A2FL: Availability-Aware Federated Learning in Heterogeneous Environments

Ahmed M. Abdelmoniem
Queen Mary University of London
United Kingdom
ahmed.sayed@qmul.ac.uk

Abstract

Distributed privacy-preserving machine learning (ML) methods are increasingly becoming the norm for training ML models on edge devices. In a popular approach known as Federated learning (FL), service providers leverage end-user data to train ML models to improve services such as text autocompletion, virtual keyboards, and item recommendations. FL is expected to grow in importance with the increasing focus on privacy and 5G/6G technologies. FL faces major challenges such as resource and user heterogeneity, communication overheads, and efficient privacy preservation. In practice, training global models via FL is very time-consuming because of the heterogeneity of clients' computation and communication speeds. Even worse, clients may not always be available to participate in training which limits their representation in the global model. Empirical analysis shows that client availability impacts the model quality which motivates the design of *A2FL*. *A2FL* is a client selection method which mitigates the quality degradation caused by the misrepresentation of the client population. Our results show that, compared to existing methods, *A2FL* can enhance clients' representation and improve the trained model quality.

Keywords: Federated Learning, Heterogeneity, Selection

1 Introduction

Recent interest in Edge AI (i.e., pushing intelligence towards the edge) is mainly driven by the need to push computation toward data sources in an effort to enhance privacy and security [12, 30]. These efforts led to the formation of the Federated Learning (FL) paradigm which transformed traditional distributed machine learning (ML) training methods. Many service providers such as Google, Facebook, and Apple use FL to train global models for natural language processing (NLP) and computer vision (CV) tasks to server applications such as virtual keyboards, object detection, image classification, and recommendation systems [20, 21, 24, 26, 53, 54, 64]. FL is commonly used with distributed medical imaging data [37]; smart camera images [29], and live network traffic data [61]. In FL, the central server managing the models ships them to the end-device learners who are responsible for performing the training locally to preserve the privacy and security of user data. Due to the lack of control over client devices, FL environments are highly heterogeneous which presents a variety of challenges. In FL, the process is participatory and relies on the availability of the learners and their data. These learners produce and store the application data used to locally train the central ML model and contribute their model updates to the central server for incorporation into the global model.

Time-to-accuracy is a vital performance measure for training quality and is the focus of much work in this area [30, 32, 35, 58, 62]Generally, the objective is to reduce the time-to-accuracy by either improving the statistical efficiency and/or reducing the training time. Statistical efficiency depends on the number of participating learners and their data as well as learning-specific hyper-parameters such as minibatch size, learning rate, and the number of local training epochs. The hyper-parameters are factors that require gradual tuning for different FL jobs. Additionally, handling heterogeneous training data is generally more challenging and therefore many efforts are dedicated to addressing the data heterogeneity problem [16, 23, 34, 43, 50, 56, 63].

Another factor that can contribute to low training quality is the selection method used to pick a subset of learners from a large population of learners affecting the data sample distribution. During the selection stage, the server samples from the available learners to participate in training the global model on their local datasets in this round. In heterogeneous environments, clients tend to be battery-powered mobile devices (e.g., smartphones, smart wear, or IoT devices) and so the availability for participation is typically dependent on one or more factors such as the charging state of the device, whether the device is connected to power, WIFI, and/or idle. Client selection has a predominant role in affecting the quality of the trained model [10, 32, 38]. Most methods focused on the heterogeneity of client devices and data. For instance, FedCS [45] favours fast clients over slow ones; InclusiveFL [38] ships models of different sizes to accommodate various clients' computational capabilities; DivFL [10] tries to mitigate the data heterogeneity by selecting a sample of clients that potentially approximates the majority of data distributions. These solutions do not take into account the dynamics of clients' availability who possess the data samples. Therefore, the term behavioural heterogeneity has been coined to represent the availability dynamics of the learners [6, 30, 62]. Specifically, learners exhibit variable availability patterns at different times during the training

rounds which makes learning tasks more challenging on heterogeneous (or non-IID) data distributions [6, 12, 30, 33, 62]. Therefore, any practical FL framework should take into account the behavioural heterogeneity of the learners to boost the quality of the trained model.

In this work, we aim to focus on addressing the impact of behavioural heterogeneity in FL training. We focus on dissecting the impact on time-to-accuracy caused by training models on non-diverse sets of data samples due to some learners (with unique data samples) being unavailable for participation during the selection stage. We address behavioural heterogeneity to improve the system's robustness to realistic (or dynamic) data distributions among learners. To this end, we introduce *A2FL*, a practical participant selection method which accounts for clients' availability and maximizes clients' representation during FL training. *A2FL* intelligently selects the least available participants in the future with higher priority ensuring their data distributions are seen by the trained model. *A2FL* is compatible with and is a plug-in component for existing practical FL systems [12–14].

Our contributions are as follows:

1. We demonstrate that learners' limited availability in FL can significantly affect the trained model quality and time-to-accuracy.
2. We propose *A2FL* to maximize the majority of learners' representation in the training process.
3. We evaluate *A2FL* using a common FL benchmark and show its benefits.

