GeoFL: A Framework for Efficient Geo-Distributed Cross-Device Federated Learning

Maolin Gan*, Lanpeng Li*, Samiul Alam†, Li Liu‡, Luyang Liu§, Mi Zhang†, Zhichao Cao*
*Michigan State University, †The Ohio State University, ‡Tsinghua University, §Google DeepMind
{ganmaoli, lilanpen, caozc}@msu.edu, {alam.140, mizhang.1}@osu.edu, liuli95@mail.tsinghua.edu.cn, luyangliu@google.com

***Abstract*—In this paper, GeoFL develops a hierarchical federated learning (FL) framework to address the unique challenges in large-scale geo-distributed scenarios. The key idea is to deploy multiple aggregators to geo-distributed clients and aggregate the local model and the global model efficiently and effectively. By assigning each aggregator as a relay layer, GeoFL can elaborately aggregate the geo-distributed clients and systematically determine when to upload the model to the central server based on bandwidth to efficiently update the global model under inadequate and heterogeneous WAN bandwidth constraints. GeoFL designs three key components to optimize the inefficient model aggregation and cope with the non-importance model updates. It further addresses the statistical heterogeneity across geo-distributed aggregators by considering the clients' graph relationship, delivering an end-to-end client-aggregator-server architecture for large-scale clients. Compared with existing works, our results on large-scale real-life datasets show that GeoFL speeds up the training process by $1.4\times$–$8\times$ and reduces 6%–80% unnecessary communication rounds between the aggregator and the central server.**

I. Introduction

Recently, more and more large machine learning models require diverse and sufficient datasets collected across much wider geographical areas in different cities, countries, or even continents for training. For example, Google Landmarks Dataset v2 (GLDv2) [1] is a worldwide dataset, containing over 5M images collected from over 200k human-made and natural landmarks across 246 of the 249 countries for widely-adopt image retrieval [2] and instance recognition [3] tasks. However, traditional centralized training raises significant concerns regarding end-user privacy [4]. On the other hand, cross-device federated learning (FL) enhances the privacy of user data by training models locally at a client and aggregating the model updates at a central server [5]–[8]. Unfortunately, the single-layer client-server architecture most existing FL studies adopt is not capable of supporting FL at such geo-distributed scale due to significantly heterogeneous network connections [9].

In this work, we bridge this critical gap and propose GeoFL, a framework for efficient geo-distributed cross-device federated learning. As shown in Figure 1, GeoFL adopts a hierarchical architecture where clients located in wide geographical areas first communicate with edge servers (i.e., aggregators) via mobile network, whereas the communication between aggregators and the central server is conducted over Wide Area Network (WAN).

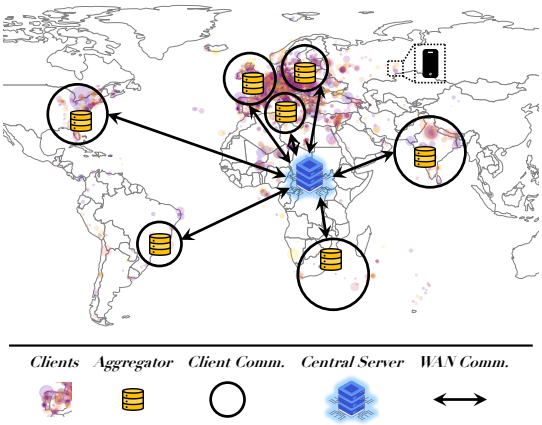


Fig. 1: Overview of geo-distributed cross-device federated learning.

Although a few hierarchical FL frameworks have been proposed [10], [11], these frameworks are *not designed for geo-distributed FL* given that they did not use real-world WAN bandwidth measurements to identify the distinctive characteristics of geo-distributed FL, making their proposed techniques fall short in the geo-distributed setting. The underpinning principle behind GeoFL is to consider the inadequacy and heterogeneity of WAN bandwidths and involve bandwidth-efficient and heterogeneity-aware schemes to enhance the efficiency of geo-distributed FL.

The design of GeoFL involves three key challenges.

- **Challenge #1: Scheduling Aggregation under Inadequate and Heterogeneous WAN Bandwidth.** We observe that WAN bandwidth between aggregators and the central server is inadequate and heterogeneous. Since GeoFL aims to optimize the trade-off between the return (i.e., the global model's time-to-accuracy) and the aggregator-server communicating cost, how to schedule the central server's communication with multiple aggregators represents the first challenge.
- **Challenge #2: Buffered Model Updates on Aggregators.** Although an adaptive aggregator-server communication approach can fully optimize the time efficiency for model aggregation, the global model could have been updated several times when receiving the trained local model update from a certain aggregator. Therefore, it is necessary to minimize the impact, which can otherwise

result in biases for the global model training.

- **Challenge #3: Aggregator Heterogeneity.** Aggregators are by nature heterogeneous. This is because aggregators, with varying population sizes, connect clients holding distinct data distributions. Therefore, it's essential to incorporate an optimized model aggregation strategy when updating the global model from multiple aggregators to avoid low-quality aggregation in non-IID situations.

To address the first challenge, GeoFL employs an importance-aware buffer for each aggregator and quantization for model updates so that aggregators can determine when to upload model updates to the central server and reduce communication costs. Specifically, we quantify the importance of model updates and measure them at each aggregator. Aggregators can buffer continuous model updates until their importance reaches an adaptive importance threshold related to aggregators' bandwidth, thus balancing the impact of aggregators' model updates and avoiding unnecessary aggregator-server communication. Quantized model updates further speed up FL training by reducing the bits of parameters.

Second, to avoid the issue of some aggregators not reaching the importance threshold for a long time, which prevents the global model from receiving timely updates of all clients, GeoFL designs an update up-bound for each aggregator. Once the local training rounds reach this limit, the aggregator can actively request to upload its local model. GeoFL also assigns a tuned weight for its model aggregation at the central server to mitigate the model bias.

Besides, GeoFL tackles aggregator heterogeneity in population size and client data distributions. By assigning different sizes of participants per aggregator, the central server can adapt its local training pace for a globally uniform participant selection. GeoFL also considers the dynamic graph relationship across clients with non-IID and unbalanced data to optimize model aggregation.

System Implementation and Evaluation Results. We have built the prototype of GeoFL on the public benchmark [12] and conducted comprehensive evaluation on the real-life mobile scenarios with massive geo-distributed clients, including image classification (OpenImage [13]), landmark recognition (Google Landmark [14]), voice command recognition (Google Speech [15]), activity recognition (HARBox [16]) and next-word prediction (StackOverflow [17]). We compare GeoFL against three baselines: HierFAVG [10], HierFAVG-Async, and Async-HFL [11]. Our results on different large-scale real-life datasets show that GeoFL can significantly accelerate the global model training process and reduce unnecessary communication rounds between aggregators and the central server, thereby considerably saving communication bandwidth.

In summary, our work makes three major contributions:

- To the best of our knowledge, GeoFL is the first FL framework that considers largely geo-distributed clients in constrained bandwidth environments. GeoFL uses real-world WAN bandwidth measurements to demonstrate the distinctive challenges of geo-distributed FL.

