.

[25] Neel Guha, Ameet Talwalkar, and Virginia Smith. One-shot federated learning. *CoRR*, abs/1902.11175, 2019. URL http://arxiv.org/abs/1902.11175. Cited on page 20.

[26] Filip Hanzely and Peter Richtárik. Federated learning of a mixture of global and local models. *CoRR*, abs/2002.05516, 2020. URL https://arxiv.org/abs/2002.05516. Cited on pages 6, 7, and 14.

[27] Andrew Hard, Kanishka Rao, Rajiv Mathews, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. Federated learning for mobile keyboard prediction. *CoRR*, abs/1811.03604, 2018. URL http://arxiv.org/abs/1811.03604. Cited on pages 5, 6, and 16.

[28] Chaoyang He, Songze Li, Jinhyun So, Mi Zhang, Hongyi Wang, Xiaoyang Wang, Praneeth Vepakomma, Abhishek Singh, Hang Qiu, Li Shen, Peilin Zhao, Yan Kang, Yang Liu, Ramesh Raskar, Qiang Yang, Murali Annavaram, and Salman Avestimehr. Fedml: A

research library and benchmark for federated machine learning. *CoRR*, abs/2007.13518, 2020. URL https://arxiv.org/abs/2007.13518. Cited on page 5.

[29] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society, 2016. doi: 10.1109/CVPR.2016.90. URL https://doi.org/10.1109/CVPR.2016.90. Cited on pages 7 and 21.

[30] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015. URL http://arxiv.org/abs/1503.02531. Cited on page 27.

[31] Kevin Hsieh, Amar Phanishayee, Onur Mutlu, and Phillip B. Gibbons. The non-iid data quagmire of decentralized machine learning. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 4387–4398. PMLR, 2020. URL http://proceedings.mlr.press/v119/hsieh20a.html. Cited on page 9.

[32] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *CoRR*, abs/1909.06335, 2019. URL http://arxiv.org/abs/1909.06335. Cited on page 12.

[33] Li Huang, Andrew L. Shea, Huining Qian, Aditya Masurkar, Hao Deng, and Dianbo Liu. Patient clustering improves efficiency of federated machine learning to predict mortality and hospital stay time using distributed electronic medical records. *J. Biomed. Informatics*, 99, 2019. doi: 10.1016/j.jbi.2019.103291. URL https://doi.org/10.1016/j.jbi.2019.103291. Cited on page 15.

[34] Dzmitry Huba, John Nguyen, Kshitiz Malik, Ruiyu Zhu, Mike Rabbat, Ashkan Yousefpour, Carole-Jean Wu, Hongyuan Zhan, Pavel Ustinov, Harish Srinivas, Kaikai Wang, Anthony Shoumikhin, Jesik Min, and Mani Malek. PAPAYA: practical, private, and scalable federated learning. In Diana Marculescu, Yuejie Chi, and Carole-Jean Wu, editors, *Proceedings of Machine Learning and Systems 2022, MLSys 2022, Santa Clara, CA, USA, August 29 - September 1, 2022*. mlsys.org, 2022. URL https://proceedings.mlsys.org/paper/2022/hash/f340f1b1f65b6df5b5e3f94d95b11daf-Abstract.html. Cited on pages 5, 6, 10, 13, 17, and 20.

[35] Alex Iacob, Pedro Porto Buarque Gusmão, and Nicholas Lane. Can fair federated learning reduce the need for personalisation? In *Proceedings of the 3rd Workshop on Machine Learning and Systems*, EuroMLSys '23, page 131–139, New York, NY, USA, 2023.

Association for Computing Machinery. ISBN 9798400700842. doi: 10.1145/3578356.3592592. URL https://doi.org/10.1145/3578356.3592592. Cited on pages 7 and 27.

[36] Alex Iacob, Pedro Porto Buarque Gusmão, Nicholas Lane, Armand Koupai, Mohammud Bocus, Raul Santos-Rodriguez, Robert Piechocki, and Ryan McConville. Privacy in multimodal federated human activity recognition. In *Proceedings of the 3rd On-Device Intelligence Workshop*, MLSys '23, 2023. URL https://sites.google.com/g.harvard.edu/on-device-workshop-23/home?authuser=0. Cited on pages 7, 28, and 29.

[37] Eunjeong Jeong, Seungeun Oh, Hyesung Kim, Jihong Park, Mehdi Bennis, and Seong-Lyun Kim. Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data. *CoRR*, abs/1811.11479, 2018. URL http://arxiv.org/abs/1811.11479. Cited on page 10.

[38] Joe H. Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244, 1963. doi: 10.1080/01621459.1963.10500845. URL https://www.tandfonline.com/doi/abs/10.1080/01621459.1963.10500845. Cited on page 15.

[39] Swanand Kadhe, Nived Rajaraman, Onur Ozan Koyluoglu, and Kannan Ramchandran. Fastsecagg: Scalable secure aggregation for privacy-preserving federated learning. *CoRR*, abs/2009.11248, 2020. URL https://arxiv.org/abs/2009.11248. Cited on pages 10 and 23.

[40] Peter Kairouz, Brendan Mcmahan, Shuang Song, Om Thakkar, Abhradeep Thakurta, and Zheng Xu. Practical and private (deep) learning without sampling or shuffling. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 5213–5225. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/kairouz21b.html. Cited on pages 5 and 10.

[41] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, and et al. Advances and open problems in federated learning. *Found. Trends Mach. Learn.*, 14 (1-2):1–210, 2021. doi: 10.1561/2200000083. URL https://doi.org/10.1561/2200000083. Cited on pages 5, 6, and 9.

[42] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *5th International Conference on Learning Representations, ICLR 2017,*

*Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=H1oyRlYgg. Cited on page 11.

[43] Ahmed Khaled, Konstantin Mishchenko, and Peter Richtárik. Tighter theory for local SGD on identical and heterogeneous data. In Silvia Chiappa and Roberto Calandra, editors, *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*, volume 108 of *Proceedings of Machine Learning Research*, pages 4519–4529. PMLR, 2020. URL http://proceedings.mlr.press/v108/bayoumi20a.html. Cited on page 6.

[44] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980. Cited on page 12.

[45] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017. doi: 10.1073/pnas.1611835114. URL https://www.pnas.org/doi/abs/10.1073/pnas.1611835114. Cited on pages 7, 10, 14, 21, and 27.

[46] Fan Lai, Yinwei Dai, Sanjay Sri Vallabh Singapuram, Jiachen Liu, Xiangfeng Zhu, Harsha V. Madhyastha, and Mosharaf Chowdhury. Fedscale: Benchmarking model and system performance of federated learning at scale. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 11814–11827. PMLR, 2022. URL https://proceedings.mlr.press/v162/lai22a.html. Cited on pages 5 and 28.

[47] Stefan Lee, Senthil Purushwalkam, Michael Cogswell, Viresh Ranjan, David J. Crandall, and Dhruv Batra. Stochastic multiple choice learning for training diverse deep ensembles. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 2119–2127, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/20d135f0f28185b84a4cf7aa51f29500-Abstract.html. Cited on pages 21 and 25.

[48] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In Inderjit S. Dhillon, Dimitris S. Papailiopoulos, and Vivienne Sze, editors, *Proceedings of Machine Learning and Systems 2020, MLSys 2020, Austin, TX, USA, March 2-4, 2020*. mlsys.org, 2020. URL https://proceedings.mlsys.org/book/316.pdf. Cited on pages 10, 13, and 25.

[49] Tian Li, Maziar Sanjabi, Ahmad Beirami, and Virginia Smith. Fair resource allocation in federated learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=ByexElSYDr. Cited on pages 14 and 27.

[50] Tian Li, Ahmad Beirami, Maziar Sanjabi, and Virginia Smith. Tilted empirical risk minimization. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=K5YasWXZT3O. Cited on page 27.

[51] Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through personalization. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 6357–6368. PMLR, 2021. URL http://proceedings.mlr.press/v139/li21h.html. Cited on pages 6, 7, 14, 15, 17, 21, and 25.

[52] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=HJxNAnVtDS. Cited on pages 9 and 25.

[53] Zhengyang Li, Shijing Si, Jianzong Wang, and Jing Xiao. Federated split BERT for heterogeneous text classification. In *International Joint Conference on Neural Networks, IJCNN 2022, Padua, Italy, July 18-23, 2022*, pages 1–8. IEEE, 2022. doi: 10.1109/IJCNN55064.2022.9892845. URL https://doi.org/10.1109/IJCNN55064.2022.9892845. Cited on pages 14 and 15.

[54] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017. Cited on pages 10 and 21.

[55] Yaobo Liang, Nan Duan, Yeyun Gong, Ning Wu, Fenfei Guo, Weizhen Qi, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, Xiaodong Fan, Bruce Zhang, Rahul Agrawal, Edward Cui, Sining Wei, Taroon Bharti, Ying Qiao, Jiun-Hung Chen, Winnie Wu, Shuguang

Liu, Fan Yang, Rangan Majumder, and Ming Zhou. XGLUE: A new benchmark dataset for cross-lingual pre-training, understanding and generation. *CoRR*, abs/2004.01401, 2020. URL https://arxiv.org/abs/2004.01401. Cited on page 29.

[56] Bill Yuchen Lin, Chaoyang He, Zihang Ze, Hulin Wang, Yufen Hua, Christophe Dupuy, Rahul Gupta, Mahdi Soltanolkotabi, Xiang Ren, and Salman Avestimehr. Fednlp: Benchmarking federated learning methods for natural language processing tasks. In Marine Carpuat, Marie-Catherine de Marneffe, and Iván Vladimir Meza Ruíz, editors, *Findings of the Association for Computational Linguistics: NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 157–175. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.findings-naacl.13. URL https://doi.org/10.18653/v1/2022.findings-naacl.13. Cited on page 5.

[57] Yujun Lin, Song Han, Huizi Mao, Yu Wang, and Bill Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=SkhQHMW0W. Cited on page 17.

[58] Jiachen Liu, Fan Lai, Yinwei Dai, Aditya Akella, Harsha V. Madhyastha, and Mosharaf Chowdhury. Auxo: Heterogeneity-mitigating federated learning via scalable client clustering. *CoRR*, abs/2210.16656, 2022. doi: 10.48550/arXiv.2210.16656. URL https://doi.org/10.48550/arXiv.2210.16656. Cited on pages 7, 14, 15, 16, 17, 26, and 30.

[59] Lumin Liu, Jun Zhang, Shenghui Song, and Khaled B. Letaief. Client-edge-cloud hierarchical federated learning. In *2020 IEEE International Conference on Communications, ICC 2020, Dublin, Ireland, June 7-11, 2020*, pages 1–6. IEEE, 2020. doi: 10.1109/ICC40277.2020.9148862. URL https://doi.org/10.1109/ICC40277.2020.9148862. Cited on pages 14, 16, 17, and 20.

[60] Stuart P. Lloyd. Least squares quantization in PCM. *IEEE Trans. Inf. Theory*, 28(2):129–136, 1982. doi: 10.1109/TIT.1982.1056489. URL https://doi.org/10.1109/TIT.1982.1056489. Cited on page 15.

[61] Siqi Luo, Xu Chen, Qiong Wu, Zhi Zhou, and Shuai Yu. HFEL: joint edge association and resource allocation for cost-efficient hierarchical federated edge learning. *IEEE Trans. Wirel. Commun.*, 19(10):6535–6548, 2020. doi: 10.1109/TWC.2020.3003744. URL https://doi.org/10.1109/TWC.2020.3003744. Cited on page 17.

[62] Lingjuan Lyu, Jiangshan Yu, Karthik Nandakumar, Yitong Li, Xingjun Ma, Jiong Jin, Han Yu, and Kee Siong Ng. Towards fair and privacy-preserving federated deep models.

*IEEE Trans. Parallel Distributed Syst.*, 31(11):2524–2541, 2020. doi: 10.1109/TPDS.2020.2996273. URL https://doi.org/10.1109/TPDS.2020.2996273. Cited on pages 5 and 10.

[63] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7765–7773, 2017. Cited on page 10.

[64] Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning. *CoRR*, abs/2002.10619, 2020. URL https://arxiv.org/abs/2002.10619. Cited on pages 14, 15, 17, and 21.

[65] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In Aarti Singh and Xiaojin (Jerry) Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR, 2017. URL http://proceedings.mlr.press/v54/mcmahan17a.html. Cited on pages 5, 9, 17, and 25.

[66] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=BJ0hF1Z0b. Cited on pages 10 and 11.

[67] Naram Mhaisen, Alaa Awad Abdellatif, Amr Mohamed, Aiman Erbad, and Mohsen Guizani. Optimal user-edge assignment in hierarchical federated learning based on statistical properties and network topology constraints. *IEEE Trans. Netw. Sci. Eng.*, 9 (1):55–66, 2022. doi: 10.1109/TNSE.2021.3053588. URL https://doi.org/10.1109/TNSE.2021.3053588. Cited on pages 10, 14, and 17.

[68] Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I. Jordan, and Ion Stoica. Ray: A distributed framework for emerging AI applications. In Andrea C. Arpaci-Dusseau and Geoff Voelker, editors, *13th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2018, Carlsbad, CA, USA, October 8-10, 2018*, pages 561–577. USENIX Association, 2018. Cited on page 28.

[69] John Nguyen, Kshitiz Malik, Hongyuan Zhan, Ashkan Yousefpour, Mike Rabbat, Mani Malek, and Dzmitry Huba. Federated learning with buffered asynchronous aggregation. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, editors, *International*

*Conference on Artificial Intelligence and Statistics, AISTATS 2022, 28-30 March 2022, Virtual Event*, volume 151 of *Proceedings of Machine Learning Research*, pages 3581–3607. PMLR, 2022. URL https://proceedings.mlr.press/v151/nguyen22b.html. Cited on pages 5, 6, 13, 17, and 20.

[70] Xiaomin Ouyang, Zhiyuan Xie, Jiayu Zhou, Jianwei Huang, and Guoliang Xing. Clusterfl: a similarity-aware federated learning system for human activity recognition. In Suman Banerjee, Luca Mottola, and Xia Zhou, editors, *MobiSys '21: The 19th Annual International Conference on Mobile Systems, Applications, and Services, Virtual Event, Wisconsin, USA, 24 June - 2 July, 2021*, pages 54–66. ACM, 2021. doi: 10.1145/3458864.3467681. URL https://doi.org/10.1145/3458864.3467681. Cited on page 5.

[71] Sarthak Pati, Ujjwal Baid, Brandon Edwards, Micah J. Sheller, Hans Shih-Han Wang, G. Anthony Reina, Patrick Foley, Alexey Gruzdev, Deepthi Karkada, Christos Davatzikos, Chiharu Sako, Satyam Ghodasara, Michel Bilello, Suyash Mohan, Philipp Vollmuth, Gianluca Brugnara, Chandrakanth J. Preetha, Felix Sahm, Klaus H. Maier-Hein, Maximilian Zenk, Martin Bendszus, Wolfgang Wick, Evan Calabrese, Jeffrey D. Rudie, Javier E. Villanueva-Meyer, Soonmee Cha, Madhura Ingalhalikar, Manali Jadhav, Umang Pandey, Jitender Saini, John Garrett, Matthew Larson, Robert Jeraj, Stuart Currie, Russell Frood, Kavi Fatania, Raymond Y. Huang, Ken Chang, Carmen Balaña Quintero, Jaume Capellades, Josep Puig, Johannes Trenkler, Josef Pichler, Georg Necker, Andreas Haunschmidt, Stephan Meckel, Gaurav Shukla, Spencer Liem, Gregory S. Alexander, and et al. Federated learning enables big data for rare cancer boundary detection. *CoRR*, abs/2204.10836, 2022. doi: 10.48550/arXiv.2204.10836. URL https://doi.org/10.48550/arXiv.2204.10836. Cited on page 5.

[72] Matthias Paulik, Matt Seigel, Henry Mason, Dominic Telaar, Joris Kluivers, Rogier C. van Dalen, Chi Wai Lau, Luke Carlson, Filip Granqvist, Chris Vandevelde, Sudeep Agarwal, Julien Freudiger, Andrew Byde, Abhishek Bhowmick, Gaurav Kapoor, Si Beaumont, Áine Cahill, Dominic Hughes, Omid Javidbakht, Fei Dong, Rehan Rishi, and Stanley Hung. Federated evaluation and tuning for on-device personalization: System design & applications. *CoRR*, abs/2102.08503, 2021. URL https://arxiv.org/abs/2102.08503. Cited on page 5.

[73] Le Trieu Phong, Yoshinori Aono, Takuya Hayashi, Lihua Wang, and Shiho Moriai. Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Trans. Inf. Forensics Secur.*, 13(5):1333–1345, 2018. doi: 10.1109/TIFS.2017.2787987. URL https://doi.org/10.1109/TIFS.2017.2787987. Cited on page 10.

[74] Alexandre Ramé, Matthieu Kirchmeyer, Thibaud Rahier, Alain Rakotomamonjy, Patrick Gallinari, and Matthieu Cord. Diverse weight av-

eraging for out-of-distribution generalization. In *NeurIPS*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/46108d807b50ad4144eb353b5d0e8851-Abstract-Conference.html. Cited on pages 21 and 25.

[75] Sashank J. Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and Hugh Brendan McMahan. Adaptive federated optimization. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=LkFG3lB13U5. Cited on pages 7, 12, 23, 25, and 28.

[76] Nicola Rieke, Jonny Hancox, Wenqi Li, Fausto Milletari, Holger Roth, Shadi Albarqouni, Spyridon Bakas, Mathieu N. Galtier, Bennett A. Landman, Klaus H. Maier-Hein, Sébastien Ourselin, Micah J. Sheller, Ronald M. Summers, Andrew Trask, Daguang Xu, Maximilian Baust, and M. Jorge Cardoso. The future of digital health with federated learning. *CoRR*, abs/2003.08119, 2020. URL https://arxiv.org/abs/2003.08119. Cited on page 5.

[77] Sebastian Ruder. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747, 2016. URL http://arxiv.org/abs/1609.04747. Cited on page 12.

[78] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE Trans. Neural Networks Learn. Syst.*, 32(8):3710–3722, 2021. doi: 10.1109/TNNLS.2020.3015958. URL https://doi.org/10.1109/TNNLS.2020.3015958. Cited on pages 15 and 17.

[79] Micah J. Sheller, Brandon Edwards, G. Anthony Reina, Jason Martin, Sarthak Pati, Aikaterini Kotrotsou, Mikhail Milchenko, Weilin Xu, Daniel Marcus, Rivka R. Colen, and Spyridon Bakas. Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data. *Scientific Reports*, 10(1):12598, 2020. doi: 10.1038/s41598-020-69250-1. URL https://doi.org/10.1038/s41598-020-69250-1. Cited on page 5.

[80] Jinhyun So, Corey J. Nolet, Chien-Sheng Yang, Songze Li, Qian Yu, Ramy E. Ali, Basak Guler, and Salman Avestimehr. Lightsecagg: a lightweight and versatile design for secure aggregation in federated learning. In Diana Marculescu, Yuejie Chi, and Carole-Jean Wu, editors, *Proceedings of Machine Learning and Systems 2022, MLSys 2022, Santa Clara, CA, USA, August 29 - September 1, 2022*. mlsys.org, 2022. URL https://proceedings.mlsys.org/paper/2022/

hash/d2ddea18f00665ce8623e36bd4e3c7c5-Abstract.html. Cited on page 10.

[81] Konstantin Sozinov, Vladimir Vlassov, and Sarunas Girdzijauskas. Human activity recognition using federated learning. In Jinjun Chen and Laurence T. Yang, editors, *IEEE International Conference on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications, ISPA/IUCC/BDCloud/SocialCom/SustainCom 2018, Melbourne, Australia, December 11-13, 2018*, pages 1103–1111. IEEE, 2018. doi: 10.1109/BDCloud.2018.00164. URL https://doi.org/10.1109/BDCloud.2018.00164. Cited on pages 5 and 28.

[82] Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. Towards personalized federated learning. *CoRR*, abs/2103.00710, 2021. URL https://arxiv.org/abs/2103.00710. Cited on page 14.

[83] Yue Tan, Guodong Long, Lu Liu, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. Fedproto: Federated prototype learning across heterogeneous clients. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 8432–8440. AAAI Press, 2022. URL https://ojs.aaai.org/index.php/AAAI/article/view/20819. Cited on page 15.

[84] Hanlin Tang, Shaoduo Gan, Ce Zhang, Tong Zhang, and Ji Liu. Communication compression for decentralized training. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 7663–7673, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/44feb0096faa8326192570788b38c1d1-Abstract.html. Cited on page 17.

[85] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971, 2023. doi: 10.48550/arXiv.2302.13971. URL https://doi.org/10.48550/arXiv.2302.13971. Cited on page 5.

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

doi: 10.48550/arXiv.2302.12862. URL https://doi.org/10.48550/arXiv.2302.12862. Cited on page 5.