2 Related Work

Federated Learning (FL): is a new distributed machine learning method that is becoming increasingly popular for its privacy-preserving and low-communication features. This motivated the growing adoption of FL to improve the end-user experience (e.g., the search suggestion quality of virtual keyboards [64]). In FL, training a global model is assigned to a sub-population of decentralized devices such as mobile or IoT devices. These devices possess private data and engage in training the model on their data [12, 41].

Participant Selection Strategies: In each round, the server selects among a subset of the available learners to train the global model. Several works designed better selection strategies other than Random selection. For instance, [45] select learners with fast hardware and network speed. [17, 19, 48] select learners whose updates result in higher loss values (i.e., statistical gain). Oort [32] combines both goals (i.e., system and statistical efficiency). These approaches result in models trained on a limited subset of the large population compromising on an inclusive representation of the clients.

System heterogeneity: One of the major contributors to system performance unpredictability is the heterogeneity inherent in many distributed systems. Mainly, in the FL context, the heterogeneity of devices' system configurations (e.g., computation, communication, battery, etc) results in unpredictable performance. For instance, the stragglers (i.e., slow workers) can halt the training process for a prolonged duration [2, 8]. Several solutions exist that address this problem through system and algorithmic solutions [2, 8, 32, 35]. Moreover, in FL, the heterogeneity is also a byproduct of other artifacts other than the devices. For example, the learner data distributions, the participants' selection method, and the behaviour of the device's owner are common sources of heterogeneity in FL setting [6, 7, 12].

Energy-conservation: Considering the uncertainties in the mobile environment, recently energy-aware federated learning techniques have been proposed [9, 30]. These works aim for energy conservation to mitigate client dropouts which is the major contributor to the degraded FL model qualities [9].

Improvements in FL: In FL, several works aim to improve the time-to-accuracy of training by leveraging techniques such as periodic updates, adaptive compression, and asynchronous updates [2, 8, 9, 12].Several methods that have been shown to achieve good performance in distributed ML settings could be applied to the FL settings such as [1, 3, 4, 22, 49, 60] Other work studied the privacy guarantees of FL settings [42, 44]. Moreover, the bias in FL is studied to ensure fair participation in the training process [5, 8].

In this work, we address a unique problem impacting the quality of the FL-trained models due to the non-inclusive selection mechanisms. To address this, there we treat learners as a diverse set of data sources and leverage their availability for improved diversity.

3 Background

We first introduce the FL training procedure while focusing on system design aspects. Then, we highlight the major challenges in existing FL systems based on empirical evidence from real FL benchmarks. Then, we motivate our work by highlighting the main drawbacks of existing solutions.

3.1 Federated Learning

In this work, we build upon the common FedAvg aggregation method [12, 41] where the training process requires a (logically) centralized FL server and a large set of decentralized devices, (e.g., sensors, smartphones, and/or IoT devices). These devices possess private training data in their local storage and are called learners. For privacy and security reasons, data are not shared with the FL server or other learners. The FL server manages the training process and invokes learners to start a collaborative task of training a joint global model on their distributed datasets.

We show the FL stages for training a common model in Fig. 1. At the start of each round, the FL server initiates the selection phase and waits for a sufficient number of learners to become available for training.[1] The server chooses a sample from

[1]A learner is available if it is: charged, idle, and/or on WIFI [12].

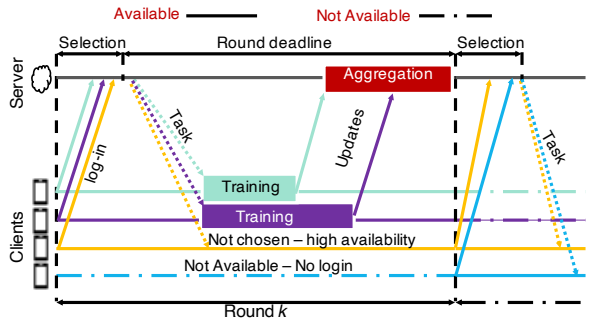


Figure 1. A FL training round. Clients are sampled to run an FL training task and submit their updates in a given round.

a large number of online learners – called participants – to participate in updating the global model in the current round. It sends each participant the task which consists of the current version of the global model and any task-specific settings (e.g., hyper-parameters).

After receiving the task, each participant runs a local optimization process to optimize the model over local data for several epochs. The updated model is sent to the aggregation server during the reporting stage. The FL server either waits until a deadline expires or when the target number of updates is received. The server then aggregates the updates and checkpoints a new version of the model. This completes the round and several rounds are repeated until the training objective is fulfilled (e.g., the target accuracy is achieved or the training cost exceeds a threshold).

The main distinction between conventional distributed data-parallel training and federated learning is that the learners are independent and not under a single entity's management. Therefore the following types of heterogeneity are common in FL: 1) ***data heterogeneity:*** the participants may have varying data samples that are different in size, number of classes, and/or distribution; 2) ***device heterogeneity:*** the participants may use devices of different computational and communication capabilities owing to variable hardware and network settings; 3) ***behavioural heterogeneity:*** the participants' availability for training changes over training rounds which is mainly driven by users' device usage patterns.