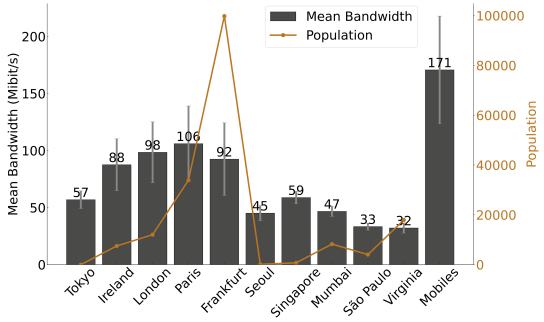


Fig. 2: Average WAN bandwidth measured across ten global Amazon EC2 sites, the mean network bandwidth of mobile clients, and aggregators' population sizes.

- We propose the design of geo-distributed aggregators, which incorporate local clients' communication and adaptively upload quantized important model updates to the central server with less communication cost. Additionally, we also present the design of non-importance model updates tuning and graph-based model aggregation to mitigate model bias issues.
- We have implemented GeoFL and evaluated it with extensive experiments. Compared with existing works, GeoFL speeds up the training process by $1.4\times$–$8\times$ and reduces 6%–80% unnecessary communication rounds between the aggregator and the central server with various tasks.

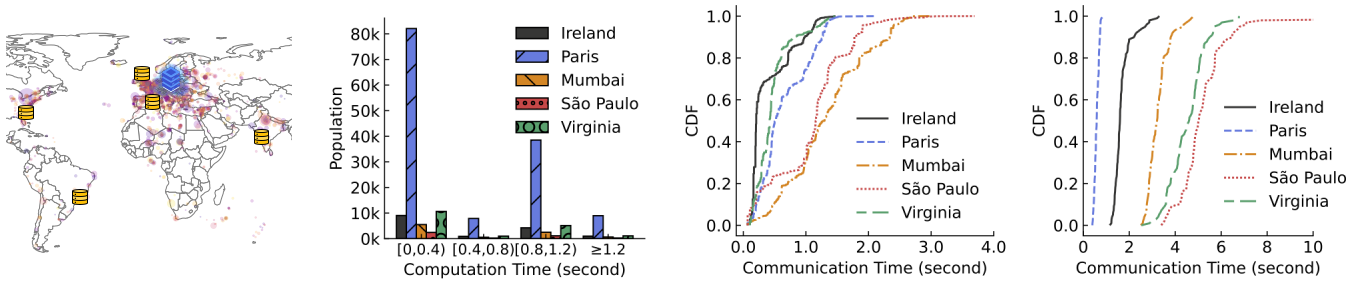
II. Empirical Study and Motivation

A. Geo-Distributed WAN Measurement

To understand the unique characteristics of designing geo-distributed FL systems, we conduct experiments to measure the bandwidths of WAN on a global scale. To do so, we measure the WAN bandwidth for each pair of Amazon Elastic Compute Cloud (EC2) sites located at 10 different regions across the globe – Tokyo, Ireland, London, Paris, Frankfurt, Seoul, Singapore, Mumbai, São Paulo, and Virginia. The selection of site locations is determined by the clients' geo-distribution in Figure 1 and aims to cover the spread of clients best. We use `iperf3` [18] to measure the WAN bandwidth and report the average over 100 communication rounds. We compare the WAN bandwidth of the aggregators we measure with the clients' network bandwidth which are profiled with real-life measurements on mobile platforms [12], [19], [20]. Additionally, we extract the clients' landmark coordinates from the Google Landmarks Dataset-v2 (GLDv2) [1], the largest landmark dataset, and assign them to the nearest aggregator to compare the population size of aggregators.

Figure 2's bar chart displays the average WAN bandwidth between each aggregator and the remaining nine aggregators, as well as the average network bandwidth from clients on mobile devices, with error bars indicating standard error. The line chart shows the population sizes of the ten aggregators. We have three key observations as follows.

Observation #1: Inadequacy of WAN Bandwidths in Comparison to Mobile Network. The WAN bandwidths between two different sites are *much more inadequate* compared to



(a) Geo-distribution of clients. (b) Clients' local computation. (c) Client-aggregator communication. (d) Aggregator-server communication.

Fig. 3: Performance of naive geo-distributed cross-device FL systems.

the mobile network bandwidth obtained from real-life mobile device measurement [12], [19], [20]. For example, the average mobile network bandwidth for all clients is 171 Mibit/s, while the average WAN bandwidth between Virginia and the other nine aggregators is only 32 Mibit/s.

Observation #2: Heterogeneity of WAN Bandwidths. For sites that are geographically close (e.g. Ireland, London, Paris, and Frankfurt in Europe), their mutual WAN bandwidth benefits from the geographical proximity, which makes their respective average WAN bandwidth relatively higher than the other six aggregators. For example, the average bandwidth from Paris to the other nine aggregators is 106 Mibit/s, while São Paulo is only 33 Mibit/s.

Observation #3: Unbalanced Populations across Aggregators. The clients' geo-locations extracted from GLDv2 have up to 180k coordinates. The populations within aggregators from various geographic locations are notably unbalanced. For example, Frankfurt contains more than 50% of clients' coordinates, while Seoul and Tokyo account for less than 1%.

B. Limitation of Naive Geo-Distributed FL

The *inadequacy and heterogeneity* of WAN bandwidths directly impact the performance of FL systems when deployed in geo-distributed settings. To identify the factors affecting geo-distributed FL design and quantify their impact, we study the communication and computation performance of a naive geo-distributed FL system, which has a central server (marked as a cube in Figure 3(a)) deployed in Frankfurt and five aggregators (marked as cylinders in Figure 3(a)) deployed in Ireland, Paris, Mumbai, São Paulo, and Virginia distributed across four continents.

Given that we extract 180k coordinates from GLDv2, representing 180k clients, we choose Frankfurt as the central server due to its shortest accumulated distance to all the aggregators and clients, Additionally, we assign clients to the nearest aggregator based on their distance to each of the five different aggregators, with the client number scattered across five aggregators ranging from 4k to 120k. Figure 3(a) depicts client locations with colored circles of different sizes. We acquire clients' device and network bandwidth information from mobile device measurement studies [12], [19], and we follow standard FL profiling [20] to emulate the device computation runtime during local FL training using data from AI Benchmark [21]. Clients participating in the FL process collaboratively train a 168MB Transformer model [22] with 21M parameters, and communicate with aggregators via mobile networks, while aggregators use measured WAN to communicate the model with the central server.

We measure three key metrics that affect the efficiency performance of geo-distributed FL systems in such a hierarchical structure: (1) client computation time of local FL training; (2) client-aggregator communication time via mobile network; (3) aggregator-server communication time via WAN.

Figure 3(b) shows the distribution of local computation time of participated geo-distributed clients. As shown, the computation time of one local training round for all clients falls within $[0.097s, 1.283s]$, and a significant portion of the client population has a device runtime of less than $0.4s$.

We map the mobile device bandwidth measurements from existing FL frameworks to our geo-distributed clients based on client-aggregator distance, and examine the average time spent on model transmission between clients and aggregators during each round of FL training. Figure 3(c) illustrates the cumulative distribution function (CDF) of the client-aggregator communication time. As shown, 99% of client-aggregator communication time across the five aggregators is less than $3s$ and the median is only $0.5s$.