[87] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris S. Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=BkluqlSFDS. Cited on page 15.

[88] Zhiyuan Wang, Hongli Xu, Jianchun Liu, He Huang, Chunming Qiao, and Yangming Zhao. Resource-efficient federated learning with hierarchical aggregation in edge computing. In *40th IEEE Conference on Computer Communications, INFOCOM 2021, Vancouver, BC, Canada, May 10-13, 2021*, pages 1–10. IEEE, 2021. doi: 10.1109/INFOCOM42981.2021.9488756. URL https://doi.org/10.1109/INFOCOM42981.2021.9488756. Cited on page 17.

[89] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H. Yang, Farhad Farokhi, Shi Jin, Tony Q. S. Quek, and H. Vincent Poor. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Trans. Inf. Forensics Secur.*, 15:3454–3469, 2020. doi: 10.1109/TIFS.2020.2988575. URL https://doi.org/10.1109/TIFS.2020.2988575. Cited on pages 10 and 23.

[90] White House. Consumer data privacy in a networked world: A framework for protecting privacy and promoting innovation in the global digital economy. *Journal of Privacy and Confidentiality*, 4(2), Mar. 2013. doi: 10.29012/jpc.v4i2.623. URL https://journalprivacyconfidentiality.org/index.php/jpc/article/view/623. Cited on page 5.

[91] Qiong Wu, Xu Chen, Zhi Zhou, and Junshan Zhang. Fedhome: Cloud-edge based personalized federated learning for in-home health monitoring. *IEEE Trans. Mob. Comput.*, 21(8):2818–2832, 2022. doi: 10.1109/TMC.2020.3045266. URL https://doi.org/10.1109/TMC.2020.3045266. Cited on page 10.

[92] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. A comprehensive survey on graph neural networks. *IEEE Trans. Neural Networks Learn. Syst.*, 32(1):4–24, 2021. doi: 10.1109/TNNLS.2020.2978386. URL https://doi.org/10.1109/TNNLS.2020.2978386. Cited on page 25.

[93] Cong Xie, Sanmi Koyejo, and Indranil Gupta. Asynchronous federated optimization. *CoRR*, abs/1903.03934, 2019. URL http://arxiv.org/abs/1903.03934. Cited on pages 10, 11, 13, and 17.

[94] Chenhao Xu, Youyang Qu, Yong Xiang, and Longxiang Gao. Asynchronous federated learning on heterogeneous devices: A survey. *CoRR*, abs/2109.04269, 2021. URL https://arxiv.org/abs/2109.04269. Cited on pages 6, 13, and 20.

[95] Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mt5: A massively multilingual pre-trained text-to-text transformer. *CoRR*, abs/2010.11934, 2020. URL https://arxiv.org/abs/2010.11934. Cited on page 29.

[96] Han Yu, Zelei Liu, Yang Liu, Tianjian Chen, Mingshu Cong, Xi Weng, Dusit Niyato, and Qiang Yang. A sustainable incentive scheme for federated learning. *IEEE Intell. Syst.*, 35(4):58–69, 2020. doi: 10.1109/MIS.2020.2987774. URL https://doi.org/10.1109/MIS.2020.2987774. Cited on page 5.

[97] Tao Yu, Eugene Bagdasaryan, and Vitaly Shmatikov. Salvaging federated learning by local adaptation. *CoRR*, abs/2002.04758, 2020. URL https://arxiv.org/abs/2002.04758. Cited on pages 14 and 17.

[98] Michael Zhang, Karan Sapra, Sanja Fidler, Serena Yeung, and Jose M. Alvarez. Personalized federated learning with first order model optimization. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=ehJqJQk9cw. Cited on page 7.

[99] Ying Zhang, Tao Xiang, Timothy M. Hospedales, and Huchuan Lu. Deep mutual learning. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4320–4328. Computer Vision Foundation / IEEE Computer Society, 2018. doi: 10.1109/CVPR.2018.00454. URL http://openaccess.thecvf.com/content_cvpr_2018/html/Zhang_Deep_Mutual_Learning_CVPR_2018_paper.html. Cited on pages 7, 14, 21, 27, and 28.

[100] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *CoRR*, abs/1806.00582, 2018. URL http://arxiv.org/abs/1806.00582. Cited on pages 9, 10, 14, and 20.

[101] Tailin Zhou, Zehong Lin, Jun Zhang, and Danny H. K. Tsang. Understanding model averaging in federated learning on heterogeneous data. *CoRR*, abs/2305.07845, 2023. doi: 10.48550/arXiv.2305.07845. URL https://doi.org/10.48550/arXiv.2305.07845. Cited on pages 7, 11, 13, 21, and 25.

[102] Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 14747–14756, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/60a6c4002cc7b29142def8871531281a-Abstract.html. Cited on page 10.

# Appendix A

# Completed Works

# Can Fair Federated Learning reduce the need for Personalisation?

Alex Iacob, Pedro P. B. Gusmão, Nicholas D. Lane
University of Cambridge

## Abstract

Federated Learning (FL) enables training ML models on edge clients without sharing data. However, the federated model's performance on local data varies, disincentivising the participation of clients who benefit little from FL. Fair FL reduces accuracy disparity by focusing on clients with higher losses while personalisation locally fine-tunes the model. Personalisation provides a participation incentive when an FL model underperforms *relative* to one trained locally. For situations where the federated model provides a lower accuracy than a model trained entirely locally by a client, personalisation improves the accuracy of the pre-trained federated weights to be similar to or exceed those of the local client model. This paper evaluates two Fair FL (FFL) algorithms as starting points for personalisation. Our results show that FFL provides no benefit to relative performance in a language task and may double the number of underperforming clients for an image task. Instead, we propose Personalisation-aware Federated Learning (PaFL) as a paradigm that pre-emptively uses personalisation losses during training. Our technique shows a 50% reduction in the number of underperforming clients for the language task while lowering the number of underperforming clients in the image task instead of doubling it. Thus, evidence indicates that it may allow a broader set of devices to benefit from FL and represents a promising avenue for future experimentation and theoretical analysis.

***CCS Concepts:*** • **Computing methodologies** → **Machine learning**; **Neural networks**; Regularization.

***Keywords:*** federated learning, fairness, personalisation, privacy, machine learning, fine-tuning, local adaptation

**ACM Reference Format:**
Alex Iacob, Pedro P. B. Gusmão, Nicholas D. Lane. 2023. Can Fair Federated Learning reduce the need for Personalisation?. In *3rd Workshop on Machine Learning and Systems (EuroMLSys '23), May 8, 2023, Rome, Italy*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3578356.3592592



*EuroMLSys '23, May 8, 2023, Rome, Italy*

ACM ISBN 979-8-4007-0084-2/23/05.
https://doi.org/10.1145/3578356.3592592

## 1 Introduction

Edge devices provide computational power and data for Machine Learning tasks; however, minimising communication costs while using them can be challenging. Federated Learning (FL) was introduced by McMahan et al. [18] to enable model training on client devices without sharing data. However, the FL model accuracy may be underwhelming on clients with unusual data and even worse than a local model, reducing the incentive for participation.

The existing body of research on balancing global and local performance has proposed several approaches. Two Fair Federated Learning (FFL) techniques, q-Fair Federated Learning (q-FFL) and Tilted Empirical Risk Minimization (TERM), proposed by Li et al. [15] and Li et al. [11] respectively, aim to improve the accuracy of the worst-performing clients by prioritising those with large losses during FL. Alternatively, Yu et al. [23] and Mansour et al. [17] recommend using personalisation (local adaptation) methods such as Freezebase (FB), Elastic Weight Consolidation (EWC), and Knowledge Distillation (KD) for fine-tuning. In this work, *relative accuracy* refers to the difference in local client test set accuracy between a federated and local model.

While the sets of potential use cases for fairness and personalisation are not identical—e.g., personalisation would be inappropriate for clients with few samples—FFL could construct a fairer relative accuracy distribution as a starting point. For FFL to reduce the need for personalisation, it would have to lower the number of underperforming clients or improve their average relative accuracy. However, in our experiments, FFL had a neutral or negative effect on the relative accuracy distribution. *Our contribution is threefold:*

1. We construct an initial empirical evaluation of the relative accuracy distribution of models trained with FFL on the Reddit, CIFAR-10, and FEMNIST datasets for next-word prediction and image recognition tasks. During our evaluation, we show that FFL does not significantly reduce the number of underperforming clients or improve the relative accuracy distribution on Reddit and brings little benefit over FedAvg and personalisation. We also show it doubles the number of underperforming clients for FEMNIST.
2. We investigate potential synergies between FFL and personalisation by adapting fair federated models. Results show that the adapted models do not significantly outperform those initially trained with FedAvg

in relative accuracy or the number of underperforming clients.

3. We propose Personalization-aware Federated Learning (PaFL) as a paradigm that uses local adaptation techniques during FL training to pre-empt personalisation. Results on the language task show a significant reduction in underperforming clients over FFL when applying KD without any downsides to subsequent personalisation. Moreover, PaFL can avoid the increase in underperforming clients observed for image recognition on FEMNIST when using EWC or KD for our tested hyperparameters. However, given that our results are based entirely on simulation, future work consisting of additional experimentation and theoretical analysis is needed to determine the exact relationship between the training loss, e.g., KD or EWC, fairness, and local accuracy after personalisation.

## 2 Background and Related Work

Li et al. [13] formulate the FL objective function as seen in Eq. (1)

$$\min_w f(w) = \sum_{k=1}^{m} p_k F_k(w) \,, \tag{1}$$

where $f$ is the federated loss, $m$ is the client count, $w$ is the model, and $F_k$ is the loss of client $k$ weighted by $p_k$. For a total number of samples $n$, $p_k$ is defined as the proportion of samples on the client $\frac{n_k}{n}$. The Federated Averaging (FedAvg) algorithm introduced by McMahan et al. [18] trains locally on clients and for each round $t$ sums the parameters of each model $G_k^t$ from client $k$ weighted by $p_k$ with the previous model $G^t$ using learning rate $\eta$, as seen in Eq. (2)

$$G^{t+1} = G^t + \eta \left( \sum_{k=1}^{m} p_k G_k^t \right) \,. \tag{2}$$

***Data and Hardware heterogeneity.*** Data generation, network speed and computation naturally vary across devices due to hardware, location, time, and behaviour. These factors make the data distribution not Idendepentend and Identically Distributed (IID), leading to feature label or quantity skew as reported by Kairouz et al. [8, sec.3.1]. Non-IID data can impact accuracy [6, 24] and convergence [16] while different hardware results in stragglers and unreliability.

### 2.1 Fair Federated Learning

Li et al. [15] propose Fair FL (FFL), which defines a "fairer" FL model as one that achieves a lower variance in its accuracy distribution over local client test sets while keeping average accuracy similar. They propose a version of FFL, q-FFL, to emphasise underperforming clients during federated training as seen in Eq. (3)

$$\min_w f(w) = \sum_{k=1}^{m} \frac{p_k}{q+1} F_k^{q+1}(w) \,, \tag{3}$$

where $q$ controls the degree of fairness. A $q = 0$ corresponds to FedAvg, while larger values prioritise higher losses to improve accuracy on clients for which the federated model underperforms. Li et al. [11] develop Tilted Empirical Risk Minimization (TERM), shown in Eq. (4), which behaves similarly to q-FFL. While the two objectives show comparable improvements in the evaluations of Li et al. [11], their interactions with personalisation are unknown.

$$\min_w f(w) = \frac{1}{t} \log\left( \sum_{k=1}^{m} p_k e^{t F_k(w)} \right) \,, \tag{4}$$

The original publications of Li et al. [11, 15] used a weighted sampling of devices based on their number of examples, followed by uniform averaging. However, the server must know the dataset size of all clients a priori, which is potentially unfeasible. Thus, we use uniform sampling and weighted averaging in this work.

The most relevant recent FFL work is Ditto, published by Li et al. [12], which constructs personalised models while encouraging fairness for the global model. Ditto keeps a persistent local model in sync with the federated one by minimising the $L_2$ distance to the federated model, similar to the personalisation techniques discussed below. While Li et al. [12] show this local regularisation to be superior to TERM in promoting fairness, it requires more resources than personalisation. Maintaining a persistent local model incurs training costs on every single round, in addition to the increased storage demands during training, without the benefit of a highly-trained federated model to provide pre-trained weights. As such, we have chosen to opt against maintaining persistent local models; however, we recommend them as a promising avenue for future work.

### 2.2 Local Adaptation

The analysis of Yu et al. [23] established that the federated model performs worse on heterogeneous clients, as previously noted by Kairouz et al. [8], Li et al. [15], and it may offer inferior performance to local ones. They propose several techniques to address this.

***Elastic-weight Consolidation (EWC).*** The task of the global model is to maintain performance on all previous clients while training locally. Equation (5) frames it as Multi-task Learning (MTL) problem using the Elastic Weight Consolidation technique (EWC) introduced by Kirkpatrick et al. [9] to avoid Catastrophic forgetting [3]

$$l(C, x) = L(C, x) + \sum_i \tfrac{\lambda}{2} M[i] (C[i] - G[i])^2 \,, \tag{5}$$

where L is the client loss, $\lambda$ determines the weighting between the two tasks and $M$ is the Fisher information matrix.

***Fine-tuning (FT) and Freezebase (FB).*** When a client receives a global model after the FL process, it can apply

Fine-tuning [17, 19, 20] to retrain the model on its data. Furthermore, to avoid potential Catastrophic forgetting, Yu et al. [23] also opt to apply Freezebase (FB) as a variant of FT which retrains only the top layer.

***Knowledge Distillation (KD).*** As an alternative to EWC and FT, Knowledge Distillation [5] uses the global model as a teacher for a client model. For the pure logit outputs of the federated model $G(x)$ and client model $C(x)$, the client minimises the loss in Eq. (6)

$$l(C,x) = \alpha T^2 L(C,x) + (1-\alpha) K_L(\sigma(G(x)\,/\,T), \sigma(C(x)\,/\,T))\,, \tag{6}$$

where $L$ is the client loss, $K_L$ is the Kullback-Leibler divergence [10], $\sigma$ is the softmax, $\alpha$ is the weighting and $T$ is the temperature.

***FedProx.*** One relevant loss function not considered by Yu et al. [23] is the constraint on the Euclidean distance of model parameters employed by FedProx [14] to limit model divergence. The loss function shown in Eq. (7) is formulated similarly to EWC in Eq. (5).

$$l(C,x) = L(C,x) + \sum_i \tfrac{\lambda}{2}(C[i] - G[i])^2\,, \tag{7}$$

Rather than using it to create personalised models, the original work of Li et al. [14] employs this loss function between the model parameters at the start of the round and the parameters being trained on the client, which serves as the inspiration for our proposal.

## 3 Personalisation-aware Federated Learning

As an alternative to FFL for reducing personalisation costs, we consider modifying local client training in a manner which pre-empts a later adaptation phase by using loss functions meant for personalisation. The procedure is roughly analogous to quantisation-aware training [2]. This work uses *Personalisation-aware Federated Learning (PaFL)* to refer to such a paradigm. While Federated Learning and local adaptation have historically been regarded as separate, the FedProx algorithm developed by Li et al. [14] may be considered prototypical to PaFL as it injects the $L_2$ norm of the model weight differences into the local loss.

The FedProx algorithm aims to mitigate model divergence caused by data heterogeneity in a manner that may improve the accuracy of the federated model on average across clients. However, it only considers the contribution of the model parameters based on their magnitude rather than their importance to the output of the federated model. While this choice is well-justified by Li et al. [14], we consider some modifications while maintaining the principle when adapting it to pre-empt personalisation.

Personalisation-aware Federated Learning modifies FedProx to allow the loss function and the afferent weight to vary across rounds. Beyond potentially improved convergence, such a process may benefit final locally-trained models by providing continuity in the local objective between FL training and the final adaptation stage if the same loss function is used. Furthermore, loss-based weighted averaging as used in q-FFL (Eq. (3)) and TERM (Eq. (4)) has no means of reconciling differences between models required by clients with equally high losses and highly divergent data partitions. By contrast, PaFL allows clients for whom the global model performance is underwhelming to diverge in a manner which maintains accuracy on the whole federated distribution on which the model was trained.

Formally, PaFL can be defined as a type of Federated Learning where each client has a loss function obeying the structure in Eq. (8):

$$l(C,x,t) = \mu(t)\, L(C,x) + (1-\mu(t))\, D(t)(C,G,x)\,, \tag{8}$$

where $t$ is the current round, $L(C,x)$ is the training loss and $D(t)$ returns a personalisation loss function for the current round—potentially dependent on the data point $x$. The weight of each term is set per round through the weighting function $\mu(t)$. PaFL can be naturally extended to incorporate local adaptation if deemed beneficial by allowing clients to keep their model received after the final training round. However, for the rest of this work, PaFL shall refer only to the federated training phase. Local adaptation happens after federated training is complete to allow comparison against combinations of FL algorithms and personalisation methods.

## 4 Experimental Design

Following the lead of Yu et al. [23] and McMahan et al. [18], we train models using FedAvg, q-FedAvg, TERM, or PaFL for next-word prediction on a version of the Reddit [1] dataset with 80 000 participants each having $150-500$ posts treated as sperate sentences. We also train on CIFAR-10 partitioned into 100 participants and the naturally heterogeneous Federated Extended MNIST (FEMNIST) [1] for image recognition. During local adaptation, we follow the parameters recommended by Yu et al. [23]. For EWC, we use a weighting of $\lambda = 5000$; for KD, we use a temperature $T = 6$ and weighting $\alpha = 0.95$. Our hyperparameter choices attempt to replicate those of Yu et al. [23] whenever possible.

***Reddit.*** Reddit contains diverse sentences from its forum users, which makes it a valuable resource for Federated Learning, as users' total word counts and vocabulary size vary across several orders of magnitude with a skewed distribution. We train a standard LSTM for next-word prediction using 2 layers, 200 hidden units and 10 million parameters. To construct tokens, we employ the dictionary of the $50,000$ most frequent words compiled by Yu et al. [23]; all other words are replaced with placeholders. The first 90% of a user's posts, chronologically, is used as a training set, with the final 10% reserved for local testing. A separate centralised

test set is maintained for evaluating global task performance during the FL training process with ≈ 5% of it used to track convergence. In contrast, the full test set is used for the final evaluation. Federated models train for 1 000 rounds using 20 clients per round. On the client side, models train for 2 internal epochs with a batch size of 20 using SGD with a learning rate of 40. For adaptation, we use a learning rate of 1 and batch size of 20 for 100 epochs of retraining.

***FEMNIST.*** Federated Extended MNIST is an image dataset comprised of 62 characters written in a $28x28$ format. It is naturally divided into clients based on the author of a character, with each client having 226 samples on average. We use a similar experimental setup to Caldas et al. [1] with a simple two-layer CNN. Rather than subsampling 5% of the data from all clients as Caldas et al. [1] do, we keep 350 clients with more than 10 samples out of the total 3 597. We use 70% of a client's data for training, 10% for local testing and add the remaining 20% to the federated test set. For the FL process, we use an aggregation learning rate of $\eta = 1.0$ with 10 clients per round for 500 rounds. During training, we use SGD for 2 internal epochs with a learning rate of 0.1 and a batch size of 32 for each client, while during adaptation, we lower the learning rate to 0.01.

***CIFAR-10.*** CIFAR-10 is an image dataset composed of $60,000$ images of 10 objects in a $32x32$ format. Since CIFAR-10 is not a naturally federated dataset as it is not split into clients, a Dirichlet distribution ($\alpha = 0.9$) is used to simulate a Non-IID partitioning similarly to Hsu et al. [7] and Yu et al. [23]. A ResNet-18 [4] model is trained over $1,000$ rounds with 10 clients per round. Clients are trained using a batch size of 32 with 2 internal epochs and a learning rate of 0.1. The test accuracy is computed by multiplying a client's per-class accuracy on the CIFAR-10 test set with its proportion of the local device data. For adaptation, we use a learning rate of $10^{-3}$ and batch size of 32 for 200 epochs. Training uses SGD with momentum 0.9 and weight decay $5 \times 10^{-4}$,

### 4.1 Experiments

We train models for FedAvg, q-FedAvg, TERM and PaFL. Specifically, FedAvg (i.e., q-FedAvg with $q = 0$) is trained using the abovementioned standard parameters and serves as the baseline for all datasets and models. For q-FedAvg, we test $q \in \{0, 0.01, 0.1, 0.5, 1, 5\}$ for Reddit and show evaluation results for the relevant values of $q \in \{0, 0.1, 5\}$ which produce sufficiently distinguished results. Similarly, we test $q \in \{0, 0.1, 1, 5, 10, 15\}$ for FEMNIST and CIFAR-10, and we report the evaluation results for $q \in \{0, 10, 15\}$ and $q \in \{0, 5, 15\}$ respectively as to showcase the overall trend that increases in fairness create. For TERM on Reddit we use $t \in \{0.1, 5\}$ while on FEMNIST we do not tune the value of $t$ for TERM and instead reuse the $t = 1$ value chosen by Li et al. [12]. For PaFL we choose a simple proof-of-concept training sequence where we apply KD or EWC with constant weightings and parameters after the model has approached convergence at the halfway point of training—denoted $H_{EWC}$ and $H_{KD}$. We use the same parameters and weightings for the losses after the halfway point as in local adaptation.