Several works have tried to address the challenges posed by different types of heterogeneity in FL settings. Though, most of these works focus on tackling data or device heterogeneity [18, 32, 35, 36, 58]. Their main objective is to improve model quality and/or training speed to boost time-to-accuracy. In this work, we focus on behavioural heterogeneity which creates unique challenges for FL systems. This is because this type of heterogeneity is harder to control and can have a large influence on the trained model quality.

To show the impact of behavioural heterogeneity, we experiment with two cases: i) A uniform case (**U**): where devices have homogeneous device configurations (i.e., all learners can submit updates in time) and learners remain available

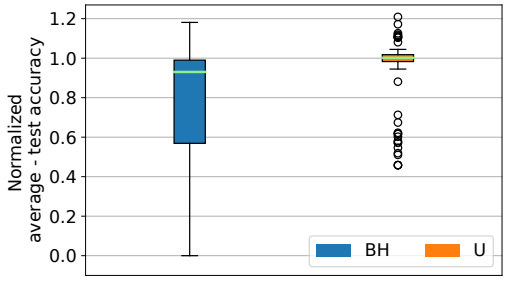


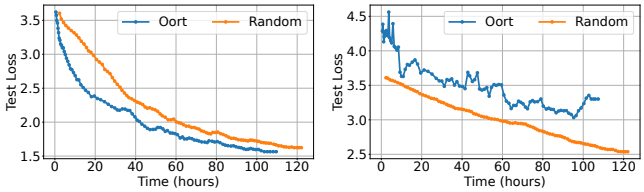
Figure 2. Influence of availability dynamics on test accuracy.

throughout training (i.e., random sampling results in fair selection). ii) A behaviourally heterogeneous case (**BH**): where learners have heterogeneous availability patterns with the hardware and network profiles of learners remaining uniform. Using various CNN and RNN benchmarks on five common FL datasets, we perform experiments across a range of FL settings (i.e., by the varying systems and learning hyper-parameters that influence model quality).Figure 2 shows box plots of the average test accuracy (normalized by the default **U** case) measured for a large number of experiments of different settings in both the **U** and **BH** cases. The results demonstrate that behavioural heterogeneity has a significant impact on model quality. The normalized average test accuracy of **BH** compared to **U** is 0.93× and 0.78× for the median (shown as a green line) and average, respectively. Moreover, the variations in final accuracy across **BH** case are significantly high and can result in training divergence. Next, we review existing systems and motivate our approach.

4 Motivation

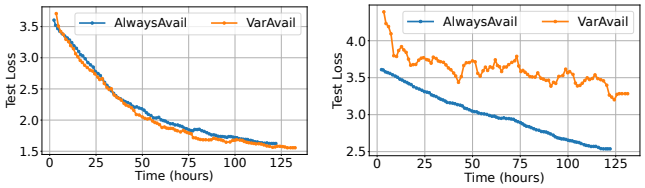
We note that many existing efforts attempt to address data heterogeneity [35, 55] and/or device heterogeneity [40, 41]. Unfortunately, existing system designs do not take into account behavioural heterogeneity which may limit the inclusion of the larger population of the data sources (i.e., some learners may never get selected due to their unavailability). This becomes especially important when data are non-IID which is the typical setting in FL environments where each learner has a unique data distribution [16, 33, 43]. Among recent designs, Oort is a state-of-the-art (SOTA) FL selection method that favours learners with high statistical and system utility [32]. Oort proposes a participant selection algorithm that favours higher utility learners to improve time-to-accuracy. The utility is composed of a statistical term which relates to the convergence speed and a system term which relates to the training time. Systematically, Oort's selection method prefers fast learners to reduce round time and can sometimes trade-off training time to include slow learners whenever statistical efficiency is not improving.

We highlight the trade-offs between two objectives **system efficiency** as targeted by Oort and **clients' representation** as targeted by *A2FL*. These are conflicting optimization goals in FL. Exploring the extremes of these two objectives, we



(a) Uniform IID **(b)** Label-Limited (non-IID)

Figure 3. Influence of data heterogeneous setting.



(a) Uniform IID **(b)** Label-Limited (non-IID)

Figure 4. Influence of dynamic clients' availability.

show empirically how the existing SOTA systems such as Oort fail to perform as designed on common FL benchmarks.