We adopt the WAN bandwidth measured in §II-A to evaluate the communication overhead between each aggregator and the central server. Figure 3(d) shows the CDF of the aggregator-server communication time via WAN. The medium aggregator-server communication time ($3.2s$) is $6.4\times$ longer than the medium client-aggregator time ($0.5s$). Moreover, we notice that the heterogeneity of WAN bandwidths across geo-distributed aggregators directly translates to the heterogeneity of aggregator-server communication time via WAN. Specifically, the medium communication time for Frankfurt-São Paulo link with narrower WAN bandwidth is $5s$. This is $8.62\times$ longer than the Frankfurt-Paris link ($0.58s$) with relatively wider WAN bandwidth.

It is obvious that in naive geo-distributed FL systems, the efficiency is most significantly affected by the communication time between aggregators and the server over the WAN, especially if each training round's global model requires waiting for the central server to aggregate the locally trained models from clients under all aggregators through such insufficient and heterogeneous WAN bandwidth. The results motivate us to design GeoFL that takes real-world WAN bandwidth into

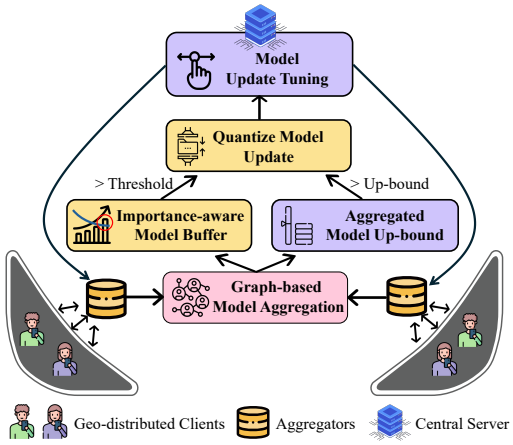


Fig. 4: The architecture of GeoFL.

account to minimize time consumption as much as possible.

III. System Overview

Figure 4 illustrates the overall architecture of GeoFL, which has three key components. First, given the unbalanced population distribution, geo-distributed aggregators in GeoFL enable their large-scale clients to participate in FL training proportionally to population sizes and can leverage dynamic graph relationships between client models to perform graph-based aggregation, thereby enhancing local models and mitigating the impact of imbalanced data distribution (§IV-C).

Second, to address the factors significantly affecting geo-distributed FL systems, GeoFL measures and calculates the importance of model updates at each aggregator and uses an importance-aware buffer with adaptive thresholds related to WAN bandwidth to determine when to upload model updates. Once the importance of the model updates exceeds the adaptive threshold, aggregators can quantize the model update and send it to the central server (§IV-A), thus saving the precious WAN bandwidth between aggregators and the central server while reducing the aggregation latency.

In addition, to prevent the importance of each aggregator's model update from failing to reach the threshold and causing model bias due to too many local training rounds, GeoFL sets an up-bound for aggregation rounds so that model updates can be quantized and uploaded promptly. At the same time, the central server also tunes the uploaded model weights according to the updated degree of different aggregators (§IV-B). In the following, we describe these components in detail.

IV. Design of GeoFL

A. Importance-aware Quantized Model Update

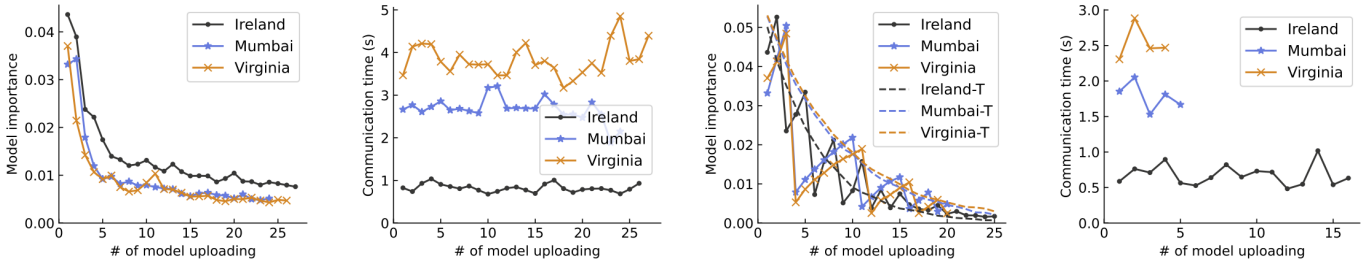
This section focuses on reducing the communication cost of GeoFL by letting aggregators only send important quantized model updates to the central server. We first measure the importance of model updates at each aggregator in a real-life FL dataset for image classification [13] when deploying a geo-distributed FL system with three aggregators in Ireland, Virginia, and Mumbai and the central server in Frankfurt. Then, GeoFL shows the design of an importance-aware buffer and the quantization for each aggregator's model communication to the central server.

1) Model Update Importance Measurement: In FL systems, the importance of model updates is determined by the extent to which each update contributes to overall model performance [23], which typically involves assessing the model differences each update brings by distance metrics, such as the L2 norm or cosine distance. L2-norm can reflect the change in the absolute value of the model parameters, and cosine distance can show the direction difference between the two model parameters, which respectively reflect the changes of the model parameters in different dimensions, and existing work usually focuses on one of them to sample [24]–[26].

To fully utilize the information from different metrics, we combine the two. On the one hand, we calculate the rate of parameter change between the global model and aggregator model under the L2 norm (L). On the other hand, we calculate the normalized cosine similarity of the two models' parameters (Cos_{norm}) by adjusting the range of original cosine distance from $[-1, 1]$ to $[0, 1]$. When Cos_{norm} is closer to 1, it means that the similarity between the two models is higher, and on the contrary, it is lower. We define importance as S on the weighted average of L and $(1 - Cos_{norm})$ to measure the importance of model update from each aggregator.

Given three geo-distributed aggregators (i.e., Ireland, Mumbai, and Virginia) with the central server in Frankfurt, in order to observe the variation in the importance of model updates over training rounds, we calculate the importance at each aggregator after it aggregates local client models and make the aggregator immediately send the update to the central server without waiting for the other aggregators. We also measure the communication time between each aggregator and the central server each round. Figure 5(a) shows that most model update importance values fall within [0.5%, 5%] and approach zero as SGD evolves. Besides, we further notice that the communication time between each aggregator varies, with aggregators that are further away from the central server having lower bandwidth and thus longer communication time shown in Figure 5(b). For instance, Virginia takes an average of about $4s$ to communicate model updates each time, while Ireland only requires around $1s$. However, being agnostic to the models' importance wastes the precious WAN bandwidth by communicating those "unimportant" models. GeoFL proposes an importance-aware model buffer to accumulate these model updates for each aggregator until they are important enough and quantizes the model updates before sending. As such, we can shrink the communication time by reducing the communication overhead of the aggregator-server directly.

2) Importance-aware Model Buffer and Quantization: To determine the best time to upload the model for each aggregator, GeoFL proposes to find the sweet point between aggregation efficiency and communication cost. Given our importance metric, an intuitive way is to set an importance threshold by which the aggregator can determine whether to upload the model or accumulate it in its buffer. However, as training converges, model update importance approaches



(a) Round-varying importance. (b) Communication time cost. (c) With GeoFL's importance buffer. (d) Fewer rounds and time.