***Centralised evaluation.*** The first experiment uses the held-out centralised test set of each dataset defined above to test federated models. It is also used to choose which models should be tested locally or adapted given our hardware constraints from Section 4.2.

***Local Accuracy Evaluation.*** The second experiment evaluates federated models on each client's local test set and reports average accuracy and variance for the client population. We also investigate accuracy and variance for the best and worst 10% of clients in terms of test accuracy. Finally, we report the average accuracy of locally trained models using the same parameters as in the local adaptation phase without the personalisation loss.

***Local Adaptation and Relative Accuracy Evaluation.*** The primary experimental setup entails comparing the accuracy of federated or adapted models with purely local models on client data; the difference between the two is referred to as relative accuracy. The effectiveness of federated models is determined by two key factors: the number of clients with positive relative accuracy and the average relative accuracy. If a synergistic relationship exists between FFL or PaFL and local adaptation methods, models trained using these techniques would significantly improve average relative accuracy or have fewer underperforming clients after adaptation.

### 4.2 Hardware Limitations

Each node of the cluster that the experiments were run on holds four Nvidia A100 GPUs. Given the quotas and service levels of the cluster, the number of clients on which the federated model could be tested locally for the language task was limited to ∼ 65 500 to avoid incurring costs beyond the allocated university funds. Similarly, the number that could be adapted was limited to (∼ 18500). Therefore, all charts and tables comparing local model or adaptation performance use data from the client set common to all results.

## 5 Results

We begin by examining the convergence process for next-word prediction on Reddit and summarise our findings on the centralised test-set accuracy and local accuracy. As shown in Fig. 1 and Table 1, the impact of q-FFL on accuracy is neutral to negative for our tested $q$-values, while that of TERM is highly negative for all tested $t$. Although fairness reliably reduces the accuracy variance for $q \geq 1$, the performance cost is too high for all values reported in Section 4.1. Meanwhile, we found that TERM did not obtain acceptable performance for any (t) we explored and excluded it from future Reddit experiments.

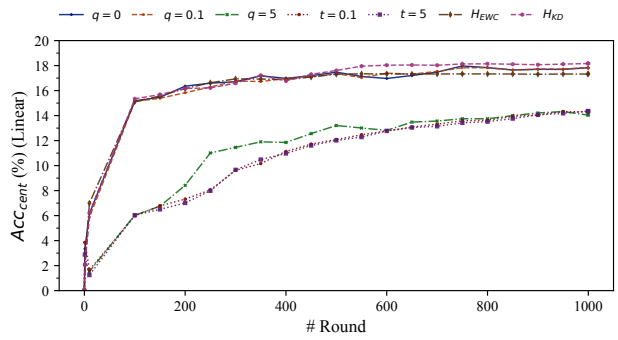



**Figure 1.** Language task test-set accuracy, TERM harms performance for our tested values while q-FFL only does so significantly for $q \geq 1.0$. Crucially, both $H_{EWC}$ and $H_{KD}$ approach the FedAvg baseline with $H_{KD}$ exceeding it.

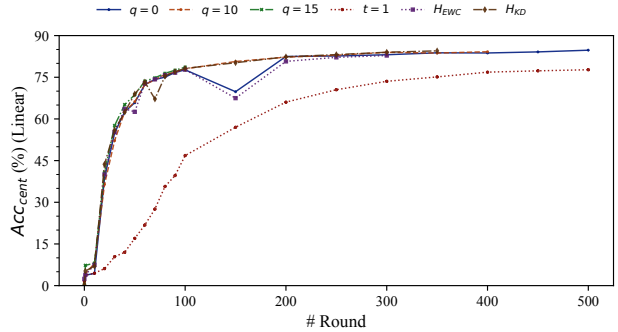



**Figure 2.** FEMNIST centralised accuracy, q-FFL and PaFl approach FedAvg for our tested hyperparameters while causing increased instability in the training process or outright divergence past a certain round—for such models, we test and adapt the last version before divergence. TERM merely underperforms without divergence for the $t = 1$ value.

Moving on to the FEMNIST training results, Fig. 2 and Table 2 show that q-FedAvg and PaFL both tend to cause instability for our hyperparameters. However, both q-FedAvg with $q = 10$ and $H_{KD}$ have a comparable centralised and average local accuracy to FedAvg. Unlike Reddit, TERM is well-behaved at a value of $t = 1$ and thus included in future experiments.

Table 3 indicates the CIFAR-10 image classification task to be more resilient to fairness than previous tasks, with a noticeable accuracy decrease only observable for $q \geq 10$. The lower sensitivity of this task to FFL is consistent with Yu et al. [23] who find CIFAR-10 highly resilient to robust [22] FL and differential privacy [21]. Due to the similarity across fairness levels, the convergence graph for CIFAR-10 is not shown. Given this lesser sensitivity for our tested values, we chose not to expand the CIFAR-10 experiments past q-FedAvg and $H_{KD}$. These findings indicate that the dataset heterogeneity

| Objective | $Acc_{cent}(\%)$ | $Avg_{loc}(\%)$ | $B_{loc}(\%)$ | $W_{loc}(\%)$ | $(Var_{Avg})$ | $(Var_B)$ | $(Var_W)$ |
|---|---|---|---|---|---|---|---|
| $q = 0$ | 17.826 | 18.645 | 24.572 | 14.815 | 9.177 | **22.114** | 1.072 |
| $q = 0.1$ | 17.789 | 18.66 | 24.843 | 14.728 | 9.81 | 22.914 | 1.036 |
| $q = 5$ | 14.056 | 14.819 | 20.208 | 11.66 | **7.983** | 26.769 | 0.69 |
| $t = 0.1$ | 14.299 | 16.476 | **28.985** | 11.806 | 39.584 | 176.42 | **0.369** |
| $t = 5$ | 14.373 | 16.438 | 28.981 | 11.766 | 39.642 | 175.96 | 0.382 |
| $H_{EWC}$ | 17.322 | 18.255 | 24.653 | 14.226 | 10.277 | 23.059 | 1.185 |
| $H_{KD}$ | **18.177** | **19.179** | 26.406 | **14.887** | 12.438 | 25.85 | 1.039 |
| Local | NaN | 4.456 | 10.227 | 1.204 | 8.777 | 31.11 | 0.893 |

**Table 1.** Results showing the centralised and local accuracy on Reddit. The $Acc_{cent}(\%)$ value refers to the accuracy of the federated model on the centralised test set. In contrast, $Avg_{loc}(\%)$, $B_{loc}(\%)$, and $W_{loc}(\%)$ refer to the average accuracy of the model on the local test sets of the whole population, the top 10% of clients in terms of local accuracy and the worst 10% respectively. The $(Var_{Avg})$, $(Var_B)$, and $(Var_W)$ values refer to the variance in accuracy seen by the populations above. While fairness does decrease variance at $q \geq 1.0$, the harm to accuracy is too great compared to $q = 0.1$. The proposed $H_{KD}$ model improves accuracy across clients but increases variance for everyone except the worst performers.

| Objective | $Acc_{cent}(\%)$ | $Avg_{loc}(\%)$ | $B_{loc}(\%)$ | $W_{loc}(\%)$ | $(Var_{Avg})$ | $(Var_B)$ | $(Var_W)$ |
|---|---|---|---|---|---|---|---|
| $q = 0$ | **84.739** | 75.341 | 99.435 | 35.717 | 432.507 | 1.151 | 73.747 |
| $q = 10$ | 84.19 | **76.591** | 99.013 | **42.055** | **320.385** | 1.714 | 64.049 |
| $q = 15$ | 78.634 | 69.749 | 96.681 | 34.988 | 374.637 | 5.177 | **47.761** |
| $t = 1$ | 77.706 | 69.134 | 98.628 | 33.478 | 417.448 | 2.771 | 56.543 |
| $H_{EWC}$ | 82.825 | 73.964 | 99.321 | 33.457 | 465.605 | 1.376 | 71.481 |
| $H_{KD}$ | 84.51 | 75.243 | **99.491** | 34.34 | 443.015 | **1.07** | 62.91 |
| Local | NaN | 46.322 | 92.848 | 0.0 | 1006.77 | 18.144 | 0.0 |

**Table 2.** Results for FEMNIST. Unlike Reddit, there did not seem to be a clear proportional relation between accuracy and fairness level for our tested parameters; as such, we chose to report the best and "fairest" value. Furthermore, using KD helps the best performers primarily; however, both $H_{KD}$ and $H_{EWC}$ do well.

may need to be meaningful rather than artificially imposed for significant effects to emerge.

*Implications:* The loss-based averaging mechanism of q-FedAvg and TERM is not guaranteed to improve the final accuracy distribution proportionally to the fairness parameter and may fail to do so under our specific experimental

| Objective | $Acc_{cent}(\%)$ | $Avg_{loc}(\%)$ | $B_{loc}(\%)$ | $W_{loc}(\%)$ | $(Var_{Avg})$ | $(Var_B)$ | $(Var_W)$ |
|---|---|---|---|---|---|---|---|
| $q = 0$ | 81.28 | **81.37** | **82.255** | **79.864** | 0.568 | 0.022 | 1.067 |
| $q = 5$ | **81.86** | 81.221 | 82.011 | 79.794 | **0.446** | **0.004** | **0.643** |
| $q = 15$ | 78.16 | 79.935 | 81.178 | 77.885 | 0.945 | 0.02 | 1.267 |
| Local | NaN | 31.718 | 38.297 | 24.649 | 16.3 | 1.543 | 0.906 |

**Table 3.** Results for CIFAR-10. Unlike the language task, $q = 5$ represents an optimum across all our tested values in terms of variance while maintaining performance; however, differences are small.

conditions. This calls for further inquiry into the viability of such methods.

### 5.1 FFL fails to improve relative accuracy

Having established baselines of accuracy for fair models, we can now evaluate the relative accuracy of FFL, PaFL and their interactions with local adaptation. Unfortunately, the CIFAR-10 data is uninformative as the federated model outperforms the local one for all clients, consistent with the findings of Yu et al. [23].

For q-FFL, the results for the language task showcased in Table 4 are less than satisfactory as fair models fail to provide benefits in terms of the number of underperforming clients, relative accuracy, or variance. Furthermore, fair models do not offer an improvement over FedAvg once adapted—this is directly visible in the Fig. 3a scatter plot of relative accuracy against local model accuracy.

The results for image recognition on FEMNIST are more unusual yet similarly discouraging for both q-FFL and TERM. Table 5 makes it clear that the fair model achieves a higher relative accuracy on average and amongst the top 10% of clients at the cost of obtaining a *negative* relative accuracy on the worst 10%. Additionally, it has over twice as many underperforming clients with negative relative accuracies. We speculate that focusing on clients with high losses harms the accuracy of fair models on those capable of training high-quality local models. This result is corroborated by the final distribution shown in Fig. 3b, as *all* the underperforming clients have high local model accuracy. Another factor to consider is the atypical personalisation behaviour of FEMNIST. Models trained with FedAvg and then adapted tend to converge to nearly the same relative accuracy regardless of adaptation technique.

*Implications:* For our tested hyperparameters, Fair FL training algorithms may harm the relative accuracy distribution for datasets where clients can train high-quality local models by themselves. Therefore, we explore a training methodology intended not to sacrifice accuracy for such clients.

### 5.2 PaFL as an Alternative

Having shown the inability of FFL to replace or enhance local adaptation, we argue that it is not the right approach for this application. In principle, for an FL algorithm to provide benefits in terms of relative accuracy, it must achieve two goals. First, it must ensure that the worst-performing clients receive sufficient accuracy to match or exceed local models. Second, for the clients with the *best* local models, it must provide disproportionately high accuracy. While FFL may help fulfil the first requirement, its inability to raise the floor of the worst performers without hurting the ceiling of those that might have an excellent local model makes it incapable of fulfilling the second in our simulations.

Personalisation-aware Federated Learning, in the most general case, offers an alternative where models can be kept closer to one another during training and only allowed to diverge in ways which hurt federated performance the least. Unlike regularisation based on the norm of the distance between model parameters (e.g., FedProx), EWC and KD offer the distinct advantage of determining how a parameter may diverge based on its importance to federated performance. Thus, the model can learn from highly heterogeneous data and raise its accuracy floor for the worst performers without hurting the accuracy ceiling of the best or even improving it.

Preliminary results for the language task are promising in the case of $H_{KD}$ as Figure 1 and Table 1 indicate that it performs better than FedAvg and FFL models in every metric except variance and best-performer variance. Notably, variance is not increased for the worst performers. On the other hand, while $H_{EWC}$ is not far below the FedAvg baseline, it fails to provide any noticeable improvements. In terms of relative accuracy, Table 4 shows that $H_{KD}$ halves the number of underperforming clients and provides the best average relative accuracy. However, this higher baseline does not translate to improved relative accuracy for adapted models. Overall, lowering the number of clients which *require* adaptation in order to receive an incentive to participate $H_{KD}$ successfully reduces the need for personalisation on Reddit. On the other hand, $H_{EWC}$ seems to double the number of underperforming clients for the fixed chosen $\lambda$, although a different value may change results.

For image recognition on FEMNIST, $H_{KD}$ and $H_{EWC}$ are satisfactory in terms of centralised and average accuracy according to Fig. 2 and Table 2. On the other hand, relative accuracy results in Table 5 are mixed. While both avoid the doubling in underperforming clients that fair models suffer, locally adapted models starting from $H_{KD}$ as a baseline do not seem to outperform those adapted from FedAvg. Perhaps surprisingly, given its failure on the language task, $H_{EWC}$ reduced the number of underperforming clients for baseline *and* adapted models despite a lower starting average relative accuracy than q-FedAvg and $H_{KD}$. While more experiments are needed, it indicates potential synergy between PaFL and local adaptation.

*Implications:* Personalisation-aware Federated Learning may successfully improve the relative accuracy distribution across all clients. By constraining divergence from the federated model in a manner meaningful to performance, PaFL may learn from clients with "harder" datasets without harming the accuracy of those capable of training high-quality local models.

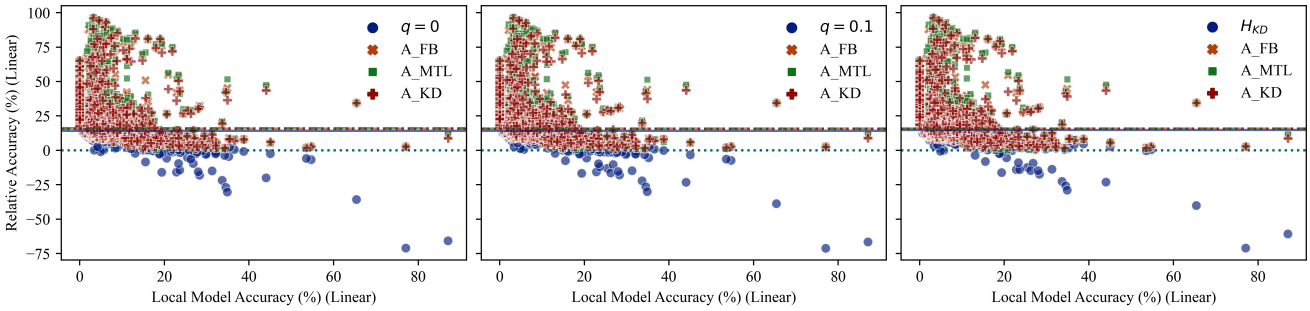
| Objective | Adapt | $Avg_{loc}(\%)$ | % <0 | $B_{loc}(\%)$ | $W_{loc}(\%)$ | $(Var_{Avg})$ | $(Var_B)$ | $(Var_W)$ |
|---|---|---|---|---|---|---|---|---|
| $q = 0$ | $q = 0$ | 14.185 | 53 | 20.715 | 9.392 | **13.323** | 34.379 | 20.201 |
| | A_FB | 15.87 | 0 | 25.849 | **11.311** | 29.216 | 149.736 | 3.246 |
| | A_EWC | **16.046** | 0 | 27.558 | 11.304 | 36.067 | 178.387 | 3.337 |
| | A_KD | 15.538 | 0 | 24.376 | 11.209 | 23.112 | 115.016 | 3.183 |
| $q = 0.1$ | $q = 0.1$ | 14.208 | 50 | 20.907 | 9.359 | 13.742 | 35.005 | 20.733 |
| | A_FB | 15.827 | 0 | 25.964 | 11.261 | 29.505 | 149.011 | 3.212 |
| | A_EWC | 15.839 | 0 | 27.692 | 11.024 | 37.108 | 179.066 | 3.336 |
| | A_KD | 15.546 | 0 | 24.614 | 11.19 | 23.95 | 118.471 | 3.166 |
| $H_{EWC}$ | $H_{EWC}$ | 13.807 | 108 | 20.723 | 8.681 | 14.994 | 38.119 | 24.938 |
| | A_FB | 15.423 | 0 | 25.709 | 10.88 | 29.795 | 149.308 | 3.02 |
| | A_EWC | 15.561 | 0 | 27.482 | 10.762 | 37.251 | 179.528 | 3.266 |
| | A_KD | 15.157 | 0 | 24.336 | 10.823 | 23.996 | 117.427 | 3.041 |
| $H_{KD}$ | $H_{KD}$ | 14.729 | 27 | 22.038 | 9.971 | 14.969 | 41.225 | 18.409 |
| | A_FB | 15.772 | 0 | 26.533 | 11.154 | 31.068 | 150.552 | **3.156** |
| | A_EWC | 15.824 | 2 | **28.217** | 10.966 | 38.439 | 178.543 | 3.469 |
| | A_KD | 15.698 | 0 | 25.358 | 11.214 | 25.367 | 119.418 | 3.241 |

**Table 4.** Results showing the relative accuracy of FFL and PaFL models on Reddit, % < 0 refers to the number of clients with negative relative accuracy. The best value in a column is bold, while the best in a group is underlined. The chosen optimal fair model does not significantly reduce the number of underperforming clients in our experiments. Alternatively, $H_{KD}$ lowers it to half. Local adaptation always provides similar results for the chosen hyperparameters.

| Objective | Adapt | $Avg_{loc}(\%)$ | % <0 | $B_{loc}(\%)$ | $W_{loc}(\%)$ | $(Var_{Avg})$ | $(Var_B)$ | $(Var_W)$ |
|---|---|---|---|---|---|---|---|---|
| $q = 0$ | $q = 0$ | 29.02 | 16 | 65.768 | 2.729 | 338.387 | 137.366 | 12.33 |
| | A_FB | 28.954 | 17 | 65.463 | 2.72 | 334.754 | 134.824 | 11.853 |
| | A_EWC | 28.994 | 16 | 65.672 | **2.802** | 336.25 | 137.507 | 11.325 |
| | A_KD | 28.986 | 16 | 65.684 | 2.788 | 337.14 | 137.796 | 11.594 |
| $q = 10$ | $q = 10$ | **30.269** | 39 | **79.687** | -14.844 | 673.351 | **111.144** | 206.995 |
| | A_FB | 28.613 | **12** | 64.818 | 2.699 | 320.729 | 123.679 | 15.552 |
| | A_EWC | 28.612 | 14 | 64.818 | 2.516 | 321.3 | 123.679 | 15.869 |
| | A_KD | 28.563 | 14 | 64.645 | 2.52 | 320.957 | 127.618 | 15.934 |
| $t = 1$ | $t = 1$ | 22.812 | 56 | 73.593 | -17.069 | 627.167 | 215.261 | 91.294 |
| | A_FB | 21.261 | 15 | 50.057 | 0.806 | **201.35** | 127.802 | 8.417 |
| | A_EWC | 21.362 | 15 | 50.218 | 0.765 | 201.703 | 123.192 | 8.102 |
| | A_KD | 21.202 | 15 | 50.1 | 0.706 | 202.488 | 125.153 | **7.71** |
| $H_{EWC}$ | $H_{EWC}$ | 27.642 | 13 | 62.157 | 1.522 | 315.388 | 123.438 | 19.225 |
| | A_FB | 27.558 | 14 | 62.431 | 1.404 | 316.622 | 124.092 | 18.888 |
| | A_EWC | 27.603 | 13 | 62.588 | 1.349 | 315.619 | 122.717 | 20.502 |
| | A_KD | 27.611 | 13 | 62.588 | 1.542 | 314.112 | 122.717 | 19.176 |
| $H_{KD}$ | $H_{KD}$ | 28.921 | 16 | 66.178 | 1.65 | 353.698 | 154.217 | 16.231 |
| | A_FB | 28.916 | 17 | 66.605 | 1.644 | 350.631 | 152.111 | 16.922 |
| | A_EWC | 28.871 | 16 | 66.605 | 1.719 | 350.742 | 152.111 | 16.865 |
| | A_KD | 28.967 | 15 | 66.329 | 1.869 | 351.991 | 150.583 | 17.124 |

**Table 5.** FEMNIST performance of the best fair model and our proposed alternative. Despite providing the highest average relative accuracy and the highest amongst the best 10%, the model trained using $q = 10$ has more than double the number of underperforming clients of FedAvg ($q = 0$). This is also true for TERM with $t = 1$. For PaFL, $H_{KD}$ is close to FedAvg while $H_{EWC}$ improves the number of underperforming clients for *both* baseline and adapted models.