4.1 Training Speed vs. Client Representation

Many FL systems aim to decrease the time taken to reach a target accuracy by prioritizing the selection of fast learners (i.e., system efficiency) [32] or increasing convergence speed by preferring learners with higher data quality (i.e., statistical efficiency) [31, 35, 36, 43]. This results in models that are biased towards learners which, due to external factors, possess faster devices and/or produce data more frequently. Though these approaches improve time-to-accuracy, they do not provide good coverage of the large pool of learners' data and fail to have good client representation. This can be addressed with a selection strategy that is more inclusive over the large population though inevitably results in increased training time due to the potential inclusion of stragglers [58]. It is evident that the two goals present two extremes of the design space which FL designers need to explore and balance against each other. One of the extremes is represented by Oort which aggressively reduces training time by exploiting fast learners while ignoring the diverse set of learners' data distributions. As a result, the trained models are less robust to variations in data distributions when deployed in practice, where non-IID data is the norm rather than the exception. It is intuitive to expect unfair selection to produce a global model that does not cover the majority of learners' data [27, 36]. At the other extreme, the designer could skip the selection phase and invoke all available learners for training to maximize client representation but this also comes at the expense of increased resource wastage [58, 59].

To balance the two goals, an FL system can perform client selection in a manner that ensures high levels of clients representation. This would result in some reduction in time to accuracy. In this work, we aim to leverage this concept and address this gap in existing system designs with a novel approach that places clients' representation at the forefront of FL system design. In the following, we empirically demonstrate that SOTA systems fail to produce satisfactory models for realistic FL scenarios and present our design of *A2FL* which aims to enhance model quality and fairness through maximizing clients' representation in FL training.

4.2 Selection Phase and Client Representation

Commonly uniform random sampling is used by existing FL systems to sample learners during the selection phase [12, 15, 64]. The authors in [32] highlight that this naive method can lead to selecting learners with random computation and communication configurations which results in a large variation in completion time. Consequently, this increases the training time as the server must wait for stragglers to submit their updates. On the contrary, Oort advocates biasing selection towards learners with fast devices to decrease the training time in each round. This is undesirable because the trained model may be biased towards a specific group of learners because the model is trained mainly on this group's data.

To empirically study the impact of the selection algorithm on the final accuracy vs time-to-accuracy, we conduct experiments using Google Speech dataset and train a speech recognition model (i.e., ResNet32) in FL setting for 1,000 rounds. We use two partitions of the data: (1) distributed uniformly the samples among clients from all labels in the dataset (i.e., IID setting); (2) each learner is constrained only to a random ≈10% of the labels but data points for these labels are sampled uniformly (non-IID setting). We compare Oort and Random selection algorithms.

Selection Method Does Matter: To analyze the role of the selection strategy, learners are set to be always available. Fig. 3 shows the average test loss (i.e., averaged over test clients) vs the training wall-clock time. Fig. 3a) shows the IID case where both Oort and Random achieve close to the same final accuracy but Oort outperforms Random in terms of training time by 12 hours. This is because when data is IID, the selection of the learner does not impact the quality of the model and therefore Oort's aggressive approach of minimizing the completion time can help achieve shorter time-to-accuracy. Conversely, Fig. 3b) shows the non-IID case where random selection achieves comparable accuracy to the IID case and outperforms Oort in terms of final model accuracy though still at the expense of higher completion time. Hence, the selection method matters when dealing with non-IID cases and Random selection (if availability is ruled out) leads to higher quality due to its higher chances for a diverse user selection.

Learners' Availability Does Matter: We next examine the role of learners' availability on the performance of selection methods. Hence we set learners' availability based on profiles from a real-world trace of mobile users. To this

end, we analyze and extract profiles from a large-scale user behaviour trace [62] involving more than 136K users of an FL application over a week. Availability is set based on devices being connected to a charger, resulting in availability duration with a long-tail distribution, and the majority (70%) of clients are available for less than 10 minutes. Hence, the availability may have a significant impact on the data distributions the model is trained on which varies per round depending on the current online learners [27, 62].

In the following, we extend our analysis to observe how the variations in availability may impact the performance of selection methods and hence the model quality. To this end, we repeat the previous experiments using the Google Speech benchmark and IID and non-IID data distributions. We compare the execution of the selection algorithm in two different conditions: i) all learners are always available (**AlwaysAvail**); ii) the availability of the learners varies over time depending on the dynamics of users' availability profile (**VarAvail**). Fig. 4 shows the results for the average test loss (i.e., averaged over the test clients) vs the training wall-clock time. We observe that in the IID case, the availability of the learners has almost no impact on the average test loss obtained in the two different approaches. This is because learners with IID datasets possess data samples with similar distributions. On the other hand, in the non-IID case, we observe that the variable availability of the learners has a detrimental impact on the achieved final test loss and hence the quality of the obtained global model.

5 Availability-Aware Federated Learning

From the previous analysis, we observe that in federated learning settings learners' data distributions play a critical role in the quality of the trained model. Therefore, especially in realistic non-IID cases, the model must be exposed to the majority of the learners' data samples (or distributions) to increase its generalization abilities. The state-of-the-art selection method (Oort) has a principled selection method, however, by biasing it towards a certain group of learners (e.g., faster ones) it does not fully address the selection problem. We propose that the selection method should maximize exposure of the model to learners' data while not sacrificing the system efficiency of the training (i.e., time-to-accuracy). This can be achieved by prioritizing clients whose availability for training in the system is limited while they possess valuable data. In essence, this takes into account the variable availability of learners as a key factor which perturbs the global data distribution over time. In the following, we present *A2FL*, an inclusive selection method for federated learning which gives selection priority to the learners with limited availability than the ones that can complete the training faster (i.e., faster computation and/or network).