Fig. 5: The importance of the model updates from aggregators and the communication costs for communicating with the central server (Frankfurt).

zero, and a fixed threshold can only hinder aggregations after certain points, leading to sub-optimal performance. Also, the WAN bandwidth heterogeneity of different aggregators will also affect the best upload time. GeoFL designs an adaptive importance threshold at the training round r_i as below:

$$S_i = max\left(S_{min}, S_0 \times (\alpha - (1-\alpha) \times B_i)^{(r_i)}\right), \quad (1)$$

where S_0 and S_{min} denote the pre-defined importance threshold and the lower bound, respectively. As the number of local training rounds increases, the decay factor α (defaulted 0.95) works with the aggregator's normalized WAN bandwidth B_i to control the reduction speed of S_0. Higher bandwidth promotes a faster decay. Figure 5(c) presents the adaptive importance threshold for each aggregator. We also adopt quantization, which is used to convert the model parameters from Float32 to target bits so that the total number of uploaded bits could be reduced [27], to reduce communication time. When the local model aggregated by the aggregator reaches the importance threshold, GeoFL quantizes the model update parameters to Float16 and sends it to the central server, thus achieving 2× saving with almost no impact on model accuracy [28].

We further measure the importance of uploaded model updates and their communication time to evaluate the effectiveness of GeoFL's importance-aware buffer and quantization. As shown in Figure 5(c), each aggregator could accumulates model importance under the buffering strategy, uploading only upon reaching the adaptive threshold. Thus, with the same local training rounds in Figure 5(a) and 5(c), the design reduces the number of communication rounds and time significantly in Figure 5(d). Especially for aggregators with lower bandwidth (Mumbai and Virginia), the number of upload times is reduced more, thereby better reducing WAN overhead.

B. Handling Non-importance Model Updates

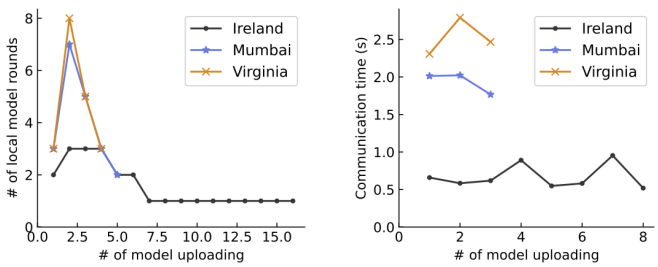
With GeoFL's importance-aware quantized model aggregation, the global model could have been updated several times before aggregating certain aggregators' updates as non-importance updates are not uploaded. Meanwhile, the local training rounds between two adjacent communications for each aggregator with the central server also affect the model divergence. We present the side-effect of GeoFL's buffer on increasing local training rounds and tune the non-importance model updates by elaborately designing the model uploading and aggregation pipeline to mitigate these effects.

1) Side-effect of GeoFL's Buffer: We use the same methodology in §IV-A but record the local training rounds between the aggregator's adjacent communications to the central server. As illustrated in Figure 6(a), GeoFL maintains a list of the local training rounds for each aggregator. These rounds should be neither too large nor too small, as larger rounds reduce the effectiveness of global model aggregation, while smaller rounds undermine time efficiency due to the frequent use of inadequate WAN bandwidth for model communication. A sole importance-aware buffer shrinks too fast in Figure 5(c) so that Ireland with the local training rounds as 1 in Figure 6(a). In contrast, the model update importance increases slower for Mumbai and Virginia, resulting in the larger local training rounds in Figure 6(a).

2) Non-importance Model Updates Tuning: GeoFL resolves the issue in §IV-B1 from both the aggregators and central server sides. First, the aggregator can upload the local model when its local training round reaches the pre-defined upbound R_0 (default 5), ensuring constrained model divergence between the local and global model for each aggregator. Meanwhile, to coordinate GeoFL's importance-aware buffer and account for non-uploaded "unimportant" updates, we add R_0 to Equation 1 and tune the importance threshold as follows:

$$S_i = max\left(S_{min}, S_0 \times (\alpha - (1-\alpha) \times B_i)^{(r_i/R_0)}\right). \quad (2)$$

We show the impact of incorporating two strategies on time cost in Figure 6(b). Given the same local training rounds, it reduces aggregator-server communications from 16 (Figure 5(d)) to 8 (Figure 6(b)), while keeping local training rounds within R_0. Second, GeoFL designs a tuned global model aggregation upon receiving the update from an aggregator a to control its impact on the global model. The central server maintains a list of per-aggregator models and each aggregator's number of involved global rounds since its last communication to the central server [29]. Upon receiving the update from aggregator a, we first derive the aggregator set $\mathbb{A}^*$, whose models have been aggregated to the central server after aggregator a's last communication. Then, we obtain the involved global training rounds vector G for each aggregator in $\mathbb{A}^*$, with the maximum value corresponding to aggregator a. Eventually, we derive the aggregated model M_a, which will be distributed to aggregator



(a) w/o up-bound. (b) with up-bound.

Fig. 6: Aggregators' local training rounds for model updates.

a for the next training round as follows:

$$M_a = \sum_{i \in \mathbb{A}^*} c_i M_i, where\ c_i = \frac{(G_i + 1)^{-\beta}}{\sum_{k \in \mathbb{A}^*} (G_k + 1)^{-\beta}}, \quad (3)$$

where c_i with the hyper-parameter β (e.g., 0.2) measures the contribution of a model update [29].

C. Graph-based Model Aggregation

Despite GeoFL designs strategies to enhance aggregator-server communication, aggregators' model can be inferior by simply aggregating client models with different distributions, as it neglects the high degree of non-IID and unbalanced clients within aggregators. Aggregators across diverse geographical areas inherently exhibit heterogeneity, including population sizes and client data distributions (e.g., photos taken in different locations). The similarities and differences in data distribution among clients can be reflected by their local model parameters. Exploring the relationships between their parameters can adjust the model contribution of distributions with different proportions. Graph networks are effective for capturing dependencies among data [30], and we design them to catch parameter correlations among clients. As shown in Figure 7, GeoFL adapts its training strategy to balance participating clients from each aggregator and employs graph networks to dynamically explore graph relationships between clients. Through graph networks, the connection weights between clients can be learned. The aggregator can weight each client's model based on the parameters of neighboring nodes, enabling nodes with similar distributions to contribute more. Thus, geo-distributed aggregators can optimize model aggregation based on their own distribution.