**(a)** Reddit relative accuracy, fairness shows no benefit while $H_{KD}$ reduces the number of underperforming clients without adaptation.




**(b)** FEMNIST results, clients with highly accurate local models are underserved by federated models trained using $q = 10$ as many become underperformers. Alternatively, those trained using $H_{EWC}$ receive a slight improvement over FedAvg even when adapted.

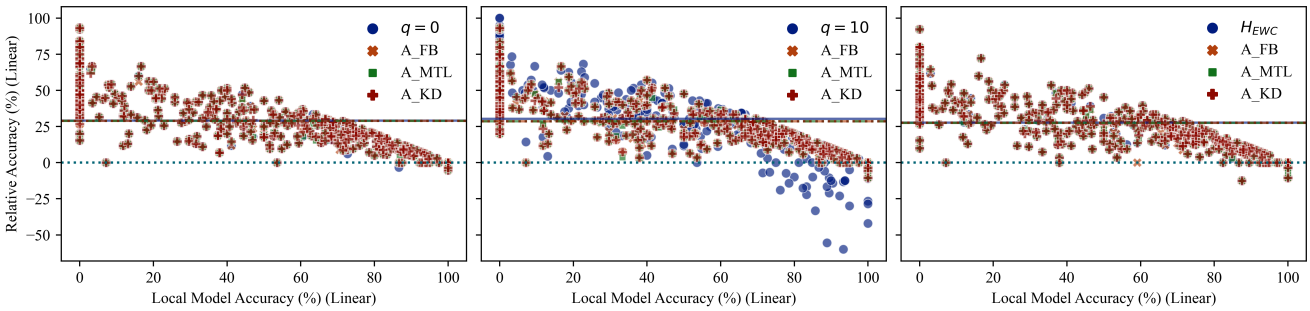



**Figure 3.** Federated or adapted model relative accuracy on a client plotted against local client model accuracy. The horizontal lines represent either the threshold for underperformance (0) or the average accuracy of a group of clients.

## 6 Conclusion

This paper set out to incentivise FL participation for clients whose local model outperforms a federated one while lowering the need for costly personalisation. Such a reduction would be relevant for federated networks containing devices with limited capabilities for retraining or little data. Our results indicate that FFL is unlikely to provide the desired properties as it did not reduce the number of underperforming clients on Reddit while doubling it on FEMNIST. We hypothesise that Fair FL harms clients on whom the federated model performs well but could train an excellent local model alone. Personalisation-aware Federated Learning offers an alternative approach, allowing loss functions used for local adaptation to be applied during FL and vary across rounds. After partial convergence, we applied EWC or KD to enable learning from worst-performing data without sacrificing performance on the federated distribution. While our chosen EWC configuration did not significantly improve over FedAvg on Reddit, KD showed promising results by reducing the number of underperforming clients by up to 50%. Furthermore, both avoided increasing the number of underperforming clients on FEMNIST while EWC slightly lowered it even for adapted models. Unlike more complex systems, which simultaneously train local and federated models, this approach does not require storing an additional model nor keeping it synchronised to the federated one. For KD, the computational overhead is smaller than a local model as it does not require training two separate networks, thus avoiding one of the two backward passes required by systems employing local models requir. For EWC, the advantage is even more apparent as computation scales only in the number of network parameters and requires no additional forward or backward passes to be performed. Consequently, we recommend using it to incentivise participation even when an explicit final local adaptation stage is not applied. In terms of future work, more extensive simulations and the addition of theoretical analyses would allow for greater insight into the relation between fairness and the loss function used during training and adaptation.

## References


[1] Sebastian Caldas, Peter Wu, Tian Li, Jakub Konečný, H. Brendan McMahan, Virginia Smith, and Ameet Talwalkar. 2018. LEAF: A Benchmark for Federated Settings. *CoRR* abs/1812.01097 (2018). arXiv:1812.01097 http://arxiv.org/abs/1812.01097

[2] Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W. Mahoney, and Kurt Keutzer. 2021. A Survey of Quantization Methods for Efficient Neural Network Inference. *CoRR* abs/2103.13630 (2021). arXiv:2103.13630 https://arxiv.org/abs/2103.13630

[3] Ian J. Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. 2013. An Empirical Investigation of Catastrophic Forgetting in Gradient-Based Neural Networks. https://doi.org/10.48550/ARXIV.1312.6211

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, 770–778. https://doi.org/10.1109/CVPR.2016.90

[5] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the Knowledge in a Neural Network. *CoRR* abs/1503.02531 (2015). arXiv:1503.02531 http://arxiv.org/abs/1503.02531

[6] Kevin Hsieh, Amar Phanishayee, Onur Mutlu, and Phillip Gibbons. 2020. The Non-IID Data Quagmire of Decentralized Machine Learning. In *Proceedings of the 37th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 119)*, Hal Daumé III and Aarti Singh (Eds.). PMLR, 4387–4398. https://proceedings.mlr.press/v119/hsieh20a.html

[7] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. 2019. Measuring the Effects of Non-Identical Data Distribution for Federated Visual Classification. *CoRR* abs/1909.06335 (2019). arXiv:1909.06335 http://arxiv.org/abs/1909.06335

[8] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista A. 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, Zaïd Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konečný, 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. *Found. Trends Mach. Learn.* 14, 1-2 (2021), 1–210. https://doi.org/10.1561/2200000083

[9] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences* 114, 13 (2017), 3521–3526.

[10] S. Kullback and R. A. Leibler. 1951. On Information and Sufficiency. *The Annals of Mathematical Statistics* 22, 1 (1951), 79 – 86. https://doi.org/10.1214/aoms/1177729694

[11] Tian Li, Ahmad Beirami, Maziar Sanjabi, and Virginia Smith. 2021. Tilted Empirical Risk Minimization. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. https://openreview.net/forum?id=K5YasWXZT3O

[12] Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. 2021. Ditto: Fair and Robust Federated Learning Through Personalization. In *Proceedings of the 38th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 139)*, Marina Meila and Tong Zhang (Eds.). PMLR, 6357–6368. https://proceedings.mlr.press/v139/li21h.html

[13] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. 2020. Federated Learning: Challenges, Methods, and Future Directions. *IEEE Signal Process. Mag.* 37, 3 (2020), 50–60. https://doi.org/10.1109/MSP.2020.2975749

[14] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated Optimization in Heterogeneous Networks. In *Proceedings of Machine Learning and Systems 2020, MLSys 2020, Austin, TX, USA, March 2-4, 2020*, Inderjit S. Dhillon, Dimitris S. Papailiopoulos, and Vivienne Sze (Eds.). mlsys.org. https://proceedings.mlsys.org/book/316.pdf

[15] Tian Li, Maziar Sanjabi, Ahmad Beirami, and Virginia Smith. 2020. Fair Resource Allocation in Federated Learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. https://openreview.net/forum?id=ByexElSYDr

[16] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. 2020. On the Convergence of FedAvg on Non-IID Data. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. https://openreview.net/forum?id=HJxNAnVtDS

[17] Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. 2020. Three Approaches for Personalization with Applications to Federated Learning. *CoRR* abs/2002.10619 (2020). arXiv:2002.10619 https://arxiv.org/abs/2002.10619

[18] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA (Proceedings of Machine Learning Research, Vol. 54)*, Aarti Singh and Xiaojin (Jerry) Zhu (Eds.). PMLR, 1273–1282. http://proceedings.mlr.press/v54/mcmahan17a.html

[19] Matthias Paulik, Matt Seigel, Henry Mason, Dominic Telaar, Joris Kluivers, Rogier C. van Dalen, Chi Wai Lau, Luke Carlson, Filip Granqvist, Chris Vandevelde, Sudeep Agarwal, Julien Freudiger, Andrew Byde, Abhishek Bhowmick, Gaurav Kapoor, Si Beaumont, Áine Cahill, Dominic Hughes, Omid Javidbakht, Fei Dong, Rehan Rishi, and Stanley Hung. 2021. Federated Evaluation and Tuning for On-Device Personalization: System Design & Applications. *CoRR* abs/2102.08503 (2021). arXiv:2102.08503 https://arxiv.org/abs/2102.08503

[20] Kangkang Wang, Rajiv Mathews, Chloé Kiddon, Hubert Eichner, Françoise Beaufays, and Daniel Ramage. 2019. Federated Evaluation of On-device Personalization. *CoRR* abs/1910.10252 (2019). arXiv:1910.10252 http://arxiv.org/abs/1910.10252

[21] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H. Yang, Farhad Farokhi, Shi Jin, Tony Q. S. 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. https://doi.org/10.1109/TIFS.2020.2988575

[22] Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. 2018. Byzantine-robust distributed learning: Towards optimal statistical rates. In *International Conference on Machine Learning*. PMLR, 5650–5659.

[23] Tao Yu, Eugene Bagdasaryan, and Vitaly Shmatikov. 2020. Salvaging Federated Learning by Local Adaptation. *CoRR* abs/2002.04758 (2020). arXiv:2002.04758 https://arxiv.org/abs/2002.04758

[24] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. 2018. Federated Learning with Non-IID Data. *CoRR* abs/1806.00582 (2018). arXiv:1806.00582 http://arxiv.org/abs/1806.00582

# PRIVACY IN MULTIMODAL FEDERATED HUMAN ACTIVITY RECOGNITION

**Alex Iacob** [1] **Pedro P. B. Gusmão** [1] **Nicholas D. Lane** [1] **Armand K. Koupai** [2] **Mohammud J. Bocus** [2]
**Raúl Santos-Rodríguez** [2] **Robert J. Piechocki** [2] **Ryan McConville** [2]

## ABSTRACT

Human Activity Recognition (HAR) training data is often privacy-sensitive or held by non-cooperative entities. Federated Learning (FL) addresses such concerns by training ML models on edge clients. This work studies the impact of privacy in federated HAR at a user, environment, and sensor level. We show that the performance of FL for HAR depends on the assumed privacy level of the FL system and primarily upon the colocation of data from different sensors. By avoiding data sharing and assuming privacy at the human or environment level, as prior works have done, the accuracy decreases by 5-7%. However, extending this to the modality level and strictly separating sensor data between multiple clients may decrease the accuracy by 19-42%. As this form of privacy is necessary for the ethical utilisation of passive sensing methods in HAR, we implement a system where clients mutually train both a general FL model and a group-level one per modality. Our evaluation shows that this method leads to only a 7-13% decrease in accuracy, making it possible to build HAR systems with diverse hardware.

## 1 INTRODUCTION

Human Activity Recognition (HAR) involves classifying human actions (Vrigkas et al., 2015; Jobanputra et al., 2019), such as running or sitting, using data from personal devices like smartphones or environmental sensors. However, practical and legal considerations limit learning from HAR data. For example, using video cameras to simulate virtual bodily-worn movement sensors (Kwon et al., 2020) may generate divergent features from Wi-Fi signals. Furthermore, privacy requirements impose data collection limitations. In this work, privacy requirements refer to constraints on collecting or centralising data at three levels:

**User(Human subject)-level Privacy** For gyroscope or accelerometer data from smartphones and wearables, end-users may be unwilling to share personal information.

**Environment-level Privacy** For locations such as hospitals and internment facilities, sensitive information must often remain private from third parties. This constraint may prove challenging as data used for HAR is susceptible to environmental characteristics. For example, a sensor may produce varying features based on object placement or room size.

**Modality-level Privacy** Data generated from different groups of sensors may be owned by competing entities.

[1]University of Cambridge [2]University of Bristol. Correspondence to: <aai30@cam.ac.uk>, <pp524@cam.ac.uk>, <ndl32@cam.ac.uk>, <uw20504@bristol.ac.uk>, <junaid.bocus@bristol.ac.uk>, <enrsr@bristol.ac.uk>, <eerjp@bristol.ac.uk>, <ryan.mcconville@bristol.ac.uk>.

Traditional Machine Learning approaches tackle feature heterogeneity by colocating data and training with Multi-task Learning techniques. However, the privacy constraints above make centralisation unfeasible on a large scale in HAR. Instead, they require a Federated Learning (FL) approach to keep data encapsulated in clients at the necessary privacy level during training. Our work brings the following contributions to Federated Human Activity Recognition:

1. First, we evaluate the performance of multiple models trained in a federated fashion on a multimodal dataset keeping data privately stored on clients at increasing privacy levels. Unlike other works, we investigate the additive effects of privacy up to the complete separation of each user, environment, and modality combination.
2. Second, we show that privacy at the modality level results in the *highest* accuracy cost, followed by the environmental level and then the user level. To mitigate this, we propose mutual learning of group-level models alongside the standard FL model to cover modalities that cannot be colocated in a single client. Our results indicate that this method can significantly reduce accuracy degradation from 19-42% to just 7-13%.

## 2 MULTIMODALITY IN FEDERATED HUMAN ACTIVITY RECOGNITION

Federated Learning, proposed by McMahan et al. (2017), trains ML models from distributed data on edge devices using efficient communication techniques for maintaining privacy. Although successful in training models from diverse users, such as keyboard prediction (Hard et al., 2018),

and diverse hardware, such as medical applications (Sheller et al., 2020), data heterogeneity remains a significant challenge (Kairouz et al., 2021, sec 3.1). Due to privacy constraints, approaches like Multi-task Learning and Continual Learning, which handle feature heterogeneity, are limited in a Federated Learning context. For instance, Elastic Weight Consolidation (Kirkpatrick et al., 2017) estimates parameter variance using past data, while Learning Without Forgetting (Li & Hoiem, 2018) stores network outputs from past tasks.

Previous work on Federated Human Activity Recognition has not fully explored the heterogeneity emerging from independent data collection systems. For instance, Sozinov et al. (2018) considers skewed label distributions and noise across smartphone users while *colocating* gyroscope and accelerometer data. Similar partitioning schemes are investigated for feature extraction (Xiao et al., 2021) and clustering methods (Ouyang et al., 2021). Furthermore, such works may use artificially partitioned centralised datasets, as in Zhao et al. (2020), or contain only one modality, as in some datasets collected by Ouyang et al. (2021). To create adaptable HAR systems that can accommodate new clients with different sensor types in the federation, investigating Federated HAR with modalities split across clients is necessary, given the shifting hardware landscape of HAR sensors.

## 3 Federating Human Activity Recognition

We construct multiple partitions of the OPERAnet dataset published by Bocus et al. (2022) to assess Federated Human Activity Recognition under privacy at user, environment, and modality privacy levels. The dataset contains five different sensors; however, Bocus et al. (2022) indicate that only Channel State Information (CSI) from a Network Card Interface (NIC) and Passive Wi-Fi Radar (PWR) should be used for HAR. The data were collected synchronously, with the multiple channels—three for CSI and two for PWR—of RF data. They cover eight hours of surveying six participants performing six activities spread across two rooms. Because room activity distribution is non-uniform, separating clients by environment also provides skewness at the label level.

We transform the time-series data into image data in keeping with the original HAR preprocessing applied by Bocus et al. (2022) and previous works (Bocus et al., 2021; Li et al., 2020; 2022). We further increase the dataset's size and modality diversity by reusing the pipeline of Koupai et al. (2022). Based on the underlying CSI and PWR data, Koupai et al. (2022) construct spectrograms of the CSI and PWR data. The complete image set contains five data views for each underlying CSI or PWR channel. We use the three most effective view types reported by Koupai et al. (2022). Since different channels for CSI and PWR are physically colocated on the device, it is assumed that fusing images generated from separate channels would not be a violation of privacy at the sensor level. Consequently, the complete image types we shall refer to as modalities for the rest of this work contain concatenated images generated from each source channel. One such image type comes from CSI; two come from PWR.

The work of Koupai et al. (2022) offers two centralised baselines to compare against, a ResNet34 (He et al., 2016) model used for HAR and a Fusion Vision Transformer (FViT). While CNNs have been successfully applied to HAR by Bevilacqua et al. (2018); Ronald et al. (2021) and Tang et al. (2023), the novel FViT addresses the issue of multimodal HAR by adapting the Vision Transformer (ViT) architecture developed by Dosovitskiy et al. (2021) to operate over *fused* images. Crucially for our experiments, FViT has a parameter count invariant to the number of images combined, making the network capacity equivalent between fused and unfused modalities. In addition to the transformer and ResNet34, we use the smallest EfficientNetV2 constructed by Tan & Le (2021) as the communication costs and compute concerns in FL make the smaller network a practical choice.

### 3.1 Partitioning by Privacy Level

The partitions we construct correspond to increasing privacy levels. For example, splitting by human subject implies that each client in that partition only contains data corresponding to one human participant and thus obeys *Subject(User)-level* privacy. Likewise, the partition splitting by participant and room implies that each participant and room combination is treated as a separate client and offers both *Subject(User)-level* and *Environment-level* privacy. The most heterogeneous partition we create treats each participant, room, and modality combination as one client and offers the previous two levels of privacy together with *Modality-level* privacy.

To create a meaningful test set for Federated HAR, we use the data of the sixth client. Since OPERAnet has not been used for FL before, our evaluation compares the accuracy of FL partitioned by subject and environment to the State of The Art centralised baselines using colocated fused modalities. Following this initial investigation, we explore privacy interactions at a subject (user), environmental and modality level when the modalities are never fused and not necessarily colocated. The *separated-modality* experiments are the ones we use to report findings, as they can cover all levels of privacy. Table 1 presents the constructed partitions and their statistics. As we intended to use $30\%$ of total clients each round, we split the data of one participant into two clients based on their room when federating by subject. For the centralised baseline, we follow Koupai et al. (2022) and train for $100$ local epochs, while FL trains for $10$ rounds with $10$ local epochs. The optimiser parameters are kept constant and at parity with Koupai et al. (2022)—see Appendix A.

*Table 1.* The partitions in our experimental setup. They include the centralised baseline (*Centralised*), those partitioned by human subject (*Subj*), by subject and environment (*Subj+Env*), or by subject, environment, and modality (*Subj+Env+Mod*). A partition can contain fused (*F*) or separated (*S*) modalities.

| Partition | Avg Samples | #Samples Subj 6 | #Train Clients | #Clients/R |
|---|---|---|---|---|
| Centralised (F) | $1947.0 \pm 0.0$ | 463 | 1 | 1 |
| Subj (F) | $324.3 \pm 32.3$ | 463 | 6 | 2 |
| Subj + Env (F) | $194.6 \pm 194.6$ | 463 | 10 | 3 |
| Centralised (S) | $5841.0 \pm 0.0$ | 1389 | 1 | 1 |
| Subj (S) | $973.0 \pm 973.0$ | 1389 | 6 | 2 |
| Subj + Env (S) | $583.8 \pm 583.8$ | 1389 | 10 | 3 |
| Subj + Env + Mod (S) | $194.6 \pm 194.6$ | 1389 | 30 | 9 |

*Table 2.* Accuracy results (mean and standard deviation) for model and partition combinations on the test set of OPERAnet. Note the impact of partitioning by modality compared to the subject or environment and the smoother decline in the performance of FViT compared to the CNNs. The "Ensemble" uses the three group models to predict the data label belonging to their modality. The results of F1-Score, shown in Table 3, follow the same trend.

| Partition | FViT | ResNet34 | EffNetB0 | Ensemble |
|---|---|---|---|---|
| Centralised (Fused) | 0.90±0.01 | **0.93±0.01** | 0.91±0.01 | - |
| Subj (Fused) | 0.85±0.02 | **0.90±0.02** | 0.88±0.01 | - |
| Subj+Env (Fused) | 0.83±0.03 | **0.84±0.07** | 0.79±0.04 | - |
| Centralised (S) | 0.83±0.01 | **0.89±0.02** | 0.87±0.01 | - |
| Subj (S) | 0.81±0.01 | 0.84±0.03 | **0.85±0.01** | - |
| Subj+Env (S) | 0.78±0.02 | **0.84±0.02** | 0.80±0.01 | - |
| Subj+Env+Mod (S) | **0.64±0.04** | 0.47±0.03 | 0.55±0.05 | **0.76±0.02** |

### 3.2 Mutual Global and Group Model Learning

To handle separating modalities across clients in a federated network, we propose a group FL structure utilising Deep Mutual Learning (Zhang et al., 2018). Two models are trained on each client and distil knowledge into each other. One model is a globally federated model trained on all clients. The other is a group-level one trained only on clients with a specific modality. The server maintains one model per modality group, providing flexibility for integrating new sensors. We chose the FViT as the global federated model because of its resilience to high heterogeneity in previous experiments. Furthermore, we use the small EfficientNetV2 as the group-level model for future scalability. We present the performance of an ensemble of group-level models, each predicting the relevant modality. Hyperparameters were optimised via Bayesian search, resulting in global and group-level distillation weights of $0.33$ and $0.75$, respectively.

