Hierarchical Federated Learning in Device-to-Device Networks with Learning-Topology Co-Optimization

Mengfan Wu, *Student Member, IEEE*, Mate Boban, *Senior Member, IEEE*, and Falko Dressler, *Fellow, IEEE*

Abstract—Federated learning (FL) enables collaborative model training across distributed devices while preserving privacy. However, growing heterogeneity in device resources and communication links challenges conventional FL, especially when relying on a single central server. Hierarchical federated learning (HFL) mitigates these issues by organizing devices into clusters coordinated through intermediate aggregators. Yet, the effectiveness of HFL critically depends on how clusters are formed: intra-cluster communication must be efficient, device computational capacities should be balanced to reduce stragglers, and data heterogeneity must be managed to ensure stable convergence. In this work, we propose a learning–topology co-optimization framework for HFL in networks where nodes communicate with each other with links of varying quality (e.g., device-to-device (D2D) or mesh networks). Our method jointly optimizes device connection topology and learning directions, leading to communication-efficient clusters that remain well aligned in optimization space. We provide a convergence analysis under mild assumptions, showing how inter- and intra-cluster divergence affect learning stability. Extensive experiments demonstrate that our approach consistently improves HFL performance, yielding at least a 6% accuracy gain under unbalanced data distributions and over 16% reduction in training time for regression tasks compared with existing clustering algorithms.

Index Terms—Hierarchical federated learning, resource optimization, wireless communications

1 Introduction

As machine learning applications advance toward deployment at the network edge, training models over data distributed across heterogeneous devices poses significant challenges in terms of communication efficiency, computation scalability, and data governance. Federated learning (FL) is a distributed machine learning paradigm that enables cooperative model training by leveraging data and computational resources from edge devices, alleviating the burden on a central server. By exchanging model updates instead of raw data, FL can reduce communication overhead [1] and limit direct data exposure, which is often desirable in privacy-sensitive applications. In FL, model training is performed locally and periodically coordinated through model aggregation, making it particularly suitable for large-scale, resource-constrained, and geographically distributed environments.

- *M. Wu (Corresponding Author) is with Munich Research Center, Huawei Technologies Duesseldorf GmbH, Munich, Germany as well as with the School of Electrical Engineering and Computer Science, TU Berlin, Berlin, Germany. E-mail: mengfan.wu@huawei.com.*
- *M. Boban is with Munich Research Center, Huawei Technologies Duesseldorf GmbH, Munich, Germany. E-mail: mate.boban@huawei.com.*
- *F. Dressler is with the School of Electrical Engineering and Computer Science, TU Berlin, Berlin, Germany. E-mail: dressler@ccs-labs.org.*
- *This work was supported in part by the Federal Ministry of Education and Research (BMBF, Germany) within the 6G Research and Innovation Cluster 6G-RIC under Grant 16KISK020K and within the framework of the HORIZON-JU-SNS-2022 project TIMES, co-funded by the European Union. The views expressed are those of the authors and do not necessarily represent the project.*

However, practical deployments must cope with substantial system heterogeneity, including variations in device computation capability, communication bandwidth, and connectivity conditions. In traditional synchronized FL, where a central server collects model updates from all participants, heterogeneity in uplink quality across devices can lead to the straggler effect [2] while data heterogeneity across clients can cause unstable convergence after model aggregation. These challenges motivate the development of layered and topology-aware FL architectures that exploit local coordination to reduce global communication overhead and mitigate divergence caused by heterogeneous data. Two representative directions are clustered FL and hierarchical federated learning (HFL) [3]. Clustered FL primarily targets data heterogeneity by partitioning clients into groups and maintaining cluster-specific models, often coordinated by a central server [4]–[7]. In contrast, HFL introduces an explicit hierarchy in which intermediate aggregation nodes coordinate subsets of clients before a global aggregation step. By retaining global synchronization across clusters, HFL maintains a globally applicable model that reflects contributions from all clients while alleviating communication bottlenecks caused by direct client-to-server interactions. This property is particularly relevant in bandwidth-constrained peer-to-peer, mesh, and device-to-device (D2D) networks, where direct connectivity between devices enables efficient local coordination.

Despite these advances, an open challenge remains: **how to form clusters in HFL that jointly balance communication efficiency, computational heterogeneity, and learning-direction alignment**. Existing approaches often optimize

only a subset of these factors, for example, focusing on communication latency or data similarity alone, or archaically assume static network conditions. In dynamic D2D environments, however, link quality and device availability can vary significantly, making cluster formation a combinatorial optimization problem with coupled system and learning objectives.

On the smaller scale of HFL, synchronization within a cluster is eased but not guaranteed. Even in settings with stable client-to-edge communication, heterogeneity in computational capacity can lead to intra-cluster straggler effects, resulting in idle time for faster devices. Balancing the compound round time, which includes both computation and communication, is therefore critical to improving training efficiency [6]. Additionally, clusters formed purely based on communication proximity may not align with the global learning objective if their data distributions diverge significantly [8]. Since HFL typically involves multiple intra-cluster updates before global aggregation, such divergence can accumulate and hinder global convergence. Consequently, effective cluster formation must carefully balance system efficiency with learning-direction consistency.

In this work, we study HFL under a D2D[1] setting, where communication conditions are inherently variable. Building on our previous work [9], which mitigates heterogeneity through adaptive training and upload scheduling, we instead actively exploit heterogeneity by forming clusters that are optimized for communication, computation, and learning objectives. In our system, the server determines cluster configurations based on devices' computation capabilities, communication conditions, and proxy representations of their learning directions, while cluster models are periodically synchronized through a global aggregation step.

Our key contributions can be summarized as follows:

1) We propose a co-optimization clustering algorithm for HFL in D2D networks, jointly accounting for devices' computation and communication resource constraints as well as their learning directions.
2) We provide a convergence analysis, deriving a bound on the averaged norm of gradients that explicitly characterize the impact of inter-cluster and intra-cluster divergence.
3) We investigate the use of substitute features to represent learning directions, with improvements on privacy preservation and communication/computation efficiency.
4) We empirically validate the need of diversifying learning directions within clusters and optimizing intra-cluster link configurations. Comparisons against two strong benchmarks highlight the effectiveness of our proposed solution.

2 Related Work

2.1 Data Heterogeneity

Data heterogeneity in FL originates from how the scenario of FL is defined: learning a machine learning model with data distributed/collected on local devices. The type / class of data is correlated with the type and location of the FL devices. Therefore, for FL encompassing a wide range of devices and application scenarios, data heterogeneity poses challenge to the learning efficiency and applicability of FL.

Tackling data heterogeneity: frequent synchronization is the way to deal with data heterogeneity as indicated in the convergence analysis of [10]. However, for the case where communication resources between FL devices and the server are limited, frequent synchronization needs to be avoided and techniques such as HFL [3] were developed to address the issue. The motivation for this is to enable frequent synchronization between clients by utilizing the locally-viable links, e.g., between the FL devices and an edge server. In other cases such as decentralized FL, where there is no server collecting model information from FL devices, Li et al. [11] estimate global moment of gradient and apply it into the local updates to tackle the inconsistency caused by data heterogeneity. In asynchronous FL where the effect of data heterogeneity is convoluted with the effect of heterogeneous computation and communication resources, Wu et al. [9] derived a formula for computing the aggregation weights based on model differences so that the disturbance from models trained on heterogeneous data is controlled.