To enhance each aggregator's model, GeoFL adapts its training strategy from top to down. Given the total number of clients, the central server first assigns different numbers of participating clients for each aggregator, which is proportional to its population size. Then, for client models uploaded to the aggregator, the aggregator quantifies the similarity among the uploaded client parameters based on their pairwise Euclidean distances and constructs an initial adjacency matrix A_{ij}:

$$A_{ij} = \mathcal{N}\left(\sqrt{\sum_{k=1}^{d}(p_{ik} - p_{jk})^2}\right), \quad (4)$$

where p_{ik} and p_{jk} are the parameter vectors of clients i and j respectively, d is the dimensionality of the parameter vectors,

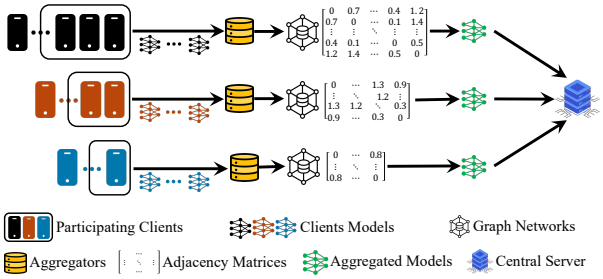


Fig. 7: Graph-based Model Aggregation.

and $\mathcal{N}$ represents the normalization function applied to the distances. This adjacency matrix is subsequently refined by the Graph Attention Network (GAT) [31], which consists of several GATConv layers endowed with attention mechanisms. GAT applies attention across multiple heads to dynamically capture and update client relationships, iteratively updating the adjacency matrix to emphasize more relevant connections and reduce the impact of less relevant ones by using Mean Squared Error (MSE) loss $\mathcal{L}_{\text{MSE}}$ to optimize:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{n^2} \sum_{i,j=1}^{n} (\hat{A}_{ij} - A_{ij})^2, \quad (5)$$

where n is the number of clients and $\hat{A}_{ij}$ is the computed adjacency matrix from the GAT output. Once the GAT processes the graph structure, Graph Convolutional Network (GCN) [32] layers utilize the adjacency matrix to guide the aggregation of client model parameters. The aggregators employ graph convolution, aggregating neighboring nodes to reconstruct each uploaded client update based on the learned graph structure and aggregate the reconstructed models to obtain an enhanced model. This design refines the aggregation mechanism, allowing the system to adaptively optimize learning outcomes based on the evolving relationships in clients' data distributions.

V. Evaluation

In this section, we evaluate the performance of GeoFL with the aim to answer the following questions.

- **Q1 (§V-C)**: *Does GeoFL outperform status quo? If so, what are the reasons?*
- **Q2 (§V-D)**: *How effective is each key component incorporated in GeoFL?*
- **Q3 (§V-E)**: *How is the performance GeoFL affected by diverse hyper-parameters?*

A. System Implementation

We implement GeoFL using the FedScale framework [12] and deploy it on 32 NVIDIA Tesla P100 GPUs to emulate large-scale real-life FL scenarios. We obtain clients' geo-locations from the landmark coordinates provided by GLDv2 [1]. We consider mobile devices as clients and incorporate computation resources, network throughput/connectivity, and availability of clients extracted from real-world mobile device measurements into client-aggregator communication [19]–[21]. We further acquire aggregator-server network bandwidth with our real-world measurements on Amazon EC2 instances in different regions.

TABLE I: Synopsis of datasets, types, and ML models.

Dataset	Type	Model	# of Clients
OpenImage [13]	Image	MobileNet [34]	7,903
Google Speech [15]	Audio	ResNet-34 [35]	2,187
Landmarks-User-160k [14]	Image	EfficientNet [36]	1,262
HARBox [16]	IMU	Customized [33]	121
StackOverflow [17]	Text	Albert-v2 [37]	342,477

B. Experimental Methodology

Tasks, Datasets, and Models: We evaluate GeoFL on five real-world datasets with different data types and scales designed for five categories of FL applications. Each dataset relies on the collection information to indicate the corresponding clients (e.g., USER ID for each data sample). Thus they can vary in data quantities, distribution, and outputs and are usually Non-IID [20], [33]. Table I summarizes data types and models for each dataset, and we list the details of each task below.

- Image Classification. OpenImage [13] contains 1.1 million images from around 7,903 clients, which play a critical role in enabling worldwide machines to identify what objects are present in the image.
- Speech Recognition. Google Speech Dataset [15] encompasses over 100,000 voice commands from 2,187 clients, aimed at assisting the training and evaluation of keyword recognition systems across different manufacturers.
- Landmark Recognition. Landmarks-User-160k [14], containing 160k images of 2,028 landmarks from 1,262 users, facilitates the development of large-scale landmark recognition systems without the need to upload or store private user photos centrally.
- Human Activity Recognition. HARBox [16] collects 34,115 samples of 9-axis IMU data from 121 users' smartphones via crowdsourcing for scalable and robust human activity recognition.
- Natural Language Processing. StackOverflow dataset [17] collects the posts, votes, tags, and badges on StackOverflow for the next-word predictions, with 42 million data samples from 342,477 clients.

Parameters Settings: We use three aggregators in Ireland, Virginia, and Mumbai, with the central server in Frankfurt. Regarding client selection, 30 to 90 participating clients in total are selected, depending on the number of feasible clients in each dataset. Each client trains for 5 local steps with a mini-batch size of 16 per training round. The initial learning rate for clients is $4e-5$ for customized DNN model [33] and Albert [37] and 0.04 for other models. We set decay factor α as 0.95 for the importance threshold in Equation 2, with an initial value S_0 from 0.4% to 20% for different tasks. The up-bound local training round for each aggregator is $R_0 = 5$. GAT is trained for 20 iterations to generate the adjacency matrix for each global aggregation, and GCN consists of 2 layers.

Baselines: We compare GeoFL against three baselines under the same geo-distributed FL configuration.

- **HierFAVG [10].** HierFAVG adopts the hierarchical architecture, in which the central server cannot start the next-round training until it aggregates all the local model updates from its geo-distributed aggregators.
- **HierFAVG-Async.** HierFAVG-Async is a variant of HierFAVG and it keeps all aggregators running based on their best-effort communication and the central server updates its global model once it receives the model updates.
- **Async-HFL [11].** Async-HFL is the state-of-the-art asynchronous FL method where clients upload updates after completing 5 epochs on all local data, while the aggregators upload after 20 updates. Unlike the other two baselines and GeoFL, Async-HFL trains all local samples of selected clients for 5 epochs, whereas the other three methods use the same mini-batch size of each training round with a fixed local training iteration.

Evaluation Metrics: We demonstrate the performance using these metrics:
(1) Target final accuracy after testing accuracy stabilizes. We adopt perplexity for the next-word prediction task, which is better when lower [33].
(2) End-to-end clock time and global training rounds required to reach target final accuracy. The wall clock time includes the local computation overhead and the time duration of aggregation communication.

C. Overall Performance

Table II summarizes the final accuracy, corresponding training wall clock time, and rounds for all four FL frameworks in terms of five different tasks. The detailed results are described as follows:

GeoFL speeds up the wall clock time to reach the target final accuracy. GeoFL achieves the final target accuracy in less time and demonstrates superior or comparable accuracy compared with all baselines. For example, GeoFL reaches the target final accuracy $8\times$ (2.11h to 0.28h), $2\times$ (0.66h to 0.28h), and $7\times$ (1.95h to 0.28h) faster than HierFAVG, HierFAVG-Async, and Async-HFL in terms of wall-clock time on OpenImage [13], while the speedup is $5\times$ (10.20h to 2.10h), $2\times$ (4.00h to 2.10h), and $7\times$ (15.37h to 2.10h) on the Google Speech [15]. Besides, GeoFL achieves about 1.56% (67.55% to 69.11%) higher final accuracy on OpenImage in comparison with HierFAVG, while our final accuracy on Google Speech increased significantly by 15.92% (49.53% to 65.45%) compared to Async-HFL.