## 4 Evaluation

Our evaluation reveals that Federated Human Activity Recognition is sensitive to sensor heterogeneity but partially resilient to subject characteristics and room structure. In Table 2, we present accuracy results of all partition and model combinations using fused (F) or separated (S) modalities. Fused modality experiments establish a baseline of comparison between federated and centralised training on OPERAnet. Separated modality experiments allow us to investigate more granular levels of privacy and will provide most of the notable figures. Accuracy convergence curves are available per model in Fig. 1; however, they only show unfused modalities to emphasise results for modality-level privacy. In the appendix, convergence curves for fused modalities are presented in Fig. 2 and follow similar trends. All experiments used the Flower (Beutel et al., 2020) FL framework.

### 4.1 Subject(User)-level Privacy

Experiments with user-level privacy showed that all three models achieved results within 3-5% of the centralised baseline, regardless of modality fusion. As shown in Fig. 1, this small gap to the centralised unfused baseline was consistently observed across rounds. In addition, a study by Sozinov et al. (2018) found a similar 4-6% accuracy gap when treating each person as a separate partition, indicating that body characteristics and slight movement differences are not significant enough to generate highly divergent features. These findings suggest that FL is a practical solution for accessing extensive private data from end users. However, unfused modalities reduced accuracy for centralised and subject-level partitions nearly uniformly compared to fused ones—showcasing the benefits of centralisation.

### 4.2 Environment-level Privacy

A further slight-to-medium accuracy degradation is perceptible in the experimental partitions applying subject-level and environmental privacy in Table 2. It is worth noting that data from OPERAnet may have reduced environmental heterogeneity as the same hardware, procedure, and subjects were used in a controlled setting. However, this is a common issue for all HAR systems (Vrigkas et al., 2015, sec.6). Fused modalities saw an additional drop in accuracy of 2-9%, while unfused modalities saw a less significant impact, with the additional maximum drop never exceeding 5%. Notably, ResNet34 operating on unfused modalities did not exhibit a significant accuracy drop when privacy was increased. Fused modalities contain more information about the environment per sample, aiding in distinguishing between rooms. However, the additional information becomes less valuable after samples are split into clients. Figure 1 highlights that the small EfficientNetV2 suffers more from not having examples from both rooms available.

### 4.3 Modality-level Privacy

The experimental results reveal an unexpected pattern when applying privacy at the modality level. The previously top-performing ResNet34 experiences a 37% further drop in accuracy with a 42% total, while the EfficientNetV2 suffers a

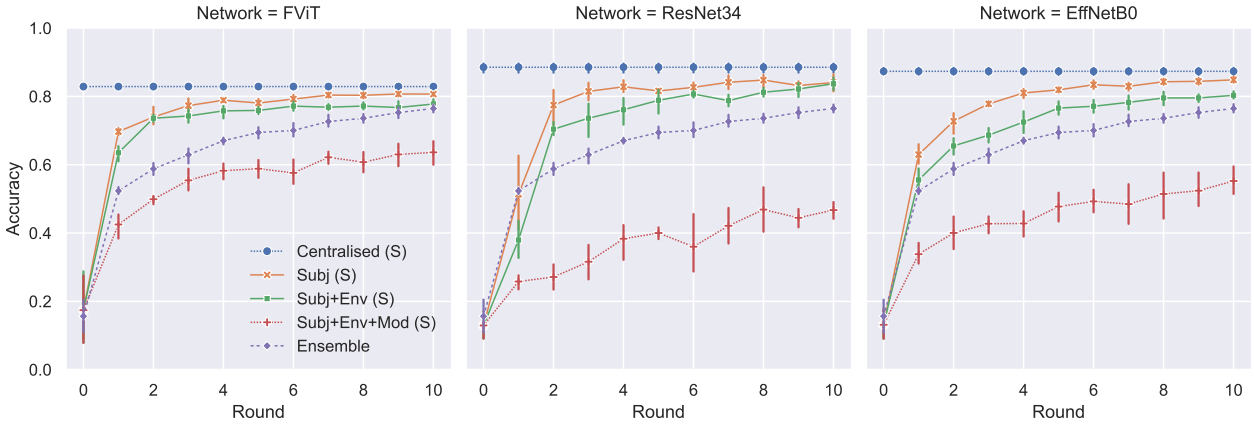



*Figure 1.* Model per-round accuracy on the sixth subject's dataset at all privacy levels. For low-heterogeneity, CNNs generally outperform FViT, but FViT gains a significant advantage when modality partitioning is introduced. Notably, FViT converges faster in the initial rounds and reaches higher accuracy with fewer data in the first two rounds. The "Ensemble" results reverse the effects of modality-level privacy and outperform all non-grouped federated models.

25% further drop in accuracy with a 32% total. By contrast, the Fusion Vision Transformer (FViT), which produced worse results in previous experiments, only experiences a 14% further drop in accuracy with 19% total and emerges as the model with the most significant performance advantage for a given privacy level. To better understand this outcome and the interplay between different model types, we turn to the plot in Fig. 1, which shows the convergence of models for different partitions. The plot immediately reveals the steeper slope of improvement that FViT obtains in the first few rounds. Furthermore, this pattern of performance aligns with the fine-tuning experiments reported in (Koupai et al., 2022), where FViT outperformed ResNet when both were trained on a small amount (1-20%) of data.

The increasing prevalence of IoT devices, surveillance cameras, personal smartphones, and passive RF sensors has led to extensive human activity recognition (HAR) data collection. However, with no uniform regulation or competitive environment, it is critical to prioritise privacy preservation and address the afferent accuracy degradation.

### 4.4 Mutual Learning with Per-modality Group Models

We evaluate the effectiveness of our ensemble, which uses mutual learning to handle the challenges of federated learning across modalities. As demonstrated in both Fig. 1 and Table 2, the ensemble achieves near-equivalent accuracy to FViT on the "Subj+Env" partition with colocated modalities. However, training the federated learning and group-level models simultaneously is costly and difficult to optimise. Moreover, our hyperparameter search, which explored 79 combinations of distillation weights, revealed that the ensemble's performance is sensitive to hyperparameter changes. Meanwhile, the federated model failed to surpass the "Subj+Env+Mod(S)" result in Table 2 through mutual learning, primarily due to the inherent difficulty of multimodal training on the same network without employing specific multi-task techniques.

## 5 Conclusion

We investigated the performance of Multimodal Federated Human Activity Recognition under privacy levels that may arise in practice, such as the subject(user), environmental, and modality levels. Our results show that performance degrades with each additional privacy layer starting with 5-7% for the subject and environmental levels. Remarkably, we observed an *overall* accuracy drop of 32-42% for CNNs when modality-level privacy is assumed. Nevertheless, our experiments determined that a Fusion Vision Transformer architecture performs well in extreme scenarios. Its fast initial convergence with few samples led to an *additional* drop of only 14% with a 19% *overall* drop for modality-level privacy. Furthermore, constructing small group-level models for each modality type trained in a mutual-learning fashion with a global one can limit the *overall* degradation to 7-13%. Such a system can adjust to shifting hardware conditions by incorporating new group-level models and utilising the global model's knowledge for bootstrapping. Despite the clear trends, this work is limited by the size of OPERAnet. Besides larger datasets, other potential future research avenues include hierarchical FL with layered aggregation and creating sparse models with task-based subnetworks.

## REFERENCES


Beutel, D. J., Topal, T., Mathur, A., Qiu, X., Parcollet, T., and Lane, N. D. Flower: A friendly federated learning research framework. *CoRR*, abs/2007.14390, 2020. URL https://arxiv.org/abs/2007.14390.

Bevilacqua, A., MacDonald, K., Rangarej, A., Widjaya, V., Caulfield, B., and Kechadi, M. T. Human activity recognition with convolutional neural networks. In Brefeld, U., Curry, E., Daly, E., MacNamee, B., Marascu, A., Pinelli, F., Berlingerio, M., and Hurley, N. (eds.), *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2018, Dublin, Ireland, September 10-14, 2018, Proceedings, Part III*, volume 11053 of *Lecture Notes in Computer Science*, pp. 541–552. Springer, 2018. doi: 10.1007/978-3-030-10997-4_33. URL https://doi.org/10.1007/978-3-030-10997-4_33.

Bocus, M. J., Li, W., Paulavicius, J., McConville, R., Santos-Rodriguez, R., Chetty, K., and Piechocki, R. Translation resilient opportunistic wifi sensing. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 5627–5633, 2021. doi: 10.1109/ICPR48806.2021.9412263.

Bocus, M. J., Li, W., Vishwakarma, S., Kou, R., Tang, C., Woodbridge, K., Craddock, I., McConville, R., Santos-Rodriguez, R., Chetty, K., and Piechocki, R. Operanet, a multimodal activity recognition dataset acquired from radio frequency and vision-based sensors. *Scientific Data*, 9(1):474, 2022. doi: 10.1038/s41597-022-01573-2. URL https://doi.org/10.1038/s41597-022-01573-2.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

Hard, A., Rao, K., Mathews, R., Beaufays, F., Augenstein, S., Eichner, H., Kiddon, C., and Ramage, D. Federated learning for mobile keyboard prediction. *CoRR*, abs/1811.03604, 2018. URL http://arxiv.org/abs/1811.03604.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 770–778. IEEE Computer Society, 2016. doi: 10.1109/CVPR.2016.90. URL https://doi.org/10.1109/CVPR.2016.90.

Jobanputra, C., Bavishi, J., and Doshi, N. Human activity recognition: A survey. *Procedia Computer Science*, 155:698–703, 2019. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2019.08.100. URL https://www.sciencedirect.com/science/article/pii/S1877050919310166. The 16th International Conference on Mobile Systems and Pervasive Computing (MobiSPC 2019),The 14th International Conference on Future Networks and Communications (FNC-2019),The 9th International Conference on Sustainable Energy Information Technology.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K. A., Charles, Z., Cormode, G., Cummings, R., D'Oliveira, R. G. L., Eichner, H., Rouayheb, S. E., Evans, D., Gardner, J., Garrett, Z., Gascón, A., Ghazi, B., Gibbons, P. B., Gruteser, M., Harchaoui, Z., He, C., He, L., Huo, Z., Hutchinson, B., Hsu, J., Jaggi, M., Javidi, T., Joshi, G., Khodak, M., Konečný, J., Korolova, A., Koushanfar, F., Koyejo, S., Lepoint, T., Liu, Y., Mittal, P., Mohri, M., Nock, R., Özgür, A., Pagh, R., Qi, H., Ramage, D., Raskar, R., Raykova, M., Song, D., Song, W., Stich, S. U., Sun, Z., Suresh, A. T., Tramèr, F., Vepakomma, P., Wang, J., Xiong, L., Xu, Z., Yang, Q., Yu, F. X., Yu, H., and Zhao, S. Advances and open problems in federated learning. *Found. Trends Mach. Learn.*, 14(1-2):1–210, 2021. doi: 10.1561/2200000083. URL https://doi.org/10.1561/2200000083.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., and Hadsell, R. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017. doi: 10.1073/pnas.1611835114. URL https://www.pnas.org/doi/abs/10.1073/pnas.1611835114.

Koupai, A. K., Bocus, M. J., Santos-Rodriguez, R., Piechocki, R. J., and McConville, R. Self-supervised multimodal fusion transformer for passive activity recognition. *IET Wireless Sensor Systems*, 12(5-6):149–160, 2022. doi: https://doi.org/10.1049/wss2.12044. URL https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/wss2.12044.

Kwon, H., Tong, C., Haresamudram, H., Gao, Y., Abowd, G. D., Lane, N. D., and Ploetz, T. Imutube: Automatic extraction of virtual on-body accelerometry from video for human activity recognition, 2020. URL https://arxiv.org/abs/2006.05675.

Li, W., Bocus, M. J., Tang, C., Vishwakarma, S., Piechocki, R. J., Woodbridge, K., and Chetty, K. A taxonomy of wifi sensing: Csi vs passive wifi radar. In *2020 IEEE Globecom Workshops (GC Wkshps*, pp. 1–6, 2020. doi: 10.1109/GCWkshps50303.2020.9367546.

Li, W., Bocus, M. J., Tang, C., Piechocki, R. J., Woodbridge, K., and Chetty, K. On csi and passive wi-fi radar for opportunistic physical activity recognition. *IEEE Transactions on Wireless Communications*, 21(1):607–620, 2022. doi: 10.1109/TWC.2021.3098526.

Li, Z. and Hoiem, D. Learning without forgetting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(12):2935–2947, 2018. doi: 10.1109/TPAMI.2017.2773081. URL https://doi.org/10.1109/TPAMI.2017.2773081.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In Singh, A. and Zhu, X. J. (eds.), *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*, volume 54 of *Proceedings of Machine Learning Research*, pp. 1273–1282. PMLR, 2017. URL http://proceedings.mlr.press/v54/mcmahan17a.html.

Ouyang, X., Xie, Z., Zhou, J., Huang, J., and Xing, G. Clusterfl: a similarity-aware federated learning system for human activity recognition. In Banerjee, S., Mottola, L., and Zhou, X. (eds.), *MobiSys '21: The 19th Annual International Conference on Mobile Systems, Applications, and Services, Virtual Event, Wisconsin, USA, 24 June - 2 July, 2021*, pp. 54–66. ACM, 2021. doi: 10.1145/3458864.3467681. URL https://doi.org/10.1145/3458864.3467681.

Ronald, M., Poulose, A., and Han, D. S. isplinception: An inception-resnet deep learning architecture for human activity recognition. *IEEE Access*, 9:68985–69001, 2021. doi: 10.1109/ACCESS.2021.3078184.

Sheller, M. J., Edwards, B., Reina, G. A., Martin, J., Pati, S., Kotrotsou, A., Milchenko, M., Xu, W., Marcus, D., Colen, R. R., and Bakas, S. Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data. *Scientific Reports*, 10(1):12598, 2020. doi: 10.1038/s41598-020-69250-1. URL https://doi.org/10.1038/s41598-020-69250-1.

Sozinov, K., Vlassov, V., and Girdzijauskas, S. Human activity recognition using federated learning. In Chen, J. and Yang, L. T. (eds.), *IEEE International Conference on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications, ISPA/IUCC/BDCloud/SocialCom/SustainCom 2018, Melbourne, Australia, December 11-13, 2018*, pp. 1103–1111. IEEE, 2018. doi: 10.1109/BDCloud.2018.00164. URL https://doi.org/10.1109/BDCloud.2018.00164.

Tan, M. and Le, Q. V. Efficientnetv2: Smaller models and faster training. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 10096–10106. PMLR, 2021. URL http://proceedings.mlr.press/v139/tan21a.html.

Tang, Y., Zhang, L., Min, F., and He, J. Multiscale deep feature learning for human activity recognition using wearable sensors. *IEEE Transactions on Industrial Electronics*, 70(2):2106–2116, 2023. doi: 10.1109/TIE.2022.3161812.

Vrigkas, M., Nikou, C., and Kakadiaris, I. A. A review of human activity recognition methods. *Frontiers Robotics AI*, 2:28, 2015. doi: 10.3389/frobt.2015.00028. URL https://doi.org/10.3389/frobt.2015.00028.

Xiao, Z., Xu, X., Xing, H., Song, F., Wang, X., and Zhao, B. A federated learning system with enhanced feature extraction for human activity recognition. *Knowl. Based Syst.*, 229:107338, 2021. doi: 10.1016/j.knosys.2021.107338. URL https://doi.org/10.1016/j.knosys.2021.107338.

Zhang, Y., Xiang, T., Hospedales, T. M., and Lu, H. Deep mutual learning. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 4320–4328. Computer Vision Foundation / IEEE Computer Society, 2018. doi: 10.1109/CVPR.2018.00454. URL http://openaccess.thecvf.com/content_cvpr_2018/html/Zhang_Deep_Mutual_Learning_CVPR_2018_paper.html.

Zhao, Y., Liu, H., Li, H., Barnaghi, P. M., and Haddadi, H. Semi-supervised federated learning for activity recognition. *CoRR*, abs/2011.00851, 2020. URL https://arxiv.org/abs/2011.00851.

# A APPENDIX

*Table 3.* The F1-Score results of partition-model combinations. The same trends from the accuracy comparisons repeat themselves with higher privacy requirements leading to worse performance. A similar strong decline can be observed when clients are partitioned by modality, with the FViT performing the best in the most heterogeneous condition despite trailing behind the CNNs for all other partitions. The ensemble group models also successfully recovered performance near the FViT levels when the partitioning was based only on subject and environment.

| Partition | FViT | ResNet34 | EffNetB0 | Ensemble |
|---|---|---|---|---|
| Centralised (Fused) | 0.80± 0.03 | **0.86 ± 0.02** | 0.83± 0.02 | - |
| Subj (Fused) | 0.73± 0.03 | **0.82 ± 0.05** | 0.78± 0.02 | - |
| Subj+Env (Fused) | 0.70± 0.05 | **0.74 ± 0.08** | 0.64± 0.04 | - |
| Centralised (Split) | 0.71± 0.01 | **0.80± 0.03** | 0.78± 0.02 | - |
| Subj (Split) | 0.68± 0.01 | 0.72± 0.06 | **0.73± 0.02** | - |
| Subj+Env (Split) | 0.64± 0.03 | **0.71± 0.04** | 0.67± 0.02 | - |
| Subj+Env+Mod (Split) | **0.50± 0.04** | 0.35± 0.03 | 0.39± 0.04 | **0.60± 0.03** |

The preprocessing pipeline we use is precisely described in Koupai et al. (2022); however, we shall offer a brief summary here. First, the CSI signal is denoised using a discrete wavelet transform and median filtering before applying PCA and generating a spectrogram through the STFT. Then, for the PWR data, the authors apply the cross ambiguity function to the PWR data followed by the CLEAN algorithm and the outputting of a Doppler spectrogram. We use three of the image types they generate. First, we use the concatenated spectrograms generated from the three-receiver surveillance channels of the PWR data. The combined images from the three channels have a dimension of $224 \times 672$. Second, we use the spectrograms generated using STFT on amplitude CSI data from two receivers with a concatenated size of $224 \times 448$. Third, we use the phase-difference spectrograms generated via STFT from the phase-difference CSI data from the two receivers with a concatenated size of $224 \times 448$. Combined in the fused partitions, they add up to $224 \times 1568$ images. Finally, we take the largest image type ($224 \times 672$) in unfused partitions and pad the rest.

Client data partitions are generated in order of person index for split-subject modalities, person index and then room index for subject and environment, and subject, room, and modality index for the final partitioning. Our indexing assumes the human subjects are ordered from one to six, the rooms from one to two, and the modalities from one to three in the above order. The subject and room indexes are directly available in the dataset. Each model and partition combination was run using five distinct seeds generating the same client sequence across models. Thus differences in performance between models are not due to randomness in client selection. The seeds we use are $42, 1337, 3407, 8711, 9370$, and the client sequence is generated by calling `np.random.choice` for the given number of clients per round out of the entire population for each of the ten rounds at the start of the script right after the seed has been set. The mean and standard deviation are reported based on the five seeds in and Table 2 and Table 3. The per-round values in Fig. 1 and Fig. 2 have their mean and standard deviation calculated based on the accuracy of the models on each of the five seeds at the specific round. Before every experiment, the same seeds are used to set the random, NumPy, and Torch modules in Python.

All models have been trained as in Koupai et al. (2022) using AdamW with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ with a weight decay of $0.01$ and batch size of 10 rather than 64 due to the small size of the federated partitions. The computational resources involved four Nvidia A40s and were extensively used during parameter tuning.