Moreover, optimization-based techniques such as Sharpness-Aware Minimization (SAM) [12] have also been adapted to FL [13], [14], where encouraging flatter minima makes the aggregated model more robust to client heterogeneity by mitigating the effects of client-specific sharp optima. Other FL methods seek to transcend naive averaging to mitigate the effect of data heterogeneity in FL. For example, FedCross [15] introduces a multi-model cross-aggregation scheme, where multiple global models are maintained and intelligently fused to guide optimization away from client-specific sharp minima and toward a generalizable solution. In a more radical way, FedMR [16] replaces aggregation with a layer-wise model recombination strategy, heuristically assembling new models from client layers to discover more robust and generalizable architectures. In this paper, we carefully adapt HFL scheme to construct clusters with frequent synchronization and data distribution similar to the global level.

Mathematic representation of data heterogeneity: difference of two model gradients are usually adopted as an implicit way to manifest the effect of data heterogeneity in FL. The most common way is to bound the difference between local model gradients and the global model gradient by a constant value, e.g., in [17], [18]. Another way is to limit the ratio between the scale of local model gradients and global model gradients, e.g., in [19], [20]. More direct representations of data heterogeneity can be found in [21], [22], where the difference of class distribution vector is an accurate measurement of how different data is distributed between two FL clients.

A more practical way of measuring model / data heterogeneity is to use the cosine similarity value of two representative vectors. In [23], cosine similarity between clients' gradients is used to adapt the aggregation weight in decentralized FL. In [24], cosine similarity is used to judge if client updates are aligned with the global learning direction, with the global gradients computed at the central server

1. Without loss of generality, in addition to D2D networks, our solution applies to any network where nodes communicate with each other (either directly or indirectly, e.g., mesh networks, Vehicle-to-Vehicle (V2V) networks, etc.) over links of variable quality.

with a proxy dataset. Based on [24], Zhang et al. [25] use cosine similarity between devices' gradients to determine the model aggregation weight in decentralized FL. Similary, we adopt cosine similarity of (gradient-based) feature to measure data heterogeneity, while also investigated the applicability of other metrics in [21], [22].

2.2 Computation and Communication Heterogeneity

Other aspects of heterogeneity in FL in addition to data distribution include computational and communication resources of learning devices. The two factors jointly result in different round time of FL devices if the same amount of computations is demanded across FL participants. A more colloquial way of describing such effect is to refer the devices of longer round time as stragglers. Common approaches to address the straggler issue include refined resource allocation strategies, allowing uneven/asynchronous contribution of learning devices, and client selection/clustering approaches.

Resource allocation approach: client selection strategies often integrate resource allocation algorithms, as client without allocated resources are excluded from participation. A notable work on resource adaptation is FedCS [26], which allocates communication resources based on computational time, prioritizing faster devices to enhance efficiency and include more participants. This idea was extended in [27] to jointly optimize bandwidth and energy consumption, aiming to maximize the data available for training. More recently, environment factors have also been incorporated into resource allocation scheme, such as [28], where trajectories of UAVs, which serve as intermediate parameter servers, are jointly optimized with device scheduling and round duration.

Model-centric approaches: the manipulation of clients' and global models can be adapted to suit diverse client capabilities, e.g., enabling clients' personalized participation to meet their resource limitations. For instance, HeteroFL [29] and DepthFL [30] generate families of sub-models of varying width and depth, respectively, allowing clients to train on a model suited to their computational resources. Futher dynamic adaptation is achieved by CoCoFL [31] and FlexFL [32], which generate personalized model variants via layer freezing/quantization and filter pruning, respectively. Though their techniques differ, with pruning improving sparsity while freezing/quantization reducing precision, both demonstrate that adapting the model to the client environment is a powerful alternative to selecting clients or allocating resources. This paradigm turns system heterogeneity into an opportunity for efficient and inclusive training, a principle also reflected in adaptive compression techniques [33] and adaptive training scheduling [34].

Easing synchronization: to accommodate different client round time, FL systems have evolved to semi-synchronous and fully asynchronous paradigms as a resource-independent solution. Semi-synchronized FL, which can be time-based or buffer-based, relax but do not eliminate synchronization. In time-based systems, [35]–[37], the server sets a global round time for model aggregation and accepts model updates that arrived within a time window. In buffer-based systems [38], [39], aggregation is triggered once the buffer of model updates reaches its capacity. Recent advancements enhance this buffer-based approach by intelligently curating its contents. For instance, CaBaFL [40] introduces a hierarchical cache system where updates are promoted to a primary aggregation buffer based on their feature distribution similarity to the global model and their local training completeness, ensuring only the most relevant and well-trained updates are aggregated. This spectrum of easing synchronization requirements culminates in fully asynchronous FL, where the buffer size is reduced to one [9], [41]–[43]. In this paradigm, the global model is updated immediately upon receiving a single client update, maximizing agility at the potential cost of update stability.

Constructing layered structure of clients: HFL and clustered FL eases synchronization constraints by organizing devices into subgroups based on their resource profiles. For instance, clustered FL frameworks explicitly group devices with similar computational capabilities and communication latencies [6], effectively homogenizing round times within each cluster to minimize stragglers and improve training efficiency. This principle of resource-aware grouping extends to networks with physical constraints. In wireless edge networks, devices are clustered by their association to proximal base stations [44], a strategy that inherently groups devices with comparable channel conditions and communication reliability. Similarly, in vehicular networks, clustering leverages geographical proximity to ensure stable, low-latency connections among members [45]. By creating subgroups where computation and communication heterogeneity is minimized, these structured hierarchies make synchronous aggregation feasible in otherwise prohibitive, dynamic environments.

2.3 Modeling Heterogeneity in HFL

Previous FL work with layered structures of clients have mainly been limited in terms of modeling heterogeneity.

Assuming easy synchronization on smaller scale: in the introduction, we noted that clustered FL [4]–[7] typically focuses on addressing data heterogeneity while assuming guaranteed communication between clients and server. In works such as [4], [5], [7], communication and computation heterogeneity is implicitly captured through client availability during the selection process. The earliest HFL framework [3] assumes each cluster of clients is connected to and managed by a dedicated edge server, with strict synchronization required both within clusters and across the server hierarchy. In this setup, heterogeneity in communication and computation resources is not explicitly considered. A similar assumption is made in [46], which incorporates optimization momentum into both local training and server aggregation; Yang et al. [47] extend the personalized FL framework pFedHN [48] to an HFL setting, where in each intra-cluster round only one client is sampled. Here, sampling probabilities can be interpreted as reflecting the heterogeneity of device resources.

Partial explicit modeling: Wang et al. [49] account for data heterogeneity (in terms of sample counts per client), but assume uniform steady communication links within clusters and from cluster to the server. Cluster-level heterogeneity in round times is addressed by asynchronous

cluster–server aggregation. Similarly, Wang et al. [50] study a two-layer FL system where edge servers are interconnected in a decentralized topology, while the links between clients and edge servers are simplified to probabilistic client selection. Zhang et al. [6] simultaneously model heterogeneity in data distribution, computation, and communication resources, with link quality to the server represented as a distance-dependent function, which is independent of cluster composition.