These speedups with the wall clock time reduction stem from the GeoFL's specially designed update strategy on the aggregator-server communication. GeoFL's importance-aware quantized model updates enable aggregators to decide the timing of uploads based on dynamic importance thresholds, avoiding waiting for stragglers in aggregators in each training round and reducing communication rounds for non-importance updates to the central server. The up-bound on local training models further prevents excessive local training rounds. Thus, it can reach the final accuracy with less wall-clock time than three baselines, and the speedup is at least $1.4\times$ (3.06h to 2.16h) and at most $8\times$ (2.11h to 0.28h) for various tasks. In addition, the significant boost in performance of GeoFL can be attributed to its model update tuning capability and

TABLE II: GeoFL's performance summary on its final accuracy, training time, and rounds compared with the HierFAVG, HierFAVG-Async, and Async-HFL baselines.

Dataset	HierFAVG			HierFAVG-Async			Async-HFL			GeoFL		
	Accuracy	Time	Round	Accuracy	Time	Round	Accuracy	Time	Round	Accuracy	Time	Round
OpenImage [13]	67.55%	2.11h	280	66.90%	0.66h	410	65.04%	1.95h	240	**69.11%**	**0.28h**	**80**
Google Speech [15]	65.00%	10.20h	130	63.86%	4.00h	320	49.53%	15.37h	430	**65.45%**	**2.10h**	**90**
Landmarks-User-160k [14]	42.83%	13.35h	800	41.38%	6.12h	2310	36.05%	9.33h	850	**43.20%**	**3.50h**	**750**
HARBox [16]	69.24%	10.06h	440	66.87%	3.06h	840	68.56%	3.59h	260	**69.40%**	**2.16h**	**380**
StackOverflow [17]	36.92	6.66h	280	37.12	2.28h	580	37.45	3.92h	260	**36.38**	**1.39h**	**200**

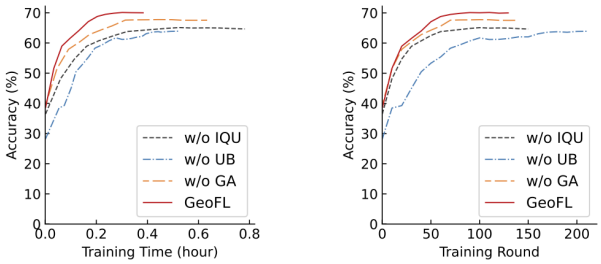
ability to resolve aggregator heterogeneity. Unlike HierFAVG, which treats updates from all aggregators equally each round, and HierFAVG-Async, which continuously aggregates non-importance model updates, as well as Async-HFL that suffers from greater convergence impacts due to larger model interaction delays, GeoFL can aggregate model updates from different aggregators more effectively.

GeoFL achieves the target final accuracy with much fewer communication rounds. Compared with HierFAVG, HierFAVG-Async, and Async-HFL, GeoFL can use the least number of communication rounds to achieve the target final accuracy. Table II shows HierFAVG-Async has a larger number of communication rounds since it has to use its aggregator-server communication bandwidth frequently. For example, HierFAVG-Async on Landmarks-User-160k [14] needs up to 2,310 global communication rounds, which uses more resources than the other three methods. In contrast, GeoFL only requires 750 global communication rounds by constricting the communication between its aggregator and the central server. It reduces communication rounds by 68% and saves precious WAN resources.

Although Async-HFL requires fewer global communication rounds on HARBox [16] compared to GeoFL, it is because it trains their entire local data for 5 epochs with each client update, allowing them to complete faster on relatively small datasets. However, when facing large-scale datasets, even with more global communication rounds than GeoFL, they cannot achieve GeoFL's performance. We attribute such an improvement to GeoFL's importance-aware model uploading, which can reduce unnecessary communication rounds. Compared with the three baselines, GeoFL significantly reduces communication rounds and the reduction ranges from 6% (800 rounds to 750 rounds) to 80% (410 rounds to 80 rounds), which greatly saves the communication cost between the aggregator and the central server.

D. Component-wise Analysis

We break down GeoFL's designs for client-aggregator-server communication and evaluate their effectiveness using OpenImage [13] with a random client selection strategy. Without importance-aware quantized model updates (§IV-A), aggregators only upload model updates without quantization to the central server when reaching non-importance model updates' up-bound. Removing up-bound (§IV-B), model updates from aggregators are only uploaded to the central server when their importance exceeds the adaptive threshold. Without graph-based model aggregation (§IV-C), each aggregator's model update is simply averaged for coordination.



(a) Time-to-accuracy. (b) Round-to-accuracy.

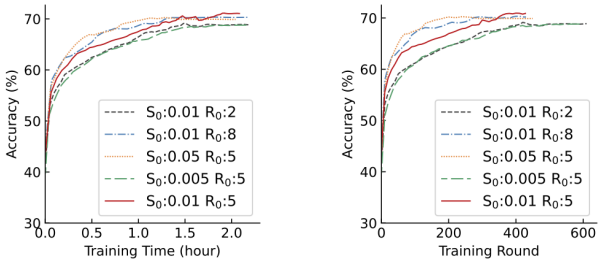
Fig. 8: GeoFL's performance on OpenImage [13] without importance-aware quantized updates (IQU), up-bound (UB), and graph-based aggregation (GA).

Figure 8(a) shows that GeoFL degrades when any component is removed. Without importance-aware quantized updates, aggregators may miss the optimal time to upload important models, resulting in reduced accuracy, and not using quantization can increase communication costs and hinder the speed of model training. Lacking non-importance model updates' up-bound causes the training pace between the aggregator model and the global model to be too different, and the original accuracy cannot be achieved even if more rounds are performed. Furthermore, ignoring aggregator heterogeneity and the correlations among clients can diminish the aggregated model's quality by imbalances, thereby reducing accuracy even with more time. With all components, GeoFL can achieve the best performance.

E. Sensitivity Analysis

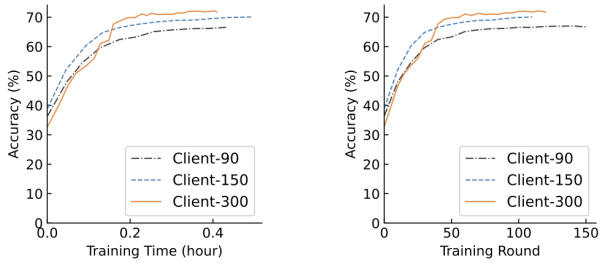
Impact of Parameters: To evaluate GeoFL's several critical parameters, we measure the impact of different values in a certain range. We examine pre-defined initial importance threshold (S_0) and local training up-bound (R_0). Note that the setting of S_0 varies by task and we sweep the importance range of each task to select the appropriate value. We take HARBox [16] as an example to illustrate. Here we choose three different values for S_0 and R_0 to measure respectively as shown in Figure 9. Although the importance threshold will change with time, the importance of model update will be insufficient if S_0 is too small, and if it is too large, it will miss the opportunity to upload the important model in time, thereby reducing the accuracy. It is the same for R_0. Thus we select S_0 to be 0.01 and R_0 to be 5 for HARBox [16].