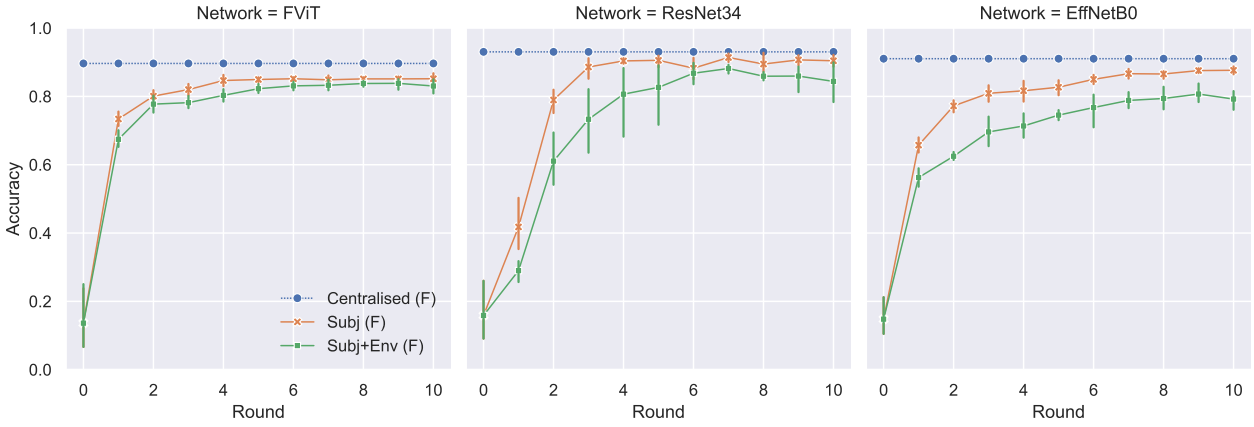



*Figure 2.* Model per-round accuracy on the fused-modality dataset. The trends observed resemble those for the split partitions with only a uniform decrease in accuracy by comparison. The only major change in results is the sensitivity of ResNet34 to environmental privacy.

# ROBUST AND PRIVATE MULTIMODAL FEDERATED HUMAN ACTIVITY RECOGNITION

**Alex Iacob** [1] **Pedro P. B. Gusmão** [1] **Nicholas Donald Lane** [1]

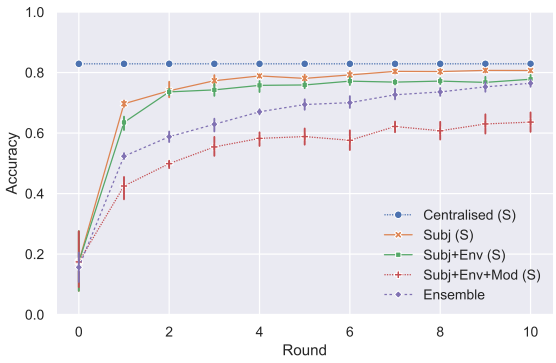



*Figure 1.* Accuracy of the model at increasing privacy levels. Ensemble refers to the accuracy of group-level models trained/tested on their afferent modality on the "Subj+Env+Mod" partition.

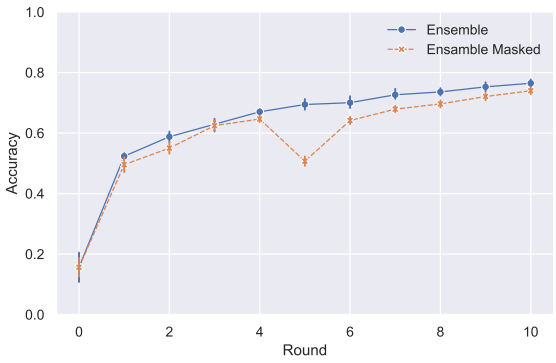



*Figure 2.* Default performance of the ensemble vs the average performance we obtain when successively masking clients from each of the three modalities up to the 5th round.

Human Activity Recognition (HAR) involves classifying human actions, such as running or sitting, using data from personal devices like smartphones or environmental sensors. However, privacy requirements impose data collection limitations. In this work, privacy requirements refer to constraints on collecting or centralising data at three levels:

**User-level Privacy** For gyroscope or accelerometer data from smartphones and wearables, end-users may be unwilling to share personal information.

**Environment-level Privacy** For locations such as hospitals and internment facilities, sensitive information must often remain private from third parties.

**Modality-level Privacy** Data generated from different groups of sensors may be owned by competing entities or raise ethical concerns if collected in public spaces.

A Federated Learning (FL) approach keeps data encapsulated in clients at the necessary privacy level during training. Our work brings the following contributions to Federated Human Activity Recognition:

1. First, we evaluate the performance of a Multimodal Vision Transformer [2] trained in a federated fashion on the recent multimodal OPERAnet [1] dataset. Unlike other works, we investigate the additive effects of privacy up to the complete separation of each user, environment, and modality combination.

2. Second, we show that privacy at the modality level results in the *highest* accuracy cost, followed by the environmental level and then the user level. To mitigate this, we propose mutual learning of smaller group-level models (EfficientNetV2B0) alongside the FL model to cover modalities that cannot be colocated in a single client. Our results in Fig. 1 indicate that this method can significantly reduce accuracy degradation. Furthermore, Fig. 2 shows that our approach is resilient to adding new modalities during training.

[1]University of Cambridge: {aai30,pp524,ndl32}@cam.ac.uk.

## REFERENCES

[1] M. J. Bocus, W. Li, S. Vishwakarma, R. Kou, C. Tang, K. Woodbridge, I. Craddock, R. McConville, R. Santos-Rodriguez, K. Chetty, and R. Piechocki. Operanet, a multimodal activity recognition dataset acquired from radio frequency and vision-based sensors. *Scientific Data*, 9(1):474, 2022. doi: 10.1038/s41597-022-01573-2. URL https://doi.org/10.1038/s41597-022-01573-2.

[2] A. K. Koupai, M. J. Bocus, R. Santos-Rodriguez, R. J. Piechocki, and R. McConville. Self-supervised multimodal fusion transformer for passive activity recognition. *IET Wireless Sensor Systems*, 12(5-6):149–160, 2022. doi: https://doi.org/10.1049/wss2.12044. URL https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/wss2.12044.

# High-throughput Simulation of Federated Learning via Resource-Aware Client Placement

Lorenzo Sani*
ls985@cam.ac.uk
University of Cambridge

Pedro Porto Buarque de Gusmão*
pp524@cam.ac.uk
University of Cambridge

Alex Iacob*
aai30@cam.ac.uk
University of Cambridge

Wanru Zhao
wz341@cam.ac.uk
University of Cambridge

Xinchi Qiu
xq227@cam.ac.uk
University of Cambridge

Yan Gao
yg381@cam.ac.uk
University of Cambridge

Javier Fernandez-Marques
jafermarq@gmail.com
Samsung AI

Nicholas Donald Lane
ndl32@cam.ac.uk
University of Cambridge

## ABSTRACT

Federated Learning (FL) is the privacy-preserving machine learning paradigm which collaboratively trains a model across millions of devices. Simulated environments are fundamental to large-scale FL research, allowing researchers to quickly test new ideas to solve system and statistical heterogeneity issues. This work proposes *Pollen*, a novel resource-aware system capable of speeding up FL simulations by efficiently placing clients across distributed and heterogeneous hardware. We propose minimising server-GPU communication and using an efficient client placement policy based on the inherent trade-offs of FL client placement on heterogeneous GPUs. These trade-offs are explored experimentally.

This exploration has been conducted via relevant baselines on three popular FL tasks: image classification, speech recognition and text generation. We compare *Pollen* to existing ad-hoc FL frameworks, such as Flower, Flute and FedScale, and show performance gains of 50% to 400%.

## CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; *Neural networks*; **Parallel computing methodologies**; **Parallel algorithms**; *Massively parallel algorithms*; **Simulation environments**; • **Computer systems organization**;

## KEYWORDS

federated learning, simulation, scalability, deep learning, resource management

*These authors contributed equally to this research.

## 1 INTRODUCTION

Machine learning (ML) is becoming increasingly viable on constrained hardware with low memory such as smartphones, wearables, and general IoT devices. Techniques originally developed for training models on smartphones with hundreds of MB of memory [10] can now run on micro-controllers with 256kB [19]. Federated learning (FL) was introduced by McMahan et al. [21] to allow training on thousands to millions of such edge clients in a distributed manner while avoiding the privacy and network costs of communicating their sensitive data in a centralised fashion.

While Federated learning allows edge devices to collaborate effectively, Kairouz et al. [12, sec. 3.1] indicate that it brings unique challenges relating to clients' different hardware and data distributions. To effectively model and address such challenges, researchers must be able to run large-scale Federated Learning simulations efficiently on their hardware.

Unlike centralised ML, dominated by long-running execution of large jobs, simulating federated learning relies on the repeated execution of small clients corresponding to the resource-constrained devices it intends to emulate. Since such clients cannot keep GPU utilisation high by themselves, scheduling many of them to run both on the same GPU and across GPUs is beneficial. Crucially, clients can vary in dataset size by orders of magnitude. Independently on the client selection procedure, existing FL frameworks [2, 7, 16] treat all clients equally during simulations and can often end up with straggling GPUs. A proper *placement* of clients on GPUs could significantly reduce training time and allow more extensive simulations. Unfortunately, such intelligent resource-based *placement* is impossible in existing FL frameworks as they rely on a *pull-based* system where workers sequentially sample clients from a server queue.

In this work, we propose *Pollen*, an adaptive *push-based* method for client placement in simulated FL, that is compatible with diverse client selection procedures, and show that it significantly improves FL training times. Instead of building an entirely new framework, we chose to modify Flower [2] due to its flexibility. We showcase the design of our system and its effectiveness and analyse the factors which decide the training time of the simulation for a given placement policy. Our contribution is *threefold:*

(1) We experimentally show a proportional but nonlinear relationship between the size of a clients' dataset and their training time across multiple workloads and GPU types. We argue that this requires intelligent client placement to optimise training time when combined with the skewness of standard FL datasets.
(2) We build *Pollen*, a client placement system to effectively partition clients amongst potentially heterogeneous GPUs using several placement strategies. Then, we compare the effectiveness of such strategies. Our results show that for heterogeneous GPU environments, a learning-based policy outperforms others by up to 81% as it can accurately learn to predict training time regardless of system configuration.
(3) Finally, we show that *push-based* client placement can significantly improve training time over previous *pull-based* systems. *Pollen* achieves a 50% to 378% reduction in training time across various datasets when training 10 000 clients split evenly across 100 rounds with heterogeneous GPUs. Moreover, for homogeneous GPUs, our method decreases communication costs enough to obtain a 200% to 400% improvement over the next-fastest system.

Our method allows FL researchers to run more extensive, faster, and scalable simulations. These improvements permit rapid prototyping and development of algorithms for production settings involving millions of edge devices.

## 2 SIMULATING FEDERATED LEARNING

We now introduce federated learning and describe the main characteristics of the standard simulation solutions adopted by popular federated learning frameworks. We then justify our proposed system *Pollen* based on the observed limitations of previous systems.

### 2.1 Federated Learning

Federated Learning is concerned with training ML models in a distributed fashion while minimising communication costs and maintaining private data on-device. In its most popular cross-device version [12], it takes the form of synchronised training where many clients pool their resources to train a model collaboratively. Such devices can differ significantly both in terms of their local data and in terms of available hardware. For example, a group of devices for human activity recognition could be composed of smartphones containing gyroscopes and accelerometers [25, 26], surveillance video cameras [15], and passive sensing devices using Radio-Frequency data [13]. Clients in these diverse categories would all have highly divergent data modalities, dataset sizes, computational power, network speed and training availability. Human activity recognition data can be sensitive and often needs to stay private to the point of origin, thus requiring a federated approach.

Federated Learning algorithms, like Federated Averaging [21] for cross-device FL, maintain data privacy via a client-server training design synchronised across *rounds*. The server controls the training and holds the federated model. It sends the model to each client at the start of the round, where it is trained on private data using Stochastic Gradient Descent (SGD). Then, the clients return the models to the server, aggregating them to create a new federated model for the next round.

### 2.2 Framework Simulation Engines

The simulation engines of FL frameworks aim to enable experiments in a constrained environment where we may not have enough resources to virtualise all the clients in the FL setting or all the clients in the sampled cohort for each round. We can better understand this scenario by representing each client's training as a job to schedule. Each of these jobs needs to allocate resources to account for a complete copy of the model to train, the dataset for the client, including the preprocessing, and the instructions for the training. The needs of a single client are usually relatively small; as such, it is possible to fit many of them into a single GPU.

Given this setting, FL simulation engines try orchestrating the training using a server-workers paradigm. The server orchestrates the simulated FL training by serving clients to workers and aggregates the results after every round of training. The workers are responsible for training clients individually and sending the trained models to the server once the training is complete. Ad-hoc FL frameworks, such as Flute [7], FedScale [16] and Flower [2], rely on this paradigm.

It is worth highlighting that workers in both FedScale and Flute are statically allocated to their GPU. On the other hand, Flower's Virtual Client Engine, developed by Beutel et al. [2] and based on Ray [22], can dynamically move workers between GPUs during the FL training. However, the control over this dynamic allocation is limited. In all these frameworks, the server serves clients to workers using a pull-based

queuing system involving many communication steps between the server and workers. The execution of an FL round on this system can be summarised as follows.

(1) At the start of a round, the server samples a subset of participating clients. This subset defines the cohort of clients to be trained in that round. The cohort is kept in a synchronised queue, allowing workers to read clients sequentially.
(2) Each worker reads from the queue, extracts the first client, and starts the training. Different workers cannot access the same element of the synchronised list.
(3) Once a worker finishes training a client, it pings the server to notify that the client's training is completed.
(4) When the server receives this message, it replies to the worker when it can receive the training results.
(5) The worker finally sends the results to the server that will store them or partially aggregate them, depending on the experiment's configuration.

The number of workers controls the concurrency of this embarrassingly parallel simulation. Hence, it is beneficial to increase the number of workers up until the resources of the system are saturated, or gains from concurrency cease.

### 2.3 Limitations of Current Systems

The limitations of pull-based queuing systems are multifold. Critically, the workers of a specific GPU cannot choose which client to train. This underlying lack of control can limit the simulator's options when attempting to train specific clients on specific GPUs. For example, when disproportionately large clients are selected, balancing client training time across GPUs and avoiding stragglers is impossible.

Another general limitation is that the communication involved may take a significant amount of time relative to the training time of most clients. Such communication bottlenecks are significant for settings with multiple machines (nodes) that need a network connection to communicate with each other. In this work, *multinode* refers to hardware configurations with GPUs distributed amongst separate machines.

We now present the specific limitations of each framework addressed in our paper.

**Flute**, introduced by Dimitriadis et al. [7], is optimised to use the *nccl* backend of PyTorch Distributed [17] when running on GPUs and the *gloo* backend for CPU training. It can only run a single worker per GPU and, because it cannot intermix GPU and CPU training, it requires an entire GPU to hold the parameter server that handles aggregation during FL simulation. This can be wasteful as aggregation is usually not a compute-intensive operation. These issues cannot be easily addressed as the codebase has highly coupled components dependent on this design.

**FedScale**, introduced by Lai et al. [16], depends on unreliable configurations of gRPC, which is felt across the many communication steps necessary for the pull-based design. As such, longer rounds may cause crashes as clients appear to either disconnect or time out. Furthermore, despite being able to place multiple workers on the same GPU, later experiments show minimal benefits when increasing either the number of workers or GPUs. Each worker is also tasked with loading the entire dataset, even if they are on the same node as other workers and can share memory. Finally, similarly to Flute, the codebase coupling level makes refactoring difficult.

**Flower**, introduced by Beutel et al. [2], depends on Ray [22] as its simulation engine. Ray may cause out of memory (OOM) in a multi-GPU configuration as workers do not fully deallocate memory from previously used GPUs. Careful configuration of parameters helps avoid OOM at the cost of slowing down the overall training. However, unlike Flute and FedScale, the codebase is highly modular, allowing us to implement our new simulation engine on top of existing components.

General limitations of GPU scheduling for ML tasks are also worth mentioning. Job scheduling on a GPU cluster for traditional ML tasks is a well-studied problem [8, 29, 30]. For example, Gandiva [29] schedules large jobs on such a cluster by profiling the rate at which they process mini-batches. At the same time, CODA [30] attempts to balance the CPU assignment of GPU jobs to optimise data loading. However, both rely on the assumption that ML tasks are highly repetitive and run long enough to be effectively optimised. As we will show in Section 3, client dataset sizes in FL are much smaller than a typical ML job and more difficult to profile. Furthermore, client dataset sizes are highly skewed, making round durations unpredictable. Together, these two considerations make the heuristics of existing GPU scheduling systems insufficient for FL. This insufficiency drives us to propose an FL-specific solution rather than plugging in an existing one.

## 3 HETEROGENEITY AND CLIENT PLACEMENT

We now describe the practical difficulties of deciding which GPU a client should be placed on for training. For these, we identify two causes in the form of client heterogeneity and hardware heterogeneity.

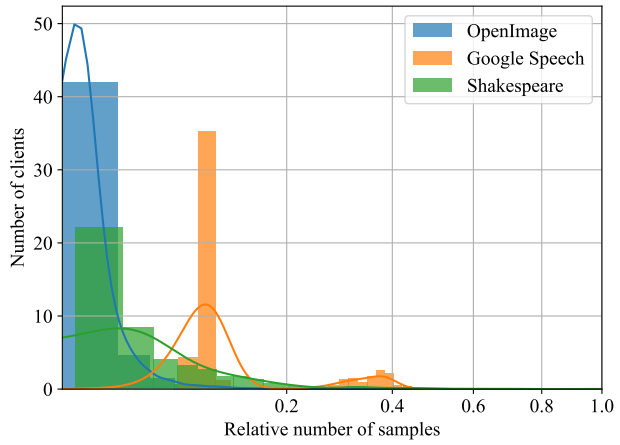



**Figure 1: Dataset size distribution over clients for OpenImage [14], Google Speech [28], and Shakespeare [4]. Natural non-uniformity in data distributions will lead to different training times. A non-linear scale was chosen for the x-axis.**

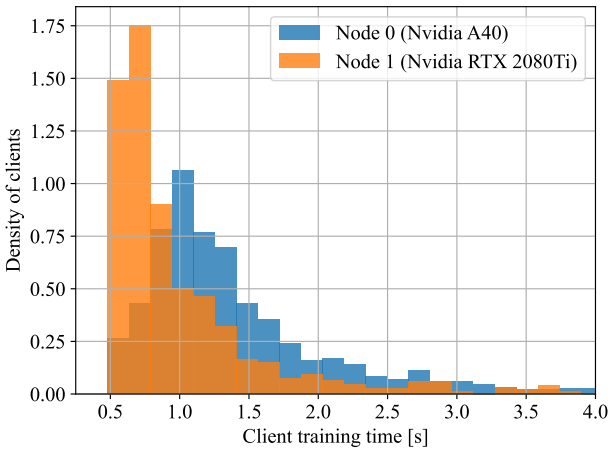



**Figure 2: Training times for two GPUs running on different client dataset sizes. The GPUs are running the same clients.**

## 3.1 Client Heterogeneity

In large-scale cross-device FL, clients collect or produce data at vastly different rates [18]. Efficient simulations should reflect this fact and account for the different training times that simulated clients will take. While some frameworks try to circumvent this issue by fixing the number of steps each client will train for [16], this assumes that the dataset distribution is not highly skewed and leads to two issues.

First, it limits the contribution of clients having large datasets. Second, clients having small datasets are forced to reuse their data (increase in local epochs). Consequently, we refrain from fixing a constant number of training steps throughout our experiments and argue that this is not a reasonable assumption to be made at a framework level. Fig. 1 shows the distributions of samples for three different naturally-partitioned datasets commonly used in FL simulations. FL datasets are usually long-tailed and skewed, leading to the abovementioned issues.

We argue that the dissimilarity in dataset distributions makes it necessary to design dynamically adaptive placement strategies capable of accommodating different datasets. Additionally, the various pre-processing pipelines such datasets use for mini-batches reinforce this requirement. For example, image datasets may require samples to be loaded from disk and transformed, while language datasets may store features in memory. We further argue that the internal skewness of a dataset requires intelligent placement of clients on devices even for the simplest case of *homogeneous* GPU configurations. For example, in extreme situations, one device may train clients with orders of magnitude more batches than another.

## 3.2 Hardware Heterogeneity

Besides client heterogeneity, it is necessary to consider the different GPUs used in FL simulations. As previously mentioned, individual client workloads are smaller than traditional ML tasks and are highly parallelisable, allowing them to be trained across various machines with little effort. However, using heterogeneous GPUs may result in workers finishing processing their clients at very different times, leading to unnecessarily long experiments.

Roughly speaking, training a single client will depend on *data loading* and *actual training*. Data loading and preprocessing generally happen on the CPU. As the number of concurrent CPU jobs increases beyond the number of CPU cores, the variability of the data loading time grows proportionally to that of the scheduling. This can act as a bottleneck when increasing the number of workers available for processing.

The actual training of a client usually happens inside the GPU, and the time it takes is affected by the client's local dataset size. For large clients, the training time is approximately determined by the average speed at which the GPU can process each of their batches. However, for small clients, the startup times are a more significant section of the total time the clients spend executing, which causes increased variability in total training time. Furthermore, we observe some variability even for huge clients, which should be the least affected by startup concerns.

Figure 2 shows the training time distribution of two Nvidia GPUs running the same population of clients with different dataset sizes. As can be observed from the figure, the two GPUs perform very differently from one another. Therefore,

allocating less work to the slower GPU is necessary to optimise training time by having them finish simultaneously.