Embedded in optimization conditions: optimization techniques well investigated in synchronized FL [27] are also applied in HFL settings, e.g., [44], [51]–[55]. In this line, communication heterogeneity is reflected in the optimization constraints, ensuring timely device participation and promoting faster convergence. Moreover, Pervej et al. [44] and Su et al. [55] also incorporate computational heterogeneity: Pervej et al. [44] optimize device pruning ratios to accelerate local updates, while Su et al. [55] design an online intra-cluster client selection algorithm that minimizes accumulated round time.

2.4 Cluster Forming Criteria

Location-based clustering: Yang et al. [46], Wang et al. [50], and Zhang et al. [51] assume fixed cluster components, with [50], [51] allowing client selection performed inside each cluster associated with a fixed edge server. Pervej et al. [44] assume fixed cluster configuration which is determined by the corresponding base station based on devices' computational power and battery. Su et al. [55] model the members associated with each edge server to be time-varying due to devices' mobility. A more detailed model of clustering clients based on mobility can be found in [45], where a Markov chain is used to model the probability of a learning device moving to neighboring clusters.

Data-based clustering: in [22], client model updates are send to central server with virtual clusters formed to achieve personalization of models. Clients with similar categorical data distributions are grouped together using Hellinger distance [56] based on encrypted distribution vector. In [21], a swap-based algorithm is applied at the beginning of the training stage to create clusters that has similar data distribution to the global dataset.

Efficiency-based clustering: among other works in which cluster configuration are changeable, Luo et al. [52] use a swap-based optimization algorithm to minimize the energy consumption and learning round time of HFL, with all formed clusters still using the same cluster round time. Liu et al. [53] and Abdellatif et al. [57] optimize learning clients' association with edge servers based on spectrum allocation to minimize latency and on data distribution to minimize the difference of cluster data and the global data. Early work in HFL [49] sets only the number of clusters as variable so that the corresponding cluster configuration conforms to the constraints of resources. FedMDS [6] clusters clients into groups based on similarity of data and individual round time.

In the following section, we showcase how efficiency and data balance is achieved simultaneously in clusters of HFL.

Notation	Definition
$f(\mathbf{x}, d)$	loss function for unified learning task w.r.t. model $\mathbf{x}$ and sampled data d
F	global loss function
N	number of participating clients
D_i	local dataset of client i
P	number of global learning rounds
r_c	number of intra-cluster learning rounds of cluster c
C_c	clients assigned to cluster c
D_c	combined dataset of cluster c
$\mathbf{x}^p$	global model of the p-th round
$\mathbf{x}_{c,i}^{s,t}$	model of client i in cluster c's s-th learning round, with t steps of local optimization
F_c, F_i	loss function of cluster c and client i
$g_i(\mathbf{x})$	stochastic gradient of client i based on model $\mathbf{x}$ and sampled local data
γ	learning rate of local stochastic gradient descent (SGD)
η	upper bound of sampling uncertainty in local SGD
$\epsilon_{i,c}, \epsilon_{c,g}$	upper bound of intra-cluster / cluster-global gradient divergence
ρ_c	aggregation weight of cluster c in global aggregation
π_i	aggregation weight of client i in intra-cluster aggregation

TABLE 1: Variables, parameters, and acronyms used in theoretical analysis and experiments

3 System Design

In this section, we start by formulating the learning objective by stating the target loss function and the hierarchical aggregation scheme of the learning platform. We then detail the setup of the computation and communication conditions of the participating devices in the HFL scenario. Based on the device setup, we further elaborate on the motivation and workflow of the optimization algorithm for clustering device in HFL. The variables used in the descriptions of the learning objectives and convergence analysis are listed in Table 1, while the variables and acronyms specific to the algorithms are provided in Table 2.

3.1 Learning Objective

We consider a HFL system with N participating devices, each with the local learning target defined as following for client i: $\mathbf{x}_i^* = \arg\min_{\mathbf{x}} F_i(\mathbf{x}) \coloneqq \mathbb{E}_{d\sim D_i} f(\mathbf{x}, d)$. Based on this, we defined the global optimization target as to obtain:

$$\mathbf{x}_g^* = \arg\min_{\mathbf{x}} F(\mathbf{x}) \coloneqq \mathbb{E}_{d\sim D} f(\mathbf{x}, d) = \sum_{i\in\{N\}} \frac{|D_i|}{|D|} F_i(\mathbf{x}), \tag{1}$$

and similarly per-cluster optimization target as:

$$\mathbf{x}_c^* = \arg\min_{\mathbf{x}} F_c(\mathbf{x}) \coloneqq \mathbb{E}_{d\sim D_c} f(\mathbf{x}, d) = \sum_{i\in C_c} \frac{|D_i|}{|D_c|} F_i(\mathbf{x}), \tag{2}$$

where $D = \bigcup_{i\in\{N\}} D_i$ and $D_c = \bigcup_{i\in C_c} D_i$.

Given time-varying D2D connections, we aim to find optimal configurations of clusters periodically and performs local optimization, intra-cluster aggregation, and inter-cluster aggregation in HFL. The final goal is to obtain a model $\mathbf{x}$ after the global learning time that achieves highest testing accuracy or lowest loss as described in Equation (1).

In the following algorithms and experiments, model aggregation is weighted by data proportion. Specifically, the intra-cluster and inter-cluster aggregation weights are defined as $\pi_i = |D_i|/|D_c|$, $\forall i \in C_c$, and $\rho_c = |D_c|/|D|$, $\forall c$. These weights—-π_i, ρ_c, and the contributions of F_i and F_c to F—-are not rigid; they can be customized to reflect different notions of client/cluster importance. In this work, we adopt data-proportional weighting to align with the global evaluation metric, which averages errors across all samples. Moreover, since clusters are temporary and reconfigurable, no fixed weight is tied to a specific cluster, as its members may change across rounds.

3.2 Learning Process

In our HFL system, we assume each learning device is connected to some peer learning devices in the system. The D2D connection stays steady (in terms of the throughput) for a global round time, then gets reset after each global round. Each learning device is equipped with different computing capacity, and also loaded with locally collected data for training and testing the machine learning model. The system used for this project is inherited from our previous project [9], where learning devices are only able to perform a certain number of local optimizations, or transmit a portion of its model updates to the server/peers. The HFL learning workflow is described in Algorithm 1.

We divide our HFL process into four stages. In the first stage of feature collection, each learning client performs $h_i' \propto D_i$ number of local optimizations and sends the model difference to the server as one of the features for co-optimization. While computing features, clients also evaluate the connection status with their peers. The D2D connection information, and if necessary the optimization speed, are sent to the server as another feature for co-optimization. Since the "true" global gradient is unavailable in FL, we use the vector ϕ as a practical surrogate signal, consistent with prior work [6]. Other possible features are examined in Section 5.4.4 and Section 5.5.3.

In the second stage, the server performs co-optimization based on optimization directions indicated by ϕ_i^p and also the round time in clusters. Details of co-optimization are documented in Section 3.3. Note that we can also substitute the feature of model difference with other numerical features, e.g., label distribution in [21]. Comparison between these features is also shown in the experiments in Section 5.

Between second and third stage, cluster configurations computed by the server are sent to each client. The server also informs clients about their cluster head, to which they should send their model updates. Moreover, the server also customizes the intra-cluster rounds r_c for each cluster c based on the computed round time. Learning clients then perform local optimizations and send their model updates to cluster head periodically for intra-cluster aggregation.