A2FL aims to mitigate the under-representation of various learners' data distributions during the FL training process without adversely increasing training time. *A2FL* is a novel selection method which prioritizes the least available learners in the future. The goal is to train the model on a larger base of data samples covering the diverse set of data distributions represented by each unique learner.

Algorithm 1: Mixed Selection Algorithm

Input : N_t-Target Number of learners
Input : L_{as}-Learners selected by alternative method
Output: L_s-Selected learners
Initialize $L_s = \Phi$, $P_t = \Phi$
Let S: the server and l: a learner
On_Event ***Learner_Join_Task***
  S: send estimation of the period for the next round;
  l: use FAP and send availability probability p_l
  S: append p_l to the probability list $P_t = P_t \cup p_l$;
On_Event ***End_Selection_Timer***
  S: add l in L_s all unavailable learners $P_l < 0.5$;
  if $len(L_s) < N_t$ **then**
    S: select a random l_a from L_{as}
    S: add l_a to L_s if $l \notin L_s$
  end

5.1 High-Level Design of *A2FL*

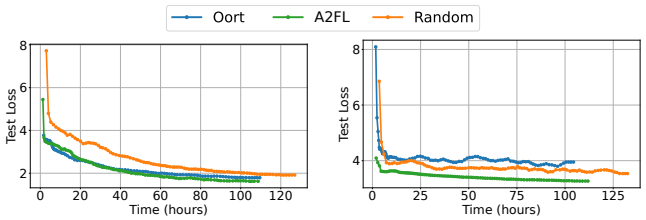
We present the design of *A2FL* by contrasting it with that of Oort. At the high level, from a design perspective, both selection methods introduce a learner selection plug-in module which is responsible for the decision logic on which learners participate in the training rounds. However, *A2FL* consists of two main parts:

1. the Future Availability Prediction (FAP) model which produces the probability of learner's availability;
2. the Availability-based Prioritization (AP) module which selects the learners based on learners' availability.

In principle, Oort avoids the selection of slow learners leaving them under-represented. On the other hand, *A2FL* improves the representation of the full client population by taking into account the learners' availability in the future regardless of their computational and/or communication capabilities. *A2FL* selects ones that are highly unlikely to be present in future rounds and then fills the remaining slots randomly with clients selected by another method (e.g., Random or Oort). We next describe the FAP module which uses a time-series model to produce availability probability for each learner and then communicate the probability to the AP module. Then, the AP module which is responsible for prioritizing participants with low availability.

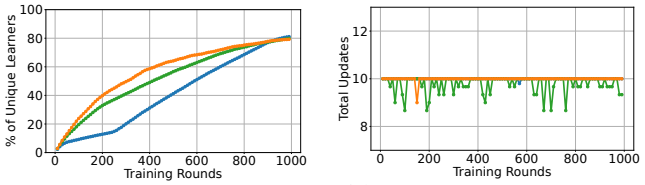
5.2 Future Availability Prediction (FAP)

The FAP model needs to be of low computational overhead on learner devices. Therefore, a linear forecasting model is used [28]. The model is trained locally on devices' changes in



(a) Oort's Data Mapping **(b)** Non-IID Label-Limited

Figure 5. Perf. of selection methods in IID vs non-IID cases



(a) Percent. of unique learners **(b)** No. of aggregated updates

Figure 6. Percentage of unique learners and number of aggregated updates in the Label-limited non-IID case

status (e.g., charging). To evaluate such a model, we use Facebook's Prophet tool [52] which trains a time-series forecasting model using linear regression. We leverage the Stunner dataset [51], which contains event traces collected worldwide from a large-scale number of mobile devices. We use a total of 135 devices after processing a total of one million trace events in May 2018 by filtering out any device with less than 1000 trace samples. For predicting availability, we use the plugged and charging state to train a forecasting model for each device. We use the first half of the devices' samples for training and the second half for testing. The results show that the models predict future states with high accuracy with the values of the mean absolute error (**MAE**), mean square error (**MSE**), and coefficient of determination (**R2Score**) averaged across devices, which are **0.027848**, **0.011563**, and **0.928258**, respectively. Notably, *A2FL* does not compromise users' privacy and integrates with the existing techniques for secure aggregation or differential privacy.

5.3 Availability-based Prioritization (AP) module

The main objective of *A2FL* is to improve all clients' representation in the training process by exposing the model to the wide-spectrum distribution(s) of learners' Non-IID data. Algorithm 1 describes how the AP module selects learners in each training round. Each learner maintains and trains the FAP model periodically on their local device state. For each learner l that joins the FL training task, the AP module sends the learner a future round time slot which can be estimated from the historical round duration. Each learner uses its FAP model to infer its availability probability in the queried time slot. When the selection phase comes to end, the AP module sorts the learners based on their availability probabilities P (breaking ties with a random shuffle) and selects the least available learners to start the round. Selected participants hold off participating for a few rounds (e.g., 5 rounds) if they successfully submit their updates, similar to [12].