## 4 POLLEN DESIGN

This section describes the key ideas and components used in our proposed solution. An overview of the complete system, dubbed *Pollen*, can be seen in Fig. 3.

### 4.1 Placement Strategy

Following our investigation in Section 2, we propose an alternative client placement system that addresses the issues reported in Section 2.3 regarding the limitations of FL frameworks. In our approach, instead of having workers *requesting* clients from the server, the system follows a *placement strategy* to partition client workloads across workers and perform a *push-based* allocation. This method allows us to reduce the number of communication steps between the server and nodes and to assign sets of clients to appropriate GPUs.

It is worth mentioning that our placement method acts on the underlying simulation layer of the FL framework. It is an independent procedure from *client selection* and is not affected by the sampling procedure used to generate the clients for a specific round nor any other algorithmic properties. It can be easily extended to use other sampling techniques such as FedCS [23], Power-of-Choice [6] and DivFL [1].

### 4.2 Resource Allocator:

Clients having different amounts of data and being trained on heterogeneous GPUs will produce disparities in workers' execution times. A solution to this is to be able to associate client workloads with appropriate GPUs.

At the beginning of the FL simulation, the *resource allocator* module receives information from all training nodes regarding their available hardware, e.g., the number of CPU cores and the number and types of GPUs. This information is used to allocate resources to workers, following user-defined constraints, such as the maximum number of workers per GPU, which can be estimated by profiling a single inference step for each GPU type contained in the node.

### 4.3 Partial Aggregation:

When using associative aggregation strategies, such as FedAvg [21], the system can benefit from partially aggregating results within a worker before sending the partial result for a last aggregation on the server.

In this approach, the worker keeps both a partially aggregate model $\theta_k^p$ and a total number of processed data samples $N_k$ after having trained its $k$-th client, as seen in Eq. (1).

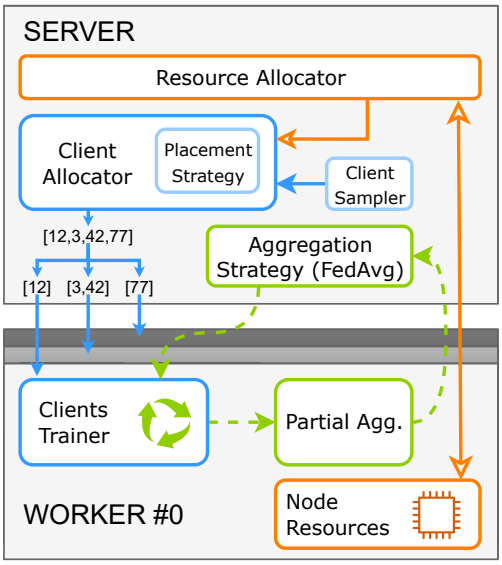



**Figure 3: Diagram describing *Pollen* with its relevant elements in both server and worker. The colour code has been used to distinguish between components related to clients (blue), models (green), and hardware (orange).**

$$\theta_{k+1}^p = \frac{\theta_k^p \times N_k + \theta_{k+1} \times n_{k+1}}{N_{k+1}} \tag{1}$$

$$N_{k+1} = N_k + n_{k+1} \tag{2}$$

Once the *worker* has completed training its list of clients, it will send both the partially aggregated model and the total sum of the samples for the final aggregation.

### 4.4 Client Allocator:

We follow the concept from Section 2 and define a *worker* as an entity capable of sequentially training lists of clients.

As FL clients can only use small batch sizes and models when training on edge devices, packing many workers and oversubscribing GPUs has proven beneficial in realistic FL simulations, as seen in Fig. 4.

The *client allocator* associates clients with individual workers on specific GPUs. The server samples a list of participating clients at the beginning of a round. Then, it passes this module, which uses a pre-defined *client placement policy*, discussed in Section 5, to determine which worker will train which client and in what order.

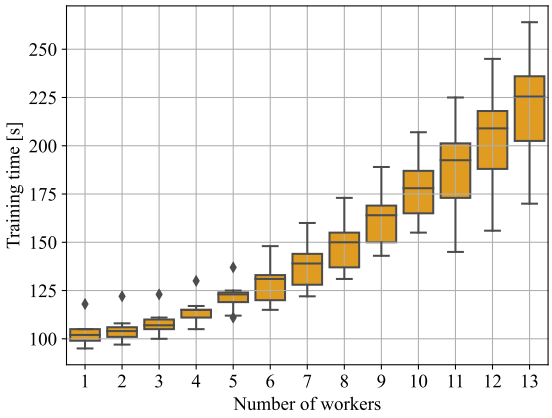



**Figure 4: The distribution of clients' training time for different values of concurrent workers on the same GPU. Every worker has the same load: the same list of 100 clients. For example, with 5 workers, the GPU trains 500 clients using 5 concurrent processes.**

## 5 CLIENT PLACEMENT STRATEGIES

This section discusses the strategies for placing clients on GPUs that we have explored in this work.

Since our design allows the simulation to have only one communication step from the server to the workers, our exploration focused on distributing the list of $N$ randomly sampled clients across workers in one step. For each round, all these strategies receive the list of integer clients' IDs to be trained, and they return a list of clients' IDs for each worker indicating which clients it should train. Since the exploration space for determining the best distribution procedure is very broad, we rely on FL features common to most experiments.

**Naíve round-robin (RR):** This strategy represents the starting point of our investigation since it naively splits the list of clients in $k$ uniformly populated lists, where $k$ is the number of workers. In particular, the first sampled client will be assigned to the first worker and the second client to the second worker, etc. If $\frac{N}{k}$ is not an integer, the remainder is distributed across the first workers.

**Batch-sorted round-robin (SRR):** The first information we used in this study was the number of batches $m$ each client has. As discussed above, the number of batches is a proxy for the training time in epochs-based FL training. Before naively splitting the list as before, this strategy orders the clients by $m$ from top to bottom. The intuition behind this procedure is that different workers will likely train the biggest clients.

**Batch-Uniform distribution (BU):** A step forward to the previous strategy is to use the same information while changing the distribution procedure. For homogeneous GPUs, we want the load across workers to be balanced. To achieve this, the strategy loops over the $N$ clients after ordering them by $m$ from top to bottom and assigns the current client to the worker whose load is lower. The load is estimated by summing the number of batches of all the clients assigned to the worker. It has to be noted that the first $k$ clients of the list are assigned the same way as the previous strategy.

**Learning-based time prediction (LB):** In settings with heterogeneous GPUs, the proxy for the training time given by $m$ may not be sufficient, as shown in previous sections. Our learning-based strategy predicts the training time for each GPU. The first FL round will use the naíve round-robin strategy to collect data about client training time from all available workers. Starting from the second round, the strategy builds one dataset for each GPU composed of tuples $(client\ training\ time,\ m)$ from previous rounds.

Then, for each dataset, the strategy fits the data points to the function in Eq. (3), where $y$ represents the client training time, and $x$ is the number of batches the client has.

$$y = ax + b\log(cx) + d \tag{3}$$

The parameters derived from the fitting are then used for predicting the training time of the clients sampled in the current round. We chose Eq. (3) in order to match the skewed training time distribution we observed empirically in Fig. 1, this choice is further discussed at the end of Section 5.1. The strategy sorts the workers by GPU type, from the fastest to the slowest, using the predicted training time of the biggest client in the current cohort. Finally, the strategy carries on the placement of clients by balancing the load between workers, similar to the batch uniform strategy. Clients are ordered by $m$ from top to bottom and assigned to the worker whose load is lower. Here, the load is estimated by summing the predicted training time of all the clients assigned to the worker.

### 5.1 Efficient Client Placement

In this work, we try to answer the question of which placement strategy is better to choose in which context. More importantly, we argue that discussing placement strategies in simulating FL is necessary. The research on large-scale [3, 5, 27] FL often relies on simulating FL settings with large cohorts (in the range $[10^2, 10^4]$) of clients over a relatively limited amount of hardware resources for thousands of rounds. Reducing the latency of experiments as much as possible while exploiting the hardware resources at their best is fundamental. Regardless of the chosen placement strategy, we expect any one-step communication method to outperform

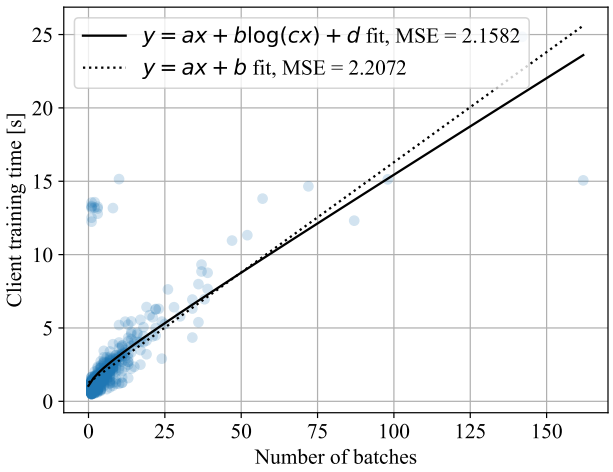



**Figure 5: Clients' training times are plotted against their number of batches. The fitting lines of the linear function and the proposed function are shown alongside The Mean Squared Errors (MSE) derived from the fitting procedure.**

the queue design in previously mentioned frameworks. However, we will show that an optimal placement can improve the simulation speed by up to 81% compared to the slowest strategy.

Different placement strategies will result in different distributions of clients to train across workers. However, this does not hold for the degenerate placements in which the number of clients trained per round equals the number of workers, and the workers are allocated on the same GPU and node. The Shakespeare baseline is the only one we have explored that can fall into this degenerate case. This is because the training procedure plans to train 10 clients per round, which can easily be fitted into a single GPU. Excluding the degenerate placement, the non-trivial metric that is impacted by different placement strategies is what we call "*timedelta workers*". We define "*timedelta workers*" as the time difference between the timestamp at which the fastest worker finishes their job and the timestamp at which the slowest worker finishes their job, where we intend the job to be the training of all the clients in the received list. When using the RR strategy, we observe that the distribution across rounds of "*timedelta workers*" is peaked at a time at least one order of magnitude greater than the training time of the smallest client in that round. We could have reduced the training time by moving the smallest client from the slowest to the fastest worker. Thus, we assume that "*timedelta workers*" approximates the time wasted due to clients' misplacement.

The impact of the placement strategy on the training time depends on the GPUs we are dealing with. We can distinguish between *homogeneous settings*, in which each worker executes on the same GPU type at the same speed, and *heterogeneous settings*, in which groups of workers are allocated on different GPU types having different speeds.

In *homogeneous settings*, we expect to observe that the ratio between the number of clients per round and the number of workers drives the difference in performance. As this ratio increases, we expect to observe similar performance across strategies. This expectation is motivated by the fact that the noise introduced by the random sampling of clients will likely balance the load over workers naturally. When this ratio is close to 1, only sampled cohorts of clients whose number of large-size clients is different from the number of workers can lead to different performance. In this case, the performance gap appears because big clients cannot be evenly distributed across workers. However, we expect the BU strategy to slightly outperform the others since it can balance the load over workers with the minimum overhead.

The scenario in which different strategies will significantly differ in performance is the *heterogeneous setting* in which clients of the same size are not trained in the same amount of time by different workers. For these settings, the LB strategy will outperform BU because of its ability to discriminate between workers executing on different GPU types. The difference in performance will mostly depend on how heterogeneous the hardware settings are. In particular, the gap between different hardware in training small clients, prevalent in FL datasets, will have the most critical impact.

The LB placement strategy strongly relies on the performance of the fitting procedure over clients' training time from previous rounds. It is worth discussing the two main ingredients upon which this strategy arranges the clients' placement: the data collected for previous rounds and the fitting function. We chose to keep all the data from previously trained rounds. In general, the robustness of curve fitting is proportional to the number of data points; conversely, its duration is proportional to the number of data points. We observe that during the last round, when the data points are 9900, the LB strategy takes an amount of time on the same order of magnitude as the RR strategy, tens of seconds. For extended experiments where the number of data points to fit could be much more significant, it is reasonable to define a time window for deleting older data. Second, our fitting function has been chosen for its mathematical properties. The logarithm combined with a linear term ensures that the fitted function never predicts negative values despite the wide cloud of data points produced by small clients, as can be observed in Fig. 5. Since small clients have greater training time variance, many may take longer than their slightly bigger counterparts. This behaviour may enforce negative slopes in the fitted curve, especially if polynomial. The logarithmic term makes the function more robust to this situation, while the linear term ensures that the bigger clients are predicted

to take longer to train. *As such, the chosen function allows us to avoid ever predicting a negative time for a client at the cost of overestimating the duration of small clients.*

## 6 EXPERIMENTAL DESIGN

This section describes a series of experiments meant to showcase our method's superiority and validate it.

### 6.1 Federated Learning Tasks

We use three representative FL tasks throughout this work to showcase our method. These are characterised by having clients with long tail distributions over the number of samples and workloads. For all tasks, we exclude clients with less than one batch of training data.

**Image Classification:** The goal of this task is to collaboratively train a ShuffleNetV2 [20] network to correctly classify images amongst 596 classes. For this, we use a federated version of the original OpenImage [14] dataset as implemented by FedScale. This dataset contains $1.6 \times 10^6$ images partitioned across 13 771 clients. We use a batch size of 20 samples.

**Speech Recognition:** In this task, we use the Google Speech Commands dataset [28] to collaboratively train a ReseNet-34 [11] to classify audio samples amongst a set of 35 predefined spoken words. The dataset contains a collection of $157K$ one-second-long clips naturally partitioned according to their 2168 speakers. We use a batch size of 20 samples.

**Text Generation:** We use the Shakespeare dataset, as implemented in TensorFlow Federated [9], to train a two-cell LSTM-based language model as defined in [4] The dataset comprises sentences extracted from *The Complete Works of William Shakespeare* and grouped into 648 fictional characters. We follow the LEAF experimental configuration and use a batch size of 4 samples.

### 6.2 Hardware Configuration

We consider two hardware configurations that reflect common research centres, namely *single-node* and *multi-node*. We also further distinguish between simulations using *homogeneous* and *heterogeneous* GPUs. As previously indicated, all the FedScale, Google Speech, and Shakespeare experiments use a fixed batch size. This makes differences in worker VRAM attributable only to model size and input data shape.

**Single-node:** Our single-node experiments are all run on *node 0* containing Nvidia A40 GPUs and an Intel (R) Xeon (R) Gold 6152 with 88 cores. For OpenImage, the A40s are filled with 13 workers each, given the size of ShuffleNetV2 [20] and the input size. For Google Speech, we use only 4 workers per GPU due to data loading requirements, while for Shakespeare, we use 10 workers total to match the number of clients. Each A40 is paired with 11 CPU cores out of the 88 mentioned above. We run all our experiments with *Two homogeneous GPUs* on this node using two A40s with 22 CPU cores available.

**Multi-node:** Our multi-node experiments are run on a combination of the aforementioned *node 0* containing A40s and *node 1*. *Node 1* contains Nvidia RTX 2080 Ti GPUs and an Intel(R) Xeon(R) CPU E5-2680 v4 containing 56 cores. For OpenImage, we use 4 workers per 2080 due to VRAM constraints. The worker setup for Google Speech and Shakespeare is the same as on *node 0*. Each 2080 is paired with 8 CPU cores out of the 56 mentioned above. Our experiments for heterogeneous hardware shall use one A40 from *node 0* paired with 1-4 Nvidia 2080s. We only use more than one 2080 in scenarios where we want to observe how balancing the number of workers across GPU types affects performance. After the workers have completed their partial aggregation on the *CPU* of their respective node, all final aggregation steps happen on *node 0* on the *CPU*.

### 6.3 Framework Benchmark

The fundamental contribution of the design of our system is to overcome the pull-based queuing system to allow for fast FL simulation. In this experiment, we aim to assess the speed of our solution, whatever the placement strategies adopted by the client allocator. We compare to the other frameworks by measuring the throughput of the simulation since this describes the system's speed as the number of clients the system can train per second. It is calculated by dividing each round completion time by the number of clients trained. We perform the three tasks under different homogeneous hardware settings having 1, 2 or 4 homogeneous GPUs. For homogeneous hardware, different placement strategies perform similarly. Thus, the performance is mainly impacted by the system design.

### 6.4 Policies Benchmark

This work proposes different placement strategies to use in the simulation. In this experiment, we aim to answer the question of how the choice of strategy impacts the performance of the simulation. We measure the throughput of the FL simulation and compare different strategies against each other. We are also interested in identifying how effective the strategies are in balancing the load between the workers. To this aim, we measure the time difference between the first and last workers to finish processing their clients and compare the strategies against each other.

The experiment is designed to be executed in the most challenging hardware setting, the heterogeneous one with two nodes having different GPU types. First, we train all the tasks in the setting with one GPU per type for two GPUs. We keep the number of clients per round to 100, except for Shakespeare, where we set 10 clients for 100 rounds. Here, we compare the throughput of the experiments using different strategies against the fastest other framework. Second, we train the Image Classification task in different heterogeneous hardware settings, increasing the number of GPUs belonging to the type that can host the lower number of workers. This second set of experiments uses the two previous nodes, one always having one GPU and the other having 1, 2, 3, and then 4 GPUs.

These settings reflect a typical situation a researcher could face: the available GPUs with more VRAM are few, but more GPUs with less VRAM are available.

### 6.5 System Scalability

Our main contribution in this paper is to allow efficient large-scale simulations at the scale of real-world applications. In this experiment, we aim to measure how our proposed method compares to other Placement Policies as we *increase the average number of clients per GPU*. The first set of experiments considers the homogeneous scenario of four GPUs of the same type on the same node. We fix the number of workers because the hardware is fixed, and we progressively increase the number of clients per round while keeping constant the overall number of clients trained. In order to maintain the total number of trained clients to 10 000, as it is in other experiments in this work, we choose the values for the number of clients per round to be 100, 200, 400, 625, 1000 training respectively for 100, 50, 25, 16, 10 rounds. The workers receive longer lists of clients to train each round, while the total work is always the same. We measure the system's throughput for all the proposed client policies and then compare them against the fastest framework.

A second set of experiments considers the most constrained and heterogeneous scenario we had: two nodes, each of them having on GPU of a different type. Similarly, we played with the number of clients each worker has to train in each round while keeping the total amount of work constant. For each round, we measure the time difference between the first and last worker to finish processing their clients and the throughput.

We run the above experiments on the Image Classification task with the largest client population. This setting reflects a common FL scenario where researchers want to investigate the benefits of sampling larger fleets of devices in one round while still constrained by the available hardware.

## 7 EVALUATION

In this section, we describe the results of the experiments proposed in Section 6. We begin presenting the comparison of the performance of *Pollen* against other frameworks. We continue focusing on the differences between the strategy proposed. Furthermore, we extend the investigation of the strategies by evaluating them at scale. Finally, a discussion about the proposed client placement's limitations is presented.

### 7.1 Comparison between frameworks

*Pollen* proves to deliver faster FL simulations compared to Flower (version 1.1) [2] , Flute (commit@0e8762b) [7], and FedScale (commit@9bfc029a3c) [16] in every setting we have tested. This consistency demonstrates the superiority of a push-based allocation of clients across workers. The results of the throughput of the simulation in a more accessible homogeneous setting put *Pollen* on top of the classification since it outperforms all the other frameworks. The example in Fig. 6 shows that, when using two GPUs of the same type, namely Nvidia A40, on a single node, *Pollen* has the highest throughput for all the tasks. Compared to the fastest framework, the speed-up obtained is 3x on the Text Generation task, about 2x on the Image Classification task, and 5x on the Speech Recognition task. It is noted that the performance of all the tasks is varied across the other frameworks except for the Speech Recognition one, in which they perform similarly. We highlight that in every comparison in which the number of clients per round is fixed, the absolute time of the experiment is inversely proportional to the reported throughput. In addition, since the Image Classification task has the lowest speed-up factor, we assume that to be the task using which further comparisons will be fairer.

**Implications:** *Pollen* outperforms all previous frameworks and can provide a great boost to the scalability of FL simulations thus allowing better FL methods to be developed for production systems.