Algorithm 1 HFL with Cluster Optimization

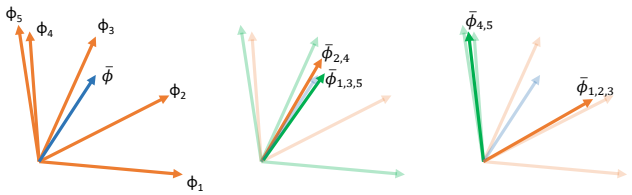
for $p \in \{1, \ldots, P\}$ **do**

Stage 1 *Feature collection*
for client $i \in \{N\}$ **do**
  Do h_i' steps of local optimization:
    $\mathbf{x}_i^{p,t+1} \leftarrow \mathbf{x}_i^{p,t} - \gamma \cdot \nabla f(\mathbf{x}_i^{p,t}, d_i^t), \quad d_t \sim D_i$
  Compute model difference as feature:
    $\phi_i^p = \mathbf{x}_i^{p,h_i'} - \mathbf{x}_i^{p,0}$
  Send ϕ_i^p and D2D connection information to server for cluster assignment
end for

Stage 2 *Co-optimization for cluster configuration*
Server performs co-optimization
Server sends cluster configuration $\mathbf{C}^p$ to clients

Stage 3 *Intra-cluster learning stage*
for cluster $c \in \mathbf{C}^p$ **do**
  $r_c \leftarrow \lfloor T_g/T_c \rfloor$
  $x_{c,i}^{0,0} \leftarrow \mathbf{x}_i^{p,k_f}, \forall i \in C_c$
  for $s \in \{0, \ldots, r_c - 1\}$ **do**
    Each client $i \in C_c$ performs h_i optimizations:
      $\mathbf{x}_{c,i}^{s,t+1} \leftarrow \mathbf{x}_{c,i}^{s,t} - \gamma \cdot \nabla f(\mathbf{x}_{c,i}^{s,t}, d_i^t), \quad d_t \sim D_i$
    Each client i sends $\mathbf{x}_{c,i}^{s,k_l}$ to the cluster head
    Cluster head aggregates models:
      $\mathbf{x}_c^{s+1} \leftarrow \sum_{i \in C_c} \pi_i \mathbf{x}_{c,i}^{s,h_i}$
  end for
  Cluster head sends $\mathbf{x}_c^{r_c}$ to server
end for

Stage 4 *Inter-cluster aggregation*
Server performs inter-cluster aggregation:
  $\mathbf{x}^{p+1} \leftarrow \sum_{c \in \mathbf{C}^p} \rho_c \mathbf{x}_c^{r_c}$
Server sends $\mathbf{x}^{p+1}$ to each client
end for



(a) Individual directions **(b)** Optimized cluster **(c)** Random cluster

Fig. 1: Individual and Clustered Optimization Direction.

After reaching the global round time, the system enters the fourth stage where each cluster head sends the aggregated cluster model to the server for inter-cluster aggregation. The aggregated global model is then sent back to clients directly or indirectly via cluster head.

3.3 Co-optimization Algorithm

3.3.1 *Motivation for Co-optimization*

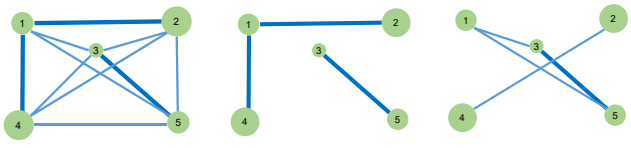
Optimization Divergence: Consider that at a certain step in the HFL process illustrated in Figure 1a with five learning

devices participating. The optimization direction of each client is different due to heterogeneous data distribution across clients. The global optimization direction $\bar{\phi}$ is the average of $\phi_i, \forall i = 1, \ldots, 5$.

Assume that the optimization of each device in the following rounds generally follows the illustrated direction, we modify the cluster formation and illustrate the corresponding optimization direction in each cluster Figures 1b and 1c. When clusters are carefully designed, their optimization direction can align with the global optimization direction $\bar{\phi}$ as shown in Figure 1b. However, if clusters are randomly constructed, there is a risk of significant deviation between the cluster optimization direction ϕ_c and global direction $\bar{\phi}$, as illustrated in Figure 1c. In HFL multiple rounds of intra-cluster learning and aggregation occur between each inter-cluster aggregation. As a result, any divergence between the cluster optimization direction and the global direction increases the risk of sub-optimal inter-cluster aggregation outcomes.

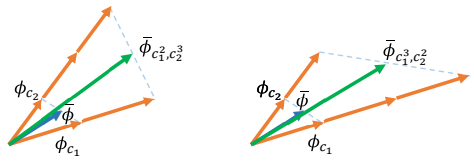
Improving Cluster Learning Round Time: We consider the case where the cluster members send their model updates to the head of the cluster, which also has computational capacity. The head of the cluster performs model aggregation only after receiving model updates from all of the cluster members, and the head of cluster also completes its own training. The round time of a single device is the summation of its computational time and the time of sending model to the head of the cluster. The learning round time of a cluster is the maximum of the round time of its members. Suppose in HFL, the frequency of inter-cluster synchronization is stipulated by the central server and each cluster is allowed to perform as many rounds of learning as possible within the global round time, we then aim to construct clusters so that the idle time of devices are minimized and clusters perform more rounds of training.

For illustration, the 5 learning devices are interconnected with each other in Figure 2a, with the thickness of a line representing the throughput of the corresponding connection and the size of node denoting its computational capacity. For simplicity, we assume that the devices have local dataset of the same size, which leads to their computational time inversely proportional to the computational capacity. In this case, creating cluster $\{1, 2, 4\}$ and $\{3, 5\}$ as in Figure 2b is the best configuration to achieve short learning round time in both clusters, with fast workers device 2 and device 4 connected to device 1 via high-throughput link, and slow workers device 3 and device 5 inter-connected with high-throughput link. Other configurations, e.g., $\{1, 3, 5\}$ and $\{2, 4\}$ in Figure 2c would result in longer round time in both clusters due to the bottleneck link $(1-3, 2-4)$.



(a) Connection topology **(b)** Clusters with fast connection **(c)** Clusters with slow connection

Fig. 2: Devices with heterogeneous direct connections.



(a) Unbalanced cluster learning direction w.r.t. cluster rounds. **(b)** Balanced cluster learning direction w.r.t. cluster rounds.

Fig. 3: Compounded Impacts of Cluster Learning Direction and Intra-Cluster Rounds.

Controlling Optimization Divergence w.r.t. Round Time: Clusters with shorter round time, which complete more local learning and aggregations within a global round, should align more closely with the global optimization direction. We illustrate two contrasting cases in Figure 3, where ϕ_{c_1} and ϕ_{c_2} represent the learning directions of two clusters c_1 and c_2, with ϕ_{c_1} being more aligned with the global learning direction $\bar{\phi}$. In Figure 3a, the less-aligned cluster c_2 completes more intra-cluster learning rounds, causing its final learning direction $\bar{\phi}_{c_1^2,c_2^3}$ to deviate further from $\bar{\phi}$. Conversely, in Figure 3b, when the biased cluster (c_2) performs fewer local learning rounds and the aligned cluster (c_1) performs more, the aggregated learning direction $\bar{\phi}_{c_1^3,c_2^2}$ deviates less from $\bar{\phi}$.