6 Evaluation

In this section, we evaluate *A2FL* against the state-of-the-art methods and show its benefits in heterogeneous settings.

Experimental Setting: We run a common speech recognition FL benchmark using Google Speech dataset [57] to be trained on ResNet model [25]. The clients are assigned realistic device and network profiles collected from AI Benchmark [11] and MobiPerf Trace [39], respectively. The experiments are event-driven in which the time is advanced based on the computational and communication profiles of the devices. That is the completion of training and upload/download of the model are based on devices' computation and networks speeds, respectively. To obtain the quality metrics of training the model, we use a cluster of 4 GPU servers and run the clients on them in batches of 4 clients in parallel. We use PyTorch v1.8.0 as the training backend [46]. Similar to Oort, YoGi [47] is used for aggregation. The number of epochs, batch size, and learning rate hyper-parameters are set to 1, 20, 0.005, respectively. The per-round target number of clients is 10. For more details, refer to [32].

Data Partitioning: We use the default data mappings in Oort [32]. And, we use more realistic Non-IID partition in which clients can have only a random 10% of the labels (i.e., 4 out of 35). Then, clients are assigned data samples from each client's pre-chosen labels uniformly at random.

Devices and Availability: learners' devices are assigned a random profile for inference[2] and network latency from AI [11] and MobiPerf [39] benchmarks, respectively. For profiling availability, we use a mobile users trace of events collected, over a period of one week, from over 136k devices from various countries [62].

Experimental Results: We use the same setup as in Oort [32] and enable the real-world behavioural trace (i.e., **VarAvail**). We compare *A2FL*, Random, and Oort selection in terms of the average test loss versus the training time. Fig. 5a and Fig. 5b show that, whether using Oort's data or non-IID data mapping, *A2FL* achieves noticeable lower test loss compared to the other methods. *A2FL* has slightly higher time compared to Oort which biases the selection to lower round time impacting the quality negatively esp. in the non-IID case.

Fig. 6a shows that both *A2FL* and Random have a high rate of unique learners contributing to model training compared to Oort. *A2FL* selects the least available participants first which results in a higher total unique number of learners which explains the better model over other methods. By selecting, the least available learners first, *A2FL* is able to harness their data early in the training to boost the clients' representation in the model especially when data is non-IID.

[2]It is assumed that the training cost is 3× of the inference [32].

Fig. 6b shows that the total aggregated updates are lower for *A2FL*, yet it achieves the best model quality. This shows that having more aggregated updates is not the only factor that contributes to the model convergence, rather the uniqueness of the data points on the model is trained. Note that, the lower number of updates for *A2FL* is because some learners become unavailable after selection during the training. We think the model quality can be improved if these updates were incorporated. This improvement in reducing the number of failed updates is left for future exploration.

7 Conclusion

We focus our study on the impact of behavioural heterogeneity on federated learning. We dissected the main cause for the low-quality models trained by state-of-the-art selection methods in realistic non-IID scenarios. Hence, we revisited the selection strategy which plays a key role in model quality. To this end, we propose inclusive federated learning (*A2FL*) to maximize the diversity of the learners' pool on which the model is trained. Experiments with a real FL benchmark show *A2FL* improves model quality with minimal training cost compared to existing methods. As part of future work, we aim to study the interplay of various selection methods and explore adaptive selection schemes.

References

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

[2] Ahmed M. Abdelmoniem and Marco Canini. 2021. Towards Mitigating Device Heterogeneity in Federated Learning via Adaptive Model Quantization. In *1st Workshop on Machine Learning and Systems (EuroMLSys)*.

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

[4] Ahmed M. Abdelmoniem, Ahmed Elzanaty, Marco Canini, and Mohamed-Slim Alouini. US Patent 63045346, 2022. Statistical-based gradient compression method for distributed training system.

[5] Ahmed M. Abdelmoniem and Yomna M. Abdelmoniem; Ahmed Elzanaty. 2023. A2FL: Availability-Aware Selection for Machine Learning on Clients with Federated Big Data. In *IEEE ICC*.

[6] Ahmed M. Abdelmoniem, Chen-Yu Ho, Pantelis Papageorgiou, and Marco Canini. 2022. Empirical Analysis of Federated Learning in Heterogeneous Environments. In *2nd European Workshop on Machine Learning and Systems (EuroMLSys)*.

[7] Ahmed M. Abdelmoniem, Chen-Yu Ho, Pantelis Papageorgiou, and Marco Canini. 2023. A Comprehensive Empirical Study of Heterogeneity in Federated Learning. *IEEE Internet of Things Journal* (2023), 1–1.

[8] Ahmed M. Abdelmoniem, Atal Narayan Sahu, Marco Canini, and Suhaib A. Fahmy. 2023. REFL: Resource-Efficient Federated Learning. *ACM EuroSys* (2023).