### 7.2 The Impact of Client Placement

Given that our design outperforms other frameworks in homogeneous settings, we evaluate the impact the client placement strategies in heterogeneous settings. Two nodes were available in the setting, with one Nvidia A40 and one Nvidia RTX 2080 Ti, respectively. The results regarding the throughput are shown in Table 1. We can assume that different strategies deliver the same performance for the Text Generation task because distributing 10 clients across 10 workers is trivial. The Image Classification task shows the biggest variation across strategies, where the LB strategy of *Pollen* outperforms the others. We note that the RR strategy

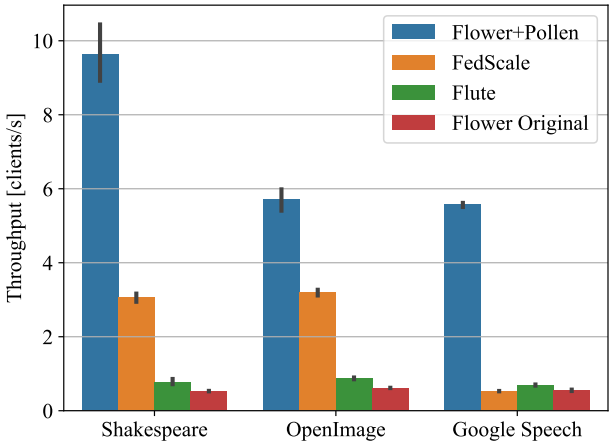



**Figure 6: The throughput (the greater, the better) of our client placement systems implemented in Flower compared with FedScale, Flute and standard Flower. Using resource-aware client placement, *Pollen* outperforms the fastest other framework by a factor 3x-5x, depending on the task. The values reported for our contribution have been chosen for the best-performing strategy that tends to be LB.**

of *Pollen* underperforms other strategies when the placement is not trivial, presenting the most significant gap in the Image Classification task. We argue that the peculiarity of the Speech Recognition dataset causes different strategies to have more minor variations in performance. Furthermore, we highlight the general superiority of our method against the fastest framework in this challenging heterogeneous scenario.

We gradually increased the number of available GPUs for the second set of experiments to evaluate the impact of different placement strategies. In this experiment, we used the different strategy of *Pollen*. We significantly increased the number of the GPU type that can host the lower amount of workers, namely the Nvidia RTX 2080 Ti. As such, four settings are compared in this multi-node scenario: $(1, 1)$, $(1, 2)$, $(1, 3)$, $(1, 4)$, where the first number refers to the available Nvidia A40s in the first node and the second to the available Nvidia RTX 2080 Ti in the second node. The plot in Fig. 7 shows the throughput of different strategies over the Image Classification task. It is worth noting that the number of clients per round has been kept constant during this particular experiment. In this way, what is changing is the *density* of clients across workers, as having more GPUs available means having more workers. The results demonstrate that the gain produced by using the LB strategy degrades as the density of clients across workers decreases. This degradation is reflected in both the metrics taken into consideration, namely the difference in training time between the fastest and slowest workers (Table 2) and the throughput (Fig. 7).

**Implications:** Intelligent client placement is highly beneficial for hardware configurations containing multiple GPU types. Given the difficulty of profiling and configuring a large-scale cluster with heterogeneous GPUs, the automatic nature of workload distribution in *Pollen* provides a great advantage.

**Table 1: The throughput (the greater, the better) measured in clients per second for three tasks. The proposed *Pollen*'s strategies are compared against FedScale in the most heterogeneous setting.**

| Datasets | FedScale | LB | BU | SRR | RR |
|---|---|---|---|---|---|
| Google Speech | 3.6±0.5 | 4.7±0.4 | **5.4±0.2** | 5.0±0.2 | 5.2±0.3 |
| OpenImage | 3.6±0.5 | **6±1** | 4.7±0.5 | 5.2±0.8 | 4.1±0.9 |
| Shakespeare | 2.3±0.8 | 10±3 | 10±4 | 10±4 | **11±4** |

## 7.3 Scalability of Placement Strategies

Having already established the effectiveness of *Pollen* and analysed the difference between different client placement strategies, we are now concerned with the proposed method's scalability. We begin by comparing with FedScale in a homogeneous setting while increasing the *density* of clients across workers. In this experiment, instead of increasing the number of available GPUs we increase the number of clients per round while keeping the total number of clients at 10 000. This represents a common setting for large-scale

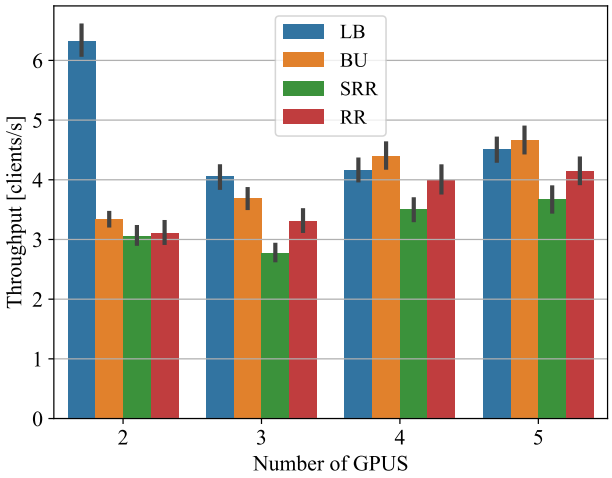



**Figure 7: The throughput (the greater, the better) measured in clients per second of Round-Robin (RR), Sorted Round Robin (SRR), Batch-uniform (BU), and Learning-based (LB) client placement policies for different multi-node heterogeneous GPU configurations. The Image Classification task has been used.**

**Table 2: Here the difference in training time between the fastest and slowest workers (the lower the better) measured in seconds for Round-Robin (RR), Sorted Round Robin (SRR), Batch-uniform (BU), and Learning-based (LB) client placement policies using different multi-node heterogeneous GPU configurations. The Image Classification task has been used.**

| # GPUs in nodes (0, 1) | BU | LB | RR | SRR |
|---|---|---|---|---|
| (1, 1) | 23±3 | **10±4** | 30±7 | 26±5 |
| (1, 2) | 21±4 | **18±4** | 28±6 | 31±6 |
| (1, 3) | **17±4** | 18±4 | 23±6 | 23±5 |
| (1, 4) | **15±3** | 16±4 | 21±5 | 21±6 |

experiments in which the number of clients per round can be very high. Such setting is often explored by theoretical works and it challenge in a simulation context. As Fig. 8 shows, our push-based placement strategies increase in throughput as the number of clients per round increases. However, the throughput of FedScale does not scale as the number of clients per round increases. This is to be expected since as the number of clients per round increases, *Pollen* does fewer and fewer total communication steps. On the other hand, FedScale does the same number of communication steps. We can also note that the exact placement strategy does not matter, even as the number of clients per round is scaled up, supporting our previous section results.

In the case of heterogeneous GPUs, our results shown in Table 3 indicates that the learning-based solution keeps its advantage over all others even as the number of clients increases. This is driven by the learning-based policy of properly distributing works across GPUs, even when many clients are involved in a round. The other placement strategies have a highly inconsistent ordering as they operate under the false assumption that workers should be treated equally. It is also worth noting that all the improvements in throughput we obtain are based on minimising the wait for the slowest worker at the end of a round. As such, while all placement strategies get an initial boost from increasing client density, once sufficient clients are available each round to keep all workers filled for most of the round, throughput stops increasing.

We now turn our attention to Table 4, which shows the difference between the fastest and slowest worker; we can observe the cause of this trend. When not accounting for hardware heterogeneity, the other policies reflect the speed difference between the GPUs the workers are placed on. It is clear that the learning-based solution minimises this gap relative to the other placement strategies.

**Implications:** The benefits of client placement for heterogeneous GPU configurations persist as the number of clients processed in a round grows. Importantly, these benefits to be highly consistent after a sufficient number of clients per round is reached.

## 8 LIMITATIONS

While the results for the push-based client placement system we have presented are compelling and consistent, it does present several limitations, which we list below:

- For homogeneous settings, the improvements are brought by the push-based design, as client placement decisions do not generally matter when using random sampling.
- For heterogeneous settings, a very low number of clients relative to the number of workers does not allow for sufficient placement decisions. Thus, it is difficult for the learning-based method to balance load across workers.
- For vast numbers of clients per round, all placement methods approach their maximum throughput for the task and cease providing further improvements.
- The learning-based policy may not provide the theoretically optimal placement for two reasons. First, even when given a perfect prediction of training time for all clients by an oracle, the distribution of these clients over workers is still non-trivial. Second, the error in estimating each client may cause suboptimal placement decisions regardless of the number of available clients or placement strategy. As we observe from Table 4, the gap between the fastest and slowest worker is always proportional to the number of clients in a fixed manner. This indicates a consistent estimation error.
- The partial aggregation algorithm required to minimise communication is incompatible with some federated learning aggregation algorithms by default. For example, we do not currently support the adaptive optimisation proposed by Reddi et al. [24]

**Table 3: The throughput (the greater, the better) for the Image Classification task using *Pollen*'s RR, SRR, BU, and LB client placement strategies with the most heterogeneous multi-node hardware configurations. The number of clients per round has been changed while keeping the total number of trained clients constant.**

| Num. Clients per round | BU | LB | RR | SRR |
|---|---|---|---|---|
| 100 | 3.3±0.4 | **6±1** | 3.1±0.6 | 3.1±0.5 |
| 200 | 6.7±0.9 | **8±1** | 7±1 | 6±1 |
| 400 | 7.1±0.5 | **7.9±0.7** | 6.9±0.8 | 7.6±0.5 |
| 625 | 7.1±0.4 | **8.4±0.7** | 7.2±0.6 | 6.9±0.4 |
| 1000 | 7.0±0.4 | **8.3±0.9** | 7.2±0.5 | 7.4±0.3 |

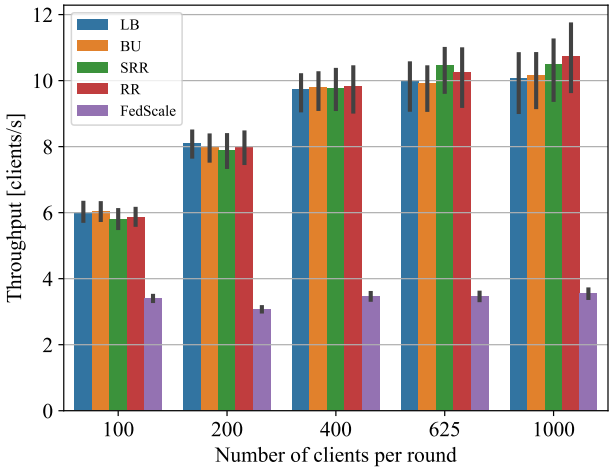



**Figure 8: Throughput of the simulation for different placement strategies compared against the fastest framework in performing the Image Classification task. The total number of trained clients has been kept constant to 10000, while the number of clients per round has been increased. The hardware setting used was homogeneous with 4 Nvidia A40.**

**Table 4: Here the difference in training time between the fastest and slowest workers (the lower the better) for Round-Robin (RR), Sorted Round Robin (SRR), Batch-uniform (BU), and Learning-based (LB) client placement policies using the most heterogeneous multi-node hardware configurations. The number of clients per round has been changed while keeping the total number of trained clients constant. The Image Classification task has been used.**

| Num. Clients per round | BU | LB | RR | SRR |
|---|---|---|---|---|
| 100 | 23±3 | **10±4** | 30±7 | 26±5 |
| 200 | 24±3 | **19±4** | 30±6 | 27±5 |
| 400 | 48±3 | **41±5** | 56±6 | 43±3 |
| 625 | 77±4 | **62±10** | 83±8 | 78±5 |
| 1000 | 125±7 | **106±18** | 134±11 | 118±6 |

## 9 CONCLUSION

In this work we have shown that the current *pull-based* approaches for client placement available in Federated Learning frameworks are incapable of exploiting both client and hardware heterogeneity. Based on this fact, we have proposed a resource-aware client placement algorithm which minimises communication costs between the server and workers responsible with training the clients. An extensive experimental evaluation has shown our proposed changes to bring improvements of 50% to 400%. This significant improvement allows for the quick development of algorithms with downstream applications in training fleets of millions of edge-devices. Since our modifications have a relatively small footprint, we recommend that all active FL frameworks adopt a similar design for their simulation engines.

## REFERENCES


[1] Ravikumar Balakrishnan, Tian Li, Tianyi Zhou, Nageen Himayat, Virginia Smith, and Jeff A. Bilmes. 2022. Diverse Client Selection for Federated Learning via Submodular Maximization. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net. https://openreview.net/forum?id=nwKXyFvaUm

[2] Daniel J. Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Titouan Parcollet, and Nicholas D. Lane. 2020. Flower: A Friendly Federated Learning Research Framework. *CoRR* abs/2007.14390 (2020). arXiv:2007.14390 https://arxiv.org/abs/2007.14390

[3] Kallista A. Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloé Kiddon, Jakub Konečný, Stefano Mazzocchi, Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. 2019. Towards Federated Learning at Scale: System Design. In *Proceedings of Machine Learning and Systems 2019, MLSys 2019, Stanford, CA, USA, March 31 - April 2, 2019*, Ameet Talwalkar, Virginia Smith, and Matei Zaharia (Eds.). mlsys.org. https://proceedings.mlsys.org/book/271.pdf

[4] Sebastian Caldas, Peter Wu, Tian Li, Jakub Konečný, H. Brendan McMahan, Virginia Smith, and Ameet Talwalkar. 2018. LEAF: A Benchmark for Federated Settings. *CoRR* abs/1812.01097 (2018). arXiv:1812.01097 http://arxiv.org/abs/1812.01097

[5] Zachary Charles, Zachary Garrett, Zhouyuan Huo, Sergei Shmulyian, and Virginia Smith. 2021. On Large-Cohort Training for Federated Learning. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (Eds.). 20461–20475. https://proceedings.neurips.cc/paper/2021/hash/ab9ebd57177b5106ad7879f0896685d4-Abstract.html

[6] Yae Jee Cho, Jianyu Wang, and Gauri Joshi. 2020. Client Selection in Federated Learning: Convergence Analysis and Power-of-Choice Selection Strategies. *CoRR* abs/2010.01243 (2020). arXiv:2010.01243 https://arxiv.org/abs/2010.01243

[7] Dimitrios Dimitriadis, Mirian Hipolito Garcia, Daniel Madrigal, Andre Manoel, and Robert Sim. 2022. FLUTE: A Scalable, Extensible Framework for High-Performance Federated Learning Simulations. https://www.microsoft.com/en-us/research/publication/flute-a-scalable-extensible-framework-for-high-performance-federated-learning-simulations/

[8] Wei Gao, Qinghao Hu, Zhisheng Ye, Peng Sun, Xiaolin Wang, Yingwei Luo, Tianwei Zhang, and Yonggang Wen. 2022. Deep Learning Workload Scheduling in GPU Datacenters: Taxonomy, Challenges and Vision. *CoRR* abs/2205.11913 (2022). https://doi.org/10.48550/arXiv.2205.11913 arXiv:2205.11913

[9] Google. 2019. Tensorflow Federated. https://www.tensorflow.org/federated

[10] Song Han, Huizi Mao, and William J. Dally. 2016. Deep Compression: Compressing Deep Neural Network with Pruning, Trained Quantization and Huffman Coding. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). http://arxiv.org/abs/1510.00149

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, 770–778. https://doi.org/10.1109/CVPR.2016.90

[12] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista A. 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, Zaïd Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konečný, 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. *Found. Trends Mach. Learn.* 14, 1-2 (2021), 1–210. https://doi.org/10.1561/2200000083

[13] Armand K. Koupai, Mohammud Junaid Bocus, Raúl Santos-Rodríguez, Robert J. Piechocki, and Ryan McConville. 2022. Self-Supervised Multimodal Fusion Transformer for Passive Activity Recognition. *CoRR* abs/2209.03765 (2022). https://doi.org/10.48550/arXiv.2209.03765 arXiv:2209.03765

[14] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper R. R. Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, Tom Duerig, and Vittorio Ferrari. 2020. The Open Images Dataset V4. *Int. J. Comput. Vis.* 128, 7 (2020), 1956–1981. https://doi.org/10.1007/s11263-020-01316-z

[15] HyeokHyen Kwon, Catherine Tong, Harish Haresamudram, Yan Gao, Gregory D. Abowd, Nicholas D. Lane, and Thomas Plötz. 2020. IMUTube: Automatic Extraction of Virtual on-body Accelerometry from Video for Human Activity Recognition. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 3 (2020), 87:1–87:29. https://doi.org/10.1145/3411841

[16] Fan Lai, Yinwei Dai, Sanjay Sri Vallabh Singapuram, Jiachen Liu, Xiangfeng Zhu, Harsha V. Madhyastha, and Mosharaf Chowdhury. 2022. FedScale: Benchmarking Model and System Performance of Federated Learning at Scale. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA (Proceedings of Machine Learning Research, Vol. 162)*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (Eds.). PMLR, 11814–11827. https://proceedings.mlr.press/v162/lai22a.html

[17] Shen Li, Yanli Zhao, Rohan Varma, Omkar Salpekar, Pieter Noordhuis, Teng Li, Adam Paszke, Jeff Smith, Brian Vaughan, Pritam Damania, and Soumith Chintala. 2020. PyTorch Distributed: Experiences on Accelerating Data Parallel Training. *Proc. VLDB Endow.* 13, 12 (2020), 3005–3018. https://doi.org/10.14778/3415478.3415530

[18] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. 2020. Federated Learning: Challenges, Methods, and Future Directions. *IEEE Signal Process. Mag.* 37, 3 (2020), 50–60. https://doi.org/10.1109/MSP.2020.2975749

[19] Ji Lin, Ligeng Zhu, Wei-Ming Chen, Wei-Chen Wang, Chuang Gan, and song han. 2022. On-Device Training Under 256KB Memory. In *Advances in Neural Information Processing Systems*, Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (Eds.). https://openreview.net/forum?id=zGvRdBW06F5

[20] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. 2018. ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XIV (Lecture Notes in Computer Science, Vol. 11218)*, Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss (Eds.). Springer, 122–138. https://doi.org/10.1007/978-3-030-01264-9_8

[21] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA (Proceedings of Machine Learning Research, Vol. 54)*, Aarti Singh and Xiaojin (Jerry) Zhu (Eds.). PMLR, 1273–1282. http://proceedings.mlr.press/v54/mcmahan17a.html

[22] Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I. Jordan, and Ion Stoica. 2018. Ray: A Distributed Framework for Emerging AI Applications. In *13th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2018, Carlsbad, CA, USA, October 8-10, 2018*, Andrea C. Arpaci-Dusseau and Geoff Voelker (Eds.). USENIX Association, 561–577. https://www.usenix.org/conference/osdi18/presentation/nishihara

[23] Takayuki Nishio and Ryo Yonetani. 2018. Client Selection for Federated Learning with Heterogeneous Resources in Mobile Edge. *CoRR* abs/1804.08333 (2018). arXiv:1804.08333 http://arxiv.org/abs/1804.08333

[24] Sashank J. Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and Hugh Brendan McMahan. 2021. Adaptive Federated Optimization. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. https://openreview.net/forum?id=LkFG3lB13U5

[25] Konstantin Sozinov, Vladimir Vlassov, and Sarunas Girdzijauskas. 2018. Human Activity Recognition Using Federated Learning. In *IEEE International Conference on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications, ISPA/IUCC/BDCloud/SocialCom/SustainCom 2018, Melbourne, Australia, December 11-13, 2018*, Jinjun Chen and Laurence T. Yang (Eds.). IEEE, 1103–1111. https://doi.org/10.1109/BDCloud.2018.00164

[26] Catherine Tong, Shyam A. Tailor, and Nicholas D. Lane. 2020. Are Accelerometers for Activity Recognition a Dead-end?. In *HotMobile '20: The 21st International Workshop on Mobile Computing Systems and Applications, Austin, TX, USA, March 3-4, 2020*, Padmanabhan Pillai and Qin Lv (Eds.). ACM, 39–44. https://doi.org/10.1145/3376897.3377867

[27] Ewen Wang, Ajay Kannan, Yuefeng Liang, Boyi Chen, and Mosharaf Chowdhury. 2023. FLINT: A Platform for Federated Learning Integration. *CoRR* abs/2302.12862 (2023). https://doi.org/10.48550/arXiv.2302.12862 arXiv:2302.12862

[28] Pete Warden. 2018. Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition. *CoRR* abs/1804.03209 (2018). arXiv:1804.03209 http://arxiv.org/abs/1804.03209

[29] Wencong Xiao, Romil Bhardwaj, Ramachandran Ramjee, Muthian Sivathanu, Nipun Kwatra, Zhenhua Han, Pratyush Patel, Xuan Peng, Hanyu Zhao, Quanlu Zhang, Fan Yang, and Lidong Zhou. 2018. Gandiva: Introspective Cluster Scheduling for Deep Learning. In *13th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2018, Carlsbad, CA, USA, October 8-10, 2018*, Andrea C. Arpaci-Dusseau and Geoff Voelker (Eds.). USENIX Association, 595–610. https://www.usenix.org/conference/osdi18/presentation/xiao

[30] Han Zhao, Weihao Cui, Quan Chen, Jingwen Leng, Kai Yu, Deze Zeng, Chao Li, and Minyi Guo. 2020. CODA: Improving Resource Utilization by Slimming and Co-locating DNN and CPU Jobs. In *40th*

*IEEE International Conference on Distributed Computing Systems, ICDCS 2020, Singapore, November 29 - December 1, 2020*. IEEE, 853–863. https://doi.org/10.1109/ICDCS47774.2020.00069