Except in scenarios where HFL devices collect location-dependent data with location-based D2D connections, most HFL settings show no correlation between data distribution and connection status. As a result, a cluster configuration that minimizes optimization divergence might yield insufficient topologies within clusters, potentially increasing cluster round times. Conversely, optimizing only for connection topology can cause optimization divergence and disrupt the learning process. To address this, we propose co-optimizing both the learning direction divergence and cluster round time, aiming for a balanced cluster configuration that ensures smooth convergence and fast learning progress.

3.3.2 *Workflow of Co-optimization for Clustering Clients*

After collecting the features indicating the optimization direction in Stage 1, the central server adopts a swap search method (Algorithm 2, inspired by [21]). Slightly different from the original algorithm in [21], we relax the limitation on fixed cluster sizes and adopt our devised evaluation metrics of optimization divergence and cluster round time. In each iteration in Algorithm 2, a pair of clusters is chosen and all possible swaps of clients between these two clusters are considered (Case 1 in Algorithm 3). Moreover, since we allow the cluster size to change, Case 2 is also included in the algorithm, where one client is moved from a cluster to another.

The evaluation metrics of the two original clusters and the potential new clusters are also computed and compared. Specifically, the evaluation metrics for the alignment of optimization direction is the angle between the cluster optimization direction and the global direction:

$$\theta_c = \arccos \frac{\phi_c \cdot \bar{\phi}}{\|\phi_c\| \cdot \|\bar{\phi}\|}, \tag{3}$$

Notation	Definition
U_C	number of clusters
$\mathbf{C}^p$	cluster configuration in global round p
n_c	size of cluster c, $n_c = \|C_c\|$
$n_{\min}, n_{\max}$	lower bound and upper bound of the size of cluster
h'_i, h_i	number of local optimization for feature collection and for a round of intra-cluster learning
t'	the computing time of a learning device performing the designated local optimizations
$t_{i,j}$	the time of device i sending model update to device j
m	size of the model updates
$\lambda_{i,j}$	the throughput of the link connecting device i and j
$\phi_i, \phi_c, \bar{\phi}$	client/cluster/global vector feature indicating learning direction
θ_c	angle between direction of cluster c (ϕ_c) and $\bar{\phi}$
T_c, T_g	round time of cluster c, global round time
μ, μ^1, μ^2	evaluation metric, $\mu, \mu^1, \mu^2 \in \{\theta, T, \frac{\theta}{T}, \ldots\}$
z	scale of compromise in co-optimization
n_{swap}	number of swaps allowed in swap search

TABLE 2: Variables, parameters, and acronyms used in algorithms

where $\phi_c = \left(\sum_{i \in C_c} |D_i| \cdot \phi_i\right) / |D_c|$.

For computing the minimum achievable round time for a cluster c, firstly each participating device report their computing time t'_i to complete a designed number of local optimization. Then the time of device i sending model updates to device j: $t''_{i,j} = \frac{m}{\lambda_{i,j}}$ is computed by the server, with $t''_{i,i} = 0$. A matrix representing clients' computing and communication time $\mathbf{T}_c$ can be obtained with entry $t_{i,j} = t'_i + t''_{i,j}$. Then the minimum achievable cluster round time is computed as:

$$T_c = \min\left(\max(\mathbf{T}_{c\{:,1\}}), \ldots, \max(\mathbf{T}_{c\{:,n_c\}})\right), \quad (4)$$

and the head of cluster leading to the shorted round time is:

$$H_c = \arg\min\left(\max(\mathbf{T}_{c\{:,1\}}), \ldots, \max(\mathbf{T}_{c\{:,n_c\}})\right) \quad (5)$$

Based on the computed θ and T and the motivation in Section 3.3.1, we propose a metric $\frac{\phi}{T}$ for cluster formation, where a short round time T leads to more local learning rounds and requires a smaller angular deviation ϕ to ensure smooth learning convergence.

The evaluation metrics of the possible new cluster θ'_{c_1}, θ'_{c_2}, T'_{c_1}, T'_{c_2} are evaluated against the original ones: θ_{c_1}, θ_{c_2}, T_{c_1}, T_{c_2}. If the criteria are satisfied, the corresponding new cluster configuration C'_{c_1}, C'_{c_2} is marked as valid. After iterating through all possible and new clusters, the algorithm either samples one from all the valid candidate pairs, or selects the pair evaluated with best metrics.

3.3.3 Evaluation Criteria in Co-optimization

When comparing multiple metrics for a pair of clusters, there are multiple perspectives regarding the relations between the metrics and the relations between the two clusters. We list the following validation criteria for Algorithm 2:

Algorithm 2 Swap Search to Find Local Optima Solution for Cluster Components

Randomly initialize clusters: $\mathbf{C}$ with $|\mathbf{C}| = U_C$
Compute initial round times per cluster: $\mathbf{T}$
$\bar{\phi} = \left(\sum_{i \in \{N\}} |D_i| \cdot \phi_i\right) / |D|$
Compute learning divergence angle w.r.t. $\bar{\phi}$: $\boldsymbol{\theta}_c$, $\forall c \in \mathbf{C}$
for each trial $t = 1$ to n_{swap} **do**
  Randomly select two clusters: c_1 and c_2
  Compute initial angles: θ_{c_1}, θ_{c_2}, and round time: T_{c_1}, T_{c_2}
  Generate Potential Cluster after Swaps and Moves: $\{(C'_{c_1}, C'_{c_2}), (C''_{c_1}, C''_{c_2}), \ldots\}$
  for each possible cluster configuration C'_{c_1}, C'_{c_2} **do**
    Compute new angles θ'_{c_1}, θ'_{c_2}
    Compute new round time T'_{c_1}, T'_{c_2}
    if evaluation criteria are satisfied **then**
      Mark the swap/move as valid
    end if
  end for
  if valid swap/move found **then**
    randomly choose one pair of valid swap C'_{c_1}, C'_{c_2}
    $C_{c_1} \leftarrow C'_{c_1}, C_{c_2} \leftarrow C'_{c_2}$
    Update H_{c_1} and H_{c_2}
    $T_{c_1} \leftarrow T'_{c_1}, T_{c_2} \leftarrow T'_{c_2}, \theta_{c_1} \leftarrow \theta'_{c_1}, \theta_{c_2} \leftarrow \theta'_{c_2}$
  end if
end for

Algorithm 3 Generate Potential Cluster

$\mathcal{L} \leftarrow \varnothing$
Case 1 *Swap elements between clusters c_1 and c_2:*
for each pair of elements $p_{c_1} \in C_{c_1}$ and $p_{c_2} \in C_{c_2}$ **do**
  $C'_{c_1} = C_{c_1} \setminus \{p_{c_1}\} \cup \{p_{c_2}\}$
  $C'_{c_2} = C_{c_2} \setminus \{p_{c_2}\} \cup \{p_{c_1}\}$
  add (C'_{c_1}, C'_{c_2}) to $\mathcal{L}$
end for