[9] Amna Arouj and Ahmed M. Abdelmoniem. 2022. Towards Energy-Aware Federated Learning on Battery-Powered Clients. In *ACM Workshop on Data Privacy and Federated Learning Technologies for Mobile Edge Network (FedEdge), MobiCom*.

[10] Ravikumar Balakrishnan, Tian Li, Tianyi Zhou, Nageen Himayat, Virginia Smith, and Jeff Bilmes. 2022. Diverse Client Selection for Federated Learning via Submodular Maximization. In *ICLR*.

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

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

[13] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. 2017. Practical Secure Aggregation for Privacy-Preserving Machine Learning. In *CCS*.

[14] K. Bonawitz, F. Salehi, J. Konečný, B. McMahan, and M. Gruteser. 2019. Federated Learning with Autotuned Communication-Efficient Secure Aggregation. In *53rd Asilomar Conference on Signals, Systems, and Computers*. 1222–1226.

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

[16] Zheng Chai, Hannan Fayyaz, Zeshan Fayyaz, Ali Anwar, Yi Zhou, Nathalie Baracaldo, Heiko Ludwig, and Yue Cheng. 2019. Towards Taming the Resource and Data Heterogeneity in Federated Learning. In *USENIX Conference on Operational Machine Learning (OpML)*.

[17] Wenlin Chen, Samuel Horvath, and Peter Richtarik. 2020. Optimal Client Sampling for Federated Learning. *arXiv 2010.13723* (2020).

[18] Y. Chen, Xiaoyan Sun, and Y. Jin. 2019. Communication-Efficient Federated Deep Learning With Layerwise Asynchronous Model Update and Temporally Weighted Aggregation. *IEEE TNNLS* (2019).

[19] Yae Jee Cho, Jianyu Wang, and Gauri Joshi. 2020. Client Selection in Federated Learning: Convergence Analysis and Power-of-Choice Selection Strategies. *arXiv 2010.01243* (2020).

[20] FaceBook. 2021. High-speed library for applying differential privacy for Pytorch. https://github.com/pytorch/opacus

[21] FedAI. 2021. Federated AI Technology Enabler. https://www.fedai.org

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

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

[24] Florian Hartmann, Sunah Suh, Arkadiusz Komarzewski, Tim D. Smith, and Ilana Segall. 2019. Federated Learning for Ranking Browser History Suggestions. *ArXiv 1911.11807* (2019).

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

[26] T. Hsu, Hang Qi, and Matthew Brown. 2020. Federated Visual Classification with Real-World Data Distribution. In *ECCV*.

[27] Jiyue Huang, Chi Hong, Lydia Y. Chen, and Stefanie Roos. 2021. Is Shapley Value fair? Improving Client Selection for Mavericks in Federated Learning. *arXiv 2106.10734* (2021).

[28] Rob J Hyndman and George Athanasopoulos. 2018. *Forecasting: principles and practice, 2nd edition.* OTexts, Melbourne, Australia.

[29] Junchen Jiang, Yuhao Zhou, Ganesh Ananthanarayanan, Yuanchao Shu, and Andrew A. Chien. 2019. Networked Cameras Are the New Big Data Clusters. In *Proceedings of the ACM Workshop on Hot Topics in Video Analytics and Intelligent Edges (HotEdgeVideo)*.

[30] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. 2019. Advances and Open