Impact of Number of Participating Clients: We evaluate the different number of participating clients in each training round for GeoFL's performance. Similarly, we deploy three aggregators and use OpenImage [13] as an example. As we can



(a) Time-to-accuracy. (b) Round-to-accuracy.

Fig. 9: Impact of pre-defined initial importance threshold (S_0) and up-bound (R_0) on HARBox [16].



(a) Time-to-accuracy. (b) Round-to-accuracy.

Fig. 10: Impact of participating clients on OpenImage [13].

see in Figure 10, GeoFL achieves higher final accuracy with more participating clients per round, which is consistent with existing FL framework [20]. With more clients participating (e.g., 300), the data becomes more heterogeneous, requiring more training rounds to learn this diversity, which may slow initial accuracy growth. Nevertheless, involving more participants allows the model updates on the aggregator to better reflect the global data distribution, and GeoFL can leverage graph relationships of client models to optimize the model aggregation on the aggregator and mitigate the negative impacts of aggregator heterogeneity, ultimately achieving higher accuracy (e.g., 71.62%).

VI. Related Work

Efficient FL. Due to its privacy-preserving nature, FL is a highly promising technology in areas such as agriculture [38]–[40], wireless network [41], [42], and advanced sensing [43]–[46]. Efficient FL aims to speed up the global model training by reducing the number of training rounds or the per-round time consumption, corresponding to its computation and communication optimization respectively. Most works focus on the efficiency ML model training, with various designs of loss optimizers [47], aggregation strategies [48], and knowledge distillation [49]. Asynchronous FL can also achieve efficient FL by allowing all clients to keep running based on the results of best-effort communication [11]. Although it avoids stragglers for time efficiency but has to account for convergence analysis [50], resolving the staleness with biased model [51]. Several works also propose to personalize the model efficiently [52]. On the other hand, FL communication optimization focuses on model compression (e.g., sparsification [53], federated dropout [54]) and parameter adaption [55] for communication.

Client Heterogeneity and Selection. The data and system heterogeneity of clients [5] in FL make it distinct from existing well-matured distributed learning paradigms [56], [57]. Clients have Non-IID data samples, especially for mobile scenarios with edge devices [16], [58]. Additionally, clients' devices also demonstrate system heterogeneity in computation resources and communication bandwidth [20], resulting in straggler participants. Guided client selections have been proposed to resolve client heterogeneity [20], [33]. The key is designing a utility function to measure the impact of clients' data and system resources, allowing the coordinator to select "good" clients for the next training round [59]. GeoFL builds on top of the clients and can be incorporated with existing client selection strategies readily.

VII. Conclusion

In this paper, we present GeoFL, a geo-distributed cross-device FL framework based on the hierarchical architecture. By assigning aggregators between clients and the central server as the relay layer, aggregators can consciously determine when to upload the model to the central server based on their WAN bandwidth, allowing for more efficient communication. It further considers the heterogeneity across geo-distributed aggregators, delivering an end-to-end client-aggregator-server architecture for large-scale clients. Compared to the baselines, our results on large-scale real-world datasets show that GeoFL speeds up global model training by $1.4\times$ to $8\times$ and reduces 6% to 80% unnecessary communication rounds between aggregators and the central server to save bandwidth resources across various models and tasks.

Acknowledgement

This study is supported in part by NSF Awards NeTS-2312674 and 2312675.

References

[1] T. Weyand, A. Araujo, B. Cao, and J. Sim, "Google landmarks dataset v2-a large-scale benchmark for instance-level recognition and retrieval," in *Proceedings of the IEEE/CVF conference on CVPR*, 2020.

[2] F. Radenović, A. Iscen, G. Tolias, Y. Avrithis, and O. Chum, "Revisiting oxford and paris: Large-scale image retrieval benchmarking," in *Proceedings of the IEEE conference on CVPR*, 2018.

[3] R. Del Chiaro, A. D. Bagdanov, and A. Del Bimbo, "Noisyart: A dataset for webly-supervised artwork recognition." in *VISIGRAPP*, 2019.

[4] F. D. Protection, "General data protection regulation (gdpr)," *Intersoft Consulting, Accessed in October*, vol. 24, no. 1, 2018.

[5] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *Foundations and Trends® in Machine Learning*, 2021.

[6] T. Zhang, L. Gao, C. He, M. Zhang, B. Krishnamachari, and S. Avestimehr, "Federated learning for internet of things: Applications, challenges, and opportunities," *IEEE IoTM*, 2022.

[7] S. I. Siam, H. Ahn, L. Liu, S. Alam, H. Shen, Z. Cao, N. Shroff, B. Krishnamachari, M. Srivastava, and M. Zhang, "Artificial intelligence of things: A survey," *ACM Transactions on Sensor Networks*, 2024.

[8] R. A. Sater and A. B. Hamza, "A federated learning approach to anomaly detection in smart buildings," *ACM Transactions on Internet of Things*, vol. 2, no. 4, pp. 1–23, 2021.

[9] B. Chen, N. Ivanov, G. Wang, and Q. Yan, "Dynamicfl: Balancing communication dynamics and client manipulation for federated learning," in *2023 20th Annual IEEE International Conference on SECON*, 2023.

[10] L. Liu, J. Zhang, S. Song, and K. B. Letaief, "Client-edge-cloud hierarchical federated learning," in *Proceedings of IEEE International Conference on Communications (ICC)*. IEEE, 2020.

[11] X. Yu, L. Cherkasova, H. Vardhan, Q. Zhao, E. Ekaireb, X. Zhang, A. Mazumdar, and T. Rosing, "Async-hfl: Efficient and robust asynchronous federated learning in hierarchical iot networks," *arXiv preprint arXiv:2301.06646*, 2023.

[12] F. Lai, Y. Dai, S. S. Singapuram, J. Liu, X. Zhu, H. V. Madhyastha, and M. Chowdhury, "Fedscale: Benchmarking model and system performance of federated learning at scale," in *ICML*, 2022.

[13] A. Kuznetsova, H. Rom, N. Alldrin, J. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Malloci *et al.*, "The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale," *arXiv preprint arXiv:1811.00982*, 2018.

[14] T.-M. H. Hsu, H. Qi, and M. Brown, "Federated visual classification with real-world data distribution," in *European Conference on Computer Vision*. Springer, 2020, pp. 76–92.

[15] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *arXiv preprint arXiv:1804.03209*, 2018.

[16] X. Ouyang, Z. Xie, J. Zhou, J. Huang, and G. Xing, "ClusterFL: A Similarity-Aware Federated Learning System for Human Activity Recognition," in *Proceedings of ACM MobiSys*, 2021.

[17] S. O. Data., "Bigquery public datasets," in *https://cloud.google.com/bigquery/public-data*, Retrieved by May 20th 2022.

[18] ESnet and L. B. N. Laboratory, "iperf3q," in *http://software.es.net/iperf/*, Accessed on May 20th, 2022.

[19] "Mobiperf," in *https://www.measurementlab.net/tests/mobiperf/*, Accessed on May 20th, 2022.

[20] F. Lai, X. Zhu, H. V. Madhyastha, and M. Chowdhury, "Oort: Efficient Federated Learning via Guided Participant Selection," in *Proceedings of USENIX OSDI*, 2021.