Case 2 *Move one element from c_1 to c_2 (or c_2 to c_1):*
if $|C_{c_1}| > n_{\min}$ and $|C_{c_2}| < n_{\max}$ **then**
  for each element $p_{c_1} \in C_{c_1}$ **do**
    $C'_{c_1} = C_{c_1} \setminus \{p_{c_1}\}$, $C'_{c_2} = C_{c_2} \cup \{p_{c_1}\}$
    add (C'_{c_1}, C'_{c_2}) to $\mathcal{L}$
  end for
end if
return $\mathcal{L}$

Individually or per pair of clusters: evaluating metrics per cluster is to demand each metric evaluated to improve in both cluster:

$$\mu'_c \leq \mu_c, \forall c \in \{c_1, c_2\}, \mu \in \{\theta, T, \frac{\theta}{T}\}$$

Evaluating metrics per cluster pair requires the combined (summed) metric value from both clusters to improve:

$$\sum_{c \in \{c_1, c_2\}} \mu'_c \leq \sum_{c \in \{c_1, c_2\}} \mu_c, \mu \in \{\theta, T, \frac{\theta}{T}\}$$

Trade-off between metrics: allowing one metric μ^1 to deteriorate (increase of α, $\alpha > 0$) to achieve a more sig-

nificant improvement (decrease of $z\alpha$, $z > 1$) on the other metric μ^2:

$$\mu^{1\prime} \leq (1+\alpha)\mu^1 \text{ and } \mu^{2\prime} \leq (1-z\alpha)\mu^2,$$
$$\mu^1 \neq \mu^2, \mu^1 \text{ and } \mu^2 \in \{\theta, T, \frac{\theta}{T}\}$$

Since there are multiple combinations of options, we choose the most representative designs (in terms of complexity and performance) to show in the experiments in Section 5.5.

4 Convergence Analysis

We aim to show the feasibility of our solutions in terms of the evolution of the loss value of the learning function. Firstly, we list assumptions made on the function properties to enable the derivation of the analysis. We then show the bounded averaged gradients based on certain parameter settings to guarantee convergence.

4.1 Assumptions of Function Properties

We made the following assumptions commonly used in the analysis of distributed optimization [20], [38], [41] to facilitate our analysis of the loss evolution of the global model through the learning process.

Assumption 4.1. (L-smoothness) Let function f satisfy L-Lipschitz smooth, so $\exists L > 0$, $\forall \mathbf{x}_1, \mathbf{x}_2 \in$ `model space`

$$\|\nabla f(\mathbf{x}_1) - \nabla f(\mathbf{x}_2)\| \leq L\|\mathbf{x}_1 - \mathbf{x}_2\|$$

There is also an equivalent version for L-Lipschitz smooth property:

$$f(\mathbf{x}_1) - f(\mathbf{x}_2) - \langle \nabla f(\mathbf{x}_2), \mathbf{x}_1 - \mathbf{x}_2 \rangle \leq \frac{L}{2}\|\mathbf{x}_1 - \mathbf{x}_2\|^2$$

We assume all client function F_i and all cluster function F_c to be L-smooth.

Assumption 4.1 is reasonable because smoothness is observable and preserved with operations like addition and affine projection (common operations in the machine learning models adopted in our experiments), and can be tested with techniques as in our previous work [9].

Assumption 4.2. When sampling data points in batch from local dataset to compute $g_i(\mathbf{x})$, the difference between the computed value and the true gradient of the local loss function is bounded:

$$\left\| g_i(\mathbf{x}; \delta) - \nabla_{\mathbf{x}} F_i(\mathbf{x}) \right\|_2^2 \leq \eta^2,$$
$$\forall \mathbf{x} \in \texttt{model space}, \delta \sim D_i, \ \forall i \in \{N\}$$

Assumption 4.2 holds in case of reasonably large batch size B of samples taken in one local gradient optimization.

Assumption 4.3. The gradient of local loss function w.r.t. any model and any batch of sampled data is bounded across all devices:

$$\|g_i(\mathbf{x}; \delta)\|_2^2 \leq G^2, \ \forall \mathbf{x} \in \texttt{model space},$$
$$\forall \delta \sim D_i, \ \forall i \in \{N\}$$

Assumption 4.3 provides a rough estimation of the scale of local gradients and is later proved to form the foundation of bounding the global gradients.

Assumption 4.4. For a cluster indexed with c, with members as C, the difference of its loss gradient and the global loss gradient is bounded:

$$\|\nabla F_c(\mathbf{x}) - \nabla F(\mathbf{x})\| \leq \epsilon_{c,g}$$
$$C \subset \{N\}, \ n_{\min} \leq |C| \leq n_{\max} \tag{6}$$

Assumption 4.5. The difference of the loss gradient of a client function and the loss gradient of its corresponding cluster is bounded:

$$\|\nabla F_i(\mathbf{x}) - \nabla F_c(\mathbf{x})\| \leq \epsilon_{i,c}, \ \forall i \in C_c$$

The scales of $\epsilon_{c,g}$ and $\epsilon_{i,c}$ depend on the specific configurations of clusters in the experiment. Our solutions in Section 3.3 aims to reduce $\epsilon_{c,g}$ in the experiment with the trade-off of increasing $\epsilon_{i,c}$.

4.2 Proof of Convergence

Consider the case where the optimization progress obtained from feature collection stage is discarded, the difference between the global models of two consecutive round $p-1$ and p is denoted as:

$$\mathbf{x}^p = \mathbf{x}^{p-1} - \sum_{c \in C^p} \rho_c \sum_{s=0}^{r_c-1} \sum_{i \in N_c} \pi_i \sum_{t=0}^{h_i-1} \gamma \cdot g_i\left(\mathbf{x}_{c,\,i}^{s,\,t}\right),$$

where for simplicity of notation we set $\mathbf{x}_{c,\,i}^{0,\,0} = \mathbf{x}^{p-1}$, $\forall c \in C^p$, $i \in N_c$. Putting into the global evaluation function F and using the L-Lipschitz smoothness property in Assumption 4.1, we have:

$$\begin{aligned} &F\left(\mathbf{x}^p\right) \leq F\left(\mathbf{x}^{p-1}\right) \\ &+ \left\langle \nabla F\left(\mathbf{x}^{p-1}\right), -\gamma \cdot \sum_{c \in C^p} \rho_c \sum_{s=0}^{r_c-1} \sum_{i \in N_c} \pi_i \sum_{t=0}^{h_i-1} g_i\left(\mathbf{x}_{c,\,i}^{s,\,t}\right) \right\rangle \\ &+ \frac{L}{2} \left\| \gamma \cdot \sum_{c \in C^p} \rho_c \sum_{s=0}^{r_c-1} \sum_{i \in N_c} \pi_i \sum_{t=0}^{h_i-1} g_i\left(\mathbf{x}_{c,\,i}^{s,\,t}\right) \right\|^2 \end{aligned} \tag{7}$$

Decomposing the superposition in the second term of Equation (7), we have:

$$\begin{aligned} &\left\langle \nabla F\left(\mathbf{x}^{p-1}\right), -g_i\left(\mathbf{x}_{c,\,i}^{s,\,t}\right) \right\rangle = \frac{1}{2}\left\| \nabla F\left(\mathbf{x}^{p-1}\right) - g_i\left(\mathbf{x}_{c,\,i}^{s,\,t}\right) \right\|^2 \\ &- \frac{1}{2}\left\| \nabla F\left(\mathbf{x}^{p-1}\right) \right\|^2 - \frac{1}{2}\left\| g_i\left(\mathbf{x}_{c,\,i}^{s,\,t}\right) \right\|^2 \end{aligned} \tag{8}$$