Problems in Federated Learning. *arXiv 1912.04977* (2019).
[31] Sai Praneeth Karimireddy, Quentin Rebjock, Sebastian U Stich, and Martin Jaggi. 2019. Error Feedback Fixes SignSGD and other Gradient Compression Schemes. In *ICML*.
[32] Fan Lai, Xiangfeng Zhu, Harsha V. Madhyastha, and Mosharaf Chowdhury. 2021. Efficient Federated Learning via Guided Participant Selection. In *USENIX OSDI*.
[33] Qinbin Li, Yiqun Diao, Quan Chen, and Bingsheng He. 2021. Federated Learning on Non-IID Data Silos: An Experimental Study. *arXiv 2102.02079* (2021).
[34] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. 2020. Federated Learning: Challenges, Methods, and Future Directions. *IEEE Signal Processing Magazine* 37, 3 (2020).
[35] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated Optimization in Heterogeneous Networks. In *MLSys*.
[36] Tian Li, Maziar Sanjabi, Ahmad Beirami, and Virginia Smith. 2020. Fair Resource Allocation in Federated Learning. In *ICLR*.
[37] Wenqi Li, Fausto Milletarì, Daguang Xu, Nicola Rieke, Jonny Hancox, Wentao Zhu, Maximilian Baust, Yan Cheng, Sébastien Ourselin, M. Jorge Cardoso, and Andrew Feng. 2019. Privacy-Preserving Federated Brain Tumour Segmentation. In *Machine Learning in Medical Imaging*.
[38] Ruixuan Liu, Fangzhao Wu, Chuhan Wu, Yanlin Wang, Lingjuan Lyu, Hong Chen, and Xing Xie. 2022. No One Left Behind: Inclusive Federated Learning over Heterogeneous Devices. In *ACM SIGKDD*.
[39] M-Lab. 2021. MobiPerf: an open source application for measuring network performance on mobile platforms. https://www.measurementlab.net/tests/mobiperf/
[40] Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. 2018. Learning Differentially Private Recurrent Language Models. In *ICLR*.
[41] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *AISTATS*.
[42] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. 2019. Exploiting unintended feature leakage in collaborative learning. In *IEEE Symposium on Security and Privacy (SP)*.
[43] M. Mohri, Gary Sivek, and A. T. Suresh. 2019. Agnostic Federated Learning. In *ICML*.
[44] Milad Nasr, Reza Shokri, and Amir Houmansadr. 2019. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *IEEE Symposium on Security and Privacy (SP)*.
[45] Takayuki Nishio and Ryo Yonetani. 2018. Client Selection for Federated Learning with Heterogeneous Resources in Mobile Edge. *ArXiv 1804.08333* (2018).
[46] Pytorch.org. 2022. PyTorch. https://pytorch.org/.
[47] Swaroop Ramaswamy, Om Thakkar, Rajiv Mathews, Galen Andrew, H. Brendan McMahan, and Françoise Beaufays. 2020. Training Production Language Models without Memorizing User Data. *arXiv 2009.10031* (2020).
[48] Yichen Ruan, Xiaoxi Zhang, Shu-Che Liang, and Carlee Joe-Wong. 2021. Towards Flexible Device Participation in Federated Learning. In *AISTATS*.
[49] Atal Sahu, Aritra Dutta, Ahmed M. Abdelmoniem, Trambak Banerjee, Marco Canini, and Panos Kalnis. 2021. Rethinking gradient sparsification as total error minimization. In *NeurIPS*.
[50] Felix Sattler, Simon Wiedemann, Klaus-Robert Muller, and Wojciech Samek. 2020. Robust and Communication-Efficient Federated Learning From Non-i.i.d. Data. *IEEE TNNLS* 31, 9 (2020), 3400—3413.
[51] Zoltán Szabó, Krisztián Téglás, Arpád Berta, Márk Jelasity, and Vilmos Bilicki. 2019. Stunner: A Smart Phone Trace for Developing Decentralized Edge Systems. In *19th International Conference on Distributed Applications and Interoperable Systems (DAIS)*.
[52] Sean J Taylor and Benjamin Letham. 2017. Forecasting at scale. *PeerJ Preprints 5:e3190v2* (2017).
[53] Apple Differential Privacy Team. 2017. Learning with privacy at scale. *Apple Machine Learning Journal* (2017).
[54] tensorflow.org. 2020. TensorFlow Federated: Machine Learning on Decentralized Data. https://www.tensorflow.org/federated
[55] Jianyu Wang and Gauri Joshi. 2019. Adaptive Communication Strategies to Achieve the Best Error-Runtime Trade-off in Local-Update SGD. In *MLSys*.
[56] Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H. Vincent Poor. 2020. Tackling the Objective Inconsistency Problem in Heterogeneous Federated Optimization. In *NeurIPS*.
[57] P. Warden. 2018. Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition. *ArXiv 1804.03209* (2018).
[58] Wentai Wu, Ligang He, Weiwei Lin, Rui Mao, Carsten Maple, and Stephen Jarvis. 2021. SAFA: A Semi-Asynchronous Protocol for Fast Federated Learning With Low Overhead. *IEEE Trans. Comput.* 70, 5 (2021).
[59] Cong Xie, Sanmi Koyejo, and Indranil Gupta. 2019. Asynchronous Federated Optimization. *arXiv 1903.03934* (2019).
[60] Hang Xu, Chen-Yu Ho, Ahmed M. Abdelmoniem, Aritra Dutta, El Houcine Bergou, Konstantinos Karatsenidis, Marco Canini, and Panos Kalnis. 2021. GRACE: A Compressed Communication Framework for Distributed Machine Learning. In *IEEE ICDCS*.
[61] Francis Y. Yan, Hudson Ayers, Chenzhi Zhu, Sadjad Fouladi, James Hong, Keyi Zhang, Philip Levis, and Keith Winstein. 2020. Learning in situ: a randomized experiment in video streaming . In *USENIX NSDI*.
[62] Chengxu Yang, Qipeng Wang, Mengwei Xu, Zhenpeng Chen, Kaigui Bian, Yunxin Liu, and Xuanzhe Liu. 2021. Characterizing Impacts of Heterogeneity in Federated Learning upon Large-Scale Smartphone Data. In *The Web Conference*.
[63] Lixuan Yang, Cedric Beliard, and Dario Rossi. 2020. Heterogeneous Data-Aware Federated Learning. In *IJCAI - Federated learning workshop*.
[64] Timothy Yang, Galen Andrew, Hubert Eichner, Haicheng Sun, Wei Li, Nicholas Kong, Daniel Ramage, and Françoise Beaufays. 2018. Applied Federated Learning: Improving Google Keyboard Query Suggestions. *arXiv 1812.02903* (2018).