[21] A. Ignatov, R. Timofte, A. Kulik, S. Yang, K. Wang, F. Baum, M. Wu, L. Xu, and L. Van Gool, "Ai benchmark: All about deep learning on smartphones in 2019," in *Proceedings of ICCVW*, 2019.

[22] J. H. Ro, T. Breiner, L. McConnaughey, M. Chen, A. T. Suresh, S. Kumar, and R. Mathews, "Scaling language model size in cross-device federated learning," *arXiv preprint arXiv:2204.09715*, 2022.

[23] A. Katharopoulos and F. Fleuret, "Not all samples are created equal: Deep learning with importance sampling," in *Proceedings of ICML*. PMLR, 2018.

[24] G. Alain, A. Lamb, C. Sankar, A. Courville, and Y. Bengio, "Variance Reduction in SGD by Distributed Importance Sampling," *arXiv:1511.06481 [cs, stat]*, 2016.

[25] P. Zhao and T. Zhang, "Stochastic optimization with importance sampling for regularized loss minimization," in *Proceedings of ICML*. PMLR, 2015.

[26] X. Xu, S. Duan, J. Zhang, Y. Luo, and D. Zhang, "Optimizing federated learning on device heterogeneity with a sampling strategy," in *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*. IEEE, 2021, pp. 1–10.

[27] A. Reisizadeh, A. Mokhtari, H. Hassani, A. Jadbabaie, and R. Pedarsani, "Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 2021–2031.

[28] "Core ml tools," in *https://coremltools.readme.io/docs/quantization/*, Accessed on February 27th, 2023.

[29] J. So, K. Hsieh, B. Arzani, S. Noghabi, S. Avestimehr, and R. Chandra, "Fedspace: An efficient federated learning framework at satellites and ground stations," *arXiv preprint arXiv:2202.01267*, 2022.

[30] S. Chen, G. Long, T. Shen, and J. Jiang, "Prompt federated learning for weather forecasting: Toward foundation models on meteorological data," *arXiv preprint arXiv:2301.09152*, 2023.

[31] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[32] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[33] C. Li, X. Zeng, M. Zhang, and Z. Cao, "Pyramidfl: Fine-grained data and system heterogeneity-aware client selection for efficient federated learning," in *Proceedings of ACM MobiCom*, 2022.

[34] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE/CVF CVPR*, 2018.

[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of IEEE/CVF CVPR*, 2016.

[36] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 6105–6114.

[37] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," *arXiv preprint arXiv:1909.11942*, 2019.

[38] M. Gan, Y. Liu, L. Liu, C. Wu, Y. Dong, H. Zeng, and Z. Cao, "Poster: mmleaf: Versatile leaf wetness detection via mmwave sensing," in *Proceedings of ACM MobiSys*, 2023.

[39] Y. Liu, M. Gan, H. Zeng, L. Liu, Y. Dong, and Z. Cao, "Hydra: Accurate multi-modal leaf wetness sensing with mm-wave and camera fusion," in *Proceedings of ACM MobiCom*, 2024.

[40] J. Wang, Y. Feng, G. Kumbhar, G. Wang, Q. Yan, Q. Jin, R. C. Ferrier, J. Xiong, and T. Li, "Soilcares: Towards low-cost soil macronutrients and moisture monitoring using rf-vnir sensing," in *Proceedings of ACM MobiSys*, 2024.

[41] C. Li, Y. Ren, S. Tong, S. I. Siam, M. Zhang, J. Wang, Y. Liu, and Z. Cao, "Chirptransformer: Versatile lora encoding for low-power wide-area iot," in *Proceedings of ACM MobiSys*, 2024.

[42] J. Du, Y. Liu, Y. Ren, L. Liu, and Z. Cao, "Loratrimmer: Optimal energy condensation with chirp trimming for lora weak signal decoding," in *Proceedings of ACM MobiCom*, 2024.

[43] R. Wang, Y. Liu, and R. Müller, "Detection of passageways in natural foliage using biomimetic sonar," *Bioinspiration & Biomimetics*, vol. 17, no. 5, p. 056009, 2022.

[44] G. Li, H. Zeng, H. Guo, Y. Ren, A. Dixon, Z. Cao, and T. Li, "Piezobud: A piezo-aided secure earbud with practical speaker authentication," in *Proceedings of ACM SenSys*, 2024.

[45] H. Zeng, G. Li, and T. Li, "Pyrosense: 3d posture reconstruction using pyroelectric infrared sensing," *Proceedings of the ACM on IMWUT*, vol. 7, no. 4, pp. 1–32, 2024.

[46] S. Zhang, Q. Wang, M. Gan, Z. Cao, and H. Zeng, "Radsee: See your handwriting through walls using fmcw radar."

[47] S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečnỳ, S. Kumar, and H. B. McMahan, "Adaptive federated optimization," in *Proceedings of ICLR*, 2021.

[48] U. Michieli and M. Ozay, "Are all users treated fairly in federated learning systems?" in *Proceedings of the IEEE/CVF CVPR*, 2021.

[49] Z. Zhu, J. Hong, and J. Zhou, "Data-free knowledge distillation for heterogeneous federated learning," in *Proceedings of International Conference on Machine Learning (ICML)*. PMLR, 2021.

[50] L. Zhu, H. Lin, Y. Lu *et al.*, "Delayed gradient averaging: Tolerate the communication latency for federated learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 29 995–30 007, 2021.

[51] Z. Chai, Y. Chen, L. Zhao, Y. Cheng, and H. Rangwala, "Fedat: A communication-efficient federated learning method with asynchronous tiers under non-iid data," *ArXivorg*, 2020.

[52] A. Li, J. Sun, X. Zeng, M. Zhang *et al.*, "Fedmask: Joint computation and communication-efficient personalized federated learning via heterogeneous masking," in *Proceedings of ACM SenSys*, 2021.

[53] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[54] N. Bouacida, J. Hou, H. Zang, and X. Liu, "Adaptive federated dropout: Improving communication efficiency and generalization for federated learning," in *Proceedings of IEEE INFOCOM Workshops*, 2021.

[55] K. Mishchenko, G. Malinovsky, S. Stich, and P. Richtárik, "Proxskip: Yes! local gradient steps provably lead to communication acceleration! finally!" *arXiv preprint arXiv:2202.09357*, 2022.

[56] K. Hsieh, A. Harlap, N. Vijaykumar, D. Konomis, G. R. Ganger, P. B. Gibbons, and O. Mutlu, "Gaia: Geo-distributed machine learning approaching {LAN} speeds," in *Proceedings of USENIX NSDI*, 2017.

[57] X. Zeng, M. Yan, and M. Zhang, "Mercury: Efficient On-Device Distributed DNN Training via Stochastic Importance Sampling," in *Proceedings of ACM SenSys*, 2021.

[58] H. Wang, D. Eklund, A. Oprea, and S. Raza, "Fl4iot: Iot device fingerprinting and identification using federated learning," *ACM Transactions on Internet of Things*, vol. 4, no. 3, pp. 1–24, 2023.

[59] L. Fu, H. Zhang, G. Gao, M. Zhang, and X. Liu, "Client selection in federated learning: Principles, challenges, and opportunities," *IEEE Internet of Things Journal*, 2023.