We work on the scale of the difference between the two gradients and assume any cluster function F_c also satisfy L-Lipschitz smooth:

$$\begin{aligned} &\left\| \nabla F\left(\mathbf{x}^{p-1}\right) - g_i\left(\mathbf{x}_{c,\,i}^{s,\,t}\right) \right\|^2 \\ \leq &(t+3)\left[\sum_{\tau=0}^{t-1} L_c^2 \gamma^2 G^2 + \epsilon_{i,c}^2 + \eta^2 \right] \\ &+ (t+3)(s+1)\left[L_c^2 \gamma^2 \sum_{\sigma=0}^{s-1} \sum_{i' \in C_c} \pi_{i'} h_{i'}^2 G^2 + \epsilon_{c,g}^2 \right] \end{aligned} \tag{9}$$

Using recursion and rearrangements of items, we have:

$$\frac{1}{P}\sum_{p=1}^{P}\left\|\nabla F\left(\mathbf{x}^{p-1}\right)\right\|^2 \leq \tag{10}$$
$$\frac{1}{P}\sum_{p=1}^{P}\frac{2}{w^p}\left[F\left(\mathbf{x}^{p-1}\right)-F\left(\mathbf{x}^{p}\right)\right]+\frac{4}{3}G^2$$
$$+\frac{(r_{\max}+1)(h_{\max}+5)\epsilon_{c,g}^2}{4}+\frac{(h_{\max}+5)\left(\epsilon_{c,i}^2+\eta^2\right)}{2}$$

under the condition of $\forall c \in \mathbf{C}^p, i \in C_c, p \in \{1,\ldots,P\}$

$$\gamma \leq \frac{1}{Lh_{\max}} \text{ and } \gamma \leq \frac{\sqrt{6}}{L\left(r_c\sqrt{\overline{h_c^2}(h_i+5)}\right)_{\max}}$$

$\overline{h_c^2}$ is defined as $\sum_{i'\in C_c}\pi_{i'}h_{i'}^2$, the weighted average of squared optimization number inside a cluster c. $(\cdot)_{\max}$ is used to take the maximum possible value of the terms in parentheses. w^p is defined as

$$w^p = \gamma\sum_{c\in C^p}\rho_c\sum_{i\in N_c}\pi_i r_c h_i,$$

10 allows us to measure the scale of the upper bound. However, it lacks clarity since the intermediate terms of $F(\mathbf{x}^p)$ for $p=1,\ldots,P$ are not eliminated. To this end, we can further define $w^{\max}=\max_{p=1}^{P}w^p$, $w^{\min}=\min_{p=1}^{P}w^p$ and get:

$$\frac{1}{P}\sum_{p=1}^{P}\left\|\nabla F\left(\mathbf{x}^{p-1}\right)\right\|^2 \leq \frac{2}{Pw^{\min}}\left[F\left(\mathbf{x}^0\right)-F\left(\mathbf{x}^P\right)\right]+$$
$$\frac{w^{\max}}{w^{\min}}\left[\frac{4}{3}G^2+\frac{1}{2\gamma L}\left(\epsilon_{i,c}^2+\eta^2\right)+\frac{r_{\max}+1}{4\gamma L}\epsilon_{c,g}^2\right], \tag{11}$$

Details of the proof are provided in Appendix A.

Equation (11) establishes an ergodic convergence rate of $\mathcal{O}(1/P)$ to an error floor, where the cluster-global gradient gap $\epsilon_{c,g}$ dominates $\epsilon_{i,c}$ due to its larger coefficient. This finding underscores our design choice to client cluster optimization directions with the global objective.

5 Experiments

This section is divided into the following subsections. In the first subsection, we elaborate on the learning tasks applied with HFL and the corresponding data distribution. In the second subsection, the detail settings of communication link throughput and computational resources are explained. The hyper-parameters and scenario settings of experiments are listed in Table 3. In the third subsection, we list the detailed metrics for co-optimization of cluster learning directions and cluster round time and mark them as the solutions in experiments. In the fourth subsection, two benchmark methods are documented with parameters aligned with notations used in this paper. The motivation and scenarios for a fair comparison between our solutions and the benchmarks are also explained. In the fifth subsection, we show the results of four sets of experiments, verifying the efficacy of our methods with respect to the two benchmarks.

Definition	Value
learning rate γ	0.004
global round time t_g	$\sim$ {80, 90, 120, 150, 180}
feature collection time t_μ, $\mu \in \{\phi, \text{skew}\}$	$t_\phi = 80$
inter-cluster sync. time $t_{c\to g}$	not simulated
classes per client I_c	$\sim$ {4, 7}
sample number standard deviation σ	$\sim$ {750, 1250}

TABLE 3: Hyper-parameter and scenario settings

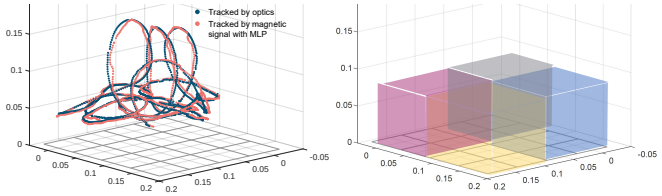


Fig. 4: Illustration of the magnet dataset. Sensors are located in the grids on the xy-plane. (a) Trajectory of the magnet tracked by multi-layer perceptron (MLP) using magnetic signals, evaluated with optically tracked trajectory. (b) Division of the tracking space for assigning virtual position label. Each subspace is coupled with magnet's two orientation category (elevation angle of magnet's north pole in $[0, 90°]$ or in $(90°, 180°]$) to create 8 virtual labels.

5.1 Learning Tasks and Data Partition

We apply the HFL solutions to categorical learning tasks CIFAR10 [58] and a regression task of magnet tracking [59]. For each task, since the optimization direction is determined by the data distribution across devices, we partition the data to create variations in optimization directions.

For classification datasets like CIFAR10, we aim to vary the data distribution among clients in terms of sample numbers and class availability. This is done in two stages. In the first stage, we set an lower bound for the number of samples per device, such that the standard deviation of sample sizes across devices matches a target value. This is achieved through quadratic optimization. In the second stage, we randomly assign classes to each client, and distribute samples to clients based on their individual availability of class. This stage is performed using linear optimization with results rounded to the nearest integer. We additionally include a data partitioning scheme where samples of each class are distributed according to a Dirichlet distribution. This approach similarly restricts the number of classes available on each client, but unlike our controlled partitioning method, it does not regulate the standard deviation of sample counts across clients.

For the regression task of magnet tracking, infinite points can be generated given a certain moving area of the magnet. An illustration of magnet tracking is shown in Section 5.1 and 10^6 points with corresponding magnetic signals are synthesized to create the dataset. Though the target of the task is to output real numbers, we still divide the points based on the position and orientation of the magnet and assign them virtual position labels accordingly. For the experiments in this paper, the data points are divided into 4 regions as shown in Section 5.1. Together with two divisions

of the magnet's orientation, the magnet dataset now has 8 position labels. With the virtual position labels, the magnetic signal dataset can be partitioned to devices in the same way as in the classification task to create data imbalance. The virtual label also serves as a sub-optional solution when applying one of the benchmark solutions to the regression task, which will be explained in Section 5.4.3.

Each partitioning and assignment of data samples of certain distributions are done with different random seeds in accordance with the randomization of environment conditions introduced in the following section.

We adopt a simple convolutional neural network (CNN) for learning CIFAR10 and an MLP with customized input/output layers, with their model structures listed in Appendix B. For CIFAR100, we use SqueezeNet [60] and ResNet18 [61] as backbone models. The learning capacity of the models is evaluated through centralized training. The CNN achieves 62% accuracy on the CIFAR10 categorical task, while the MLP attains a position error of 2.5 mm and angular error of 3.2° in the magnet tracking regression task. Using stachastic gradient descent as the optimizer, ResNet18 achieves 50% and SqueezeNet achieves 60% of testing accuracy on the CIFAR100 task.

5.2 Environment Settings

Inherited from the simulation platform in our previous work [9], the computational resources per client and communication link throughput in each D2D link can be dynamically adjusted at each simulation step.

In practice for HFL, the duration of a global round is assumed to be selected according to the variability of computation and communication resources, such that link and computation statistics can be reasonably approximated as stable within a round. In our experiments, we adopt this modeling choice by randomizing both computational resources and communication link throughput across global rounds, while keeping them fixed within each round. For specific environment dynamics, we conduct experiments with different random seeds and report the average learning performance. We list the settings of D2D links and variation bound of computational resources in Table 4.

Exp. ID Dataset	Set-1a CIFAR10	Set-1b/2 CIFAR10	Set-3 CIFAR10 CIFAR100	Set-1 magnet	Set-2/3 magnet
min D2D link	3	1	0/1	3	1
max D2D link	3	5	5	3	5
min bat. per step	10	10	5	80	80
max bat. per step	10	10	15	80	80
batch size	16	16	16	32	32

TABLE 4: Environment/resource settings for experiments.

5.3 Evaluation Criteria

We choose at most two metrics, μ^1 and μ^2, to evaluate in Algorithm 2. A swap is considered valid if a deterioration of α in metric μ^1 is compensated by an improvement of at least $z\alpha$ in μ^2, and vice-versa. Since all metrics (θ, T, θ/T) need to be minimized, we formulate the condition of a valid swap (Line 12 in Algorithm 2) to be:

$$\mu^{1\prime} \leq (1+\alpha)\mu^1 \text{ and } \mu^{2\prime} \leq (1-z\alpha)\mu^2,$$

with $\alpha > 0$ and $\mu^1 \neq \mu^2$.

For the selection of metrics, we experiment on the following options:

Solution 1.1: pair-wise optimization on θ and T

$$\mu^1 \text{ and } \mu^2 \in \{\theta_{c_1} + \theta_{c_2}, T_{c_1} + T_{c_2}\}$$

Solution 1.2: individual optimization on θ and T

$$\mu^1 \text{ and } \mu^2 \in \{\theta_c, T_c\}\,,\ \forall c \in \{c_1, c_2\}$$

Solution 2: pair-wise optimization on θ and θ/T

$$\mu^1 \text{ and } \mu^2 \in \left\{\theta_{c_1} + \theta_{c_2}, \frac{\theta_{c_1}}{T_{c_1}} + \frac{\theta_{c_2}}{T_{c_2}}\right\}$$

Solution 3: pair-wise optimization on T and θ/T

$$\mu^1 \text{ and } \mu^2 \in \left\{T_{c_1} + T_{c_2}, \frac{\theta_{c_1}}{T_{c_1}} + \frac{\theta_{c_2}}{T_{c_2}}\right\}$$

Moreover, we choose $z = 1.05$ for solutions 1.1, 1.2, and 2, and $z = 1.8$ for solution 3. n_{swap} is set to 80. The settings for z and n_{swap} are grid-searched and we choose the setting resulting in high evaluation accuracy simultaneously in various data distributions. The grid search result for z and n_{swap} are shown in Appendix C.1.

5.4 Benchmarks and Feature Substitutions

5.4.1 Benchmark A: Spectral Clustering

We adopt the spectral clustering method from [6] as Benchmark A. In this approach, the clustering feature vector for each device combines its estimated round time with pairwise model similarity values:

$$\text{Feature}_i = (t_i, \frac{\phi_i \cdot \phi_1}{\|\phi_i\| \cdot \|\phi_1\|}, \dots \frac{\phi_i \cdot \phi_N}{\|\phi_i\| \cdot \|\phi_N\|}).$$

In the original formulation of [6], all communication occurs over the device–server link, and clustering is performed virtually at the server without physically partitioning the network. Consequently, a device's communication latency is unaffected by its assigned cluster. For a fair comparison, we evaluate our solution against Benchmark A under conditions where all D2D links are identical and static. This ensures that the communication latency between each device and its cluster head remains stable, as detailed in Table 4 as settings for experiment Set 1a.

Moreover, the similarity matrix G for spectral clustering is computed using:

$$G_{i,j} = \exp\left(-\|\text{Feature}_i - \text{Feature}_j\|/2\sigma^2\right),$$

where devices with similar learning time and similar learning directions are grouped together, thus decreasing $\epsilon_{i,c}$ and increasing $\epsilon_{c,g}$. We perform line search for the best performing σ in Appendix Section C.2 and use the corresponding value for Benchmark A in Section 5.5.1 experiments.

5.4.2 Benchmark A+: Adapted Spectral Clustering

The clustering approach in Benchmark A, which groups devices with similar learning directions, is the opposite of our solution. To validate our approach for HFL, we modified the similarity matrix for spectral clustering to be the sum of the time similarity and feature dissimilarity.

In this case, we have the time similarity matrix entry computed as: $G^t_{i,j} = \exp\left(-\|t_i - t_j\|/2\sigma_t^2\right)$, and the feature dissimilarity matrix entry computed as:

$$G^{ft.}_{i,j} = \exp\left(\|\text{Feature}'_i - \text{Feature}'_j\|/2\sigma^2_{ft.}\right).$$

The similarity matrix for spectral clustering of Benchmark A+ is then: $G_{i,j} = G^t_{i,j} + G^{ft.}_{i,j}$. Here, clients with differing learning directions have higher $G^{ft.}_{i,j}$, and clients with similar computing time have higher $G^t_{i,j}$, leading them to be grouped together in spectral clustering. We perform grid search for the best performing σ_t and $\sigma_{ft.}$ in Appendix Section C.2 and use the corresponding value for Benchmark A+ in Section 5.5.1 experiments.

5.4.3 Benchmark B: Clustering based on Label Distribution

We refer to the clustering method based on label distribution [21] as Benchmark B, where a stricter version of swap searching is applied with the cluster sizes fixed. The metrics for evaluating cluster formation are defined as:

$$\text{skew}_c = \sum_{*\in\text{labels}} |p_c(*) - p_g(*)|,$$

where $p_c(*)$ and $p_g(*)$ is the probability of sampling a given label within cluster c and in the global data distribution, respectively. The computation of skew does not involve gradients or model differences. Additionally, Benchmark B assumes uniform D2D connections and an even partitioning of data, while also disregarding the optimization speed of learning devices. As a result, the clustering process in [21] does not account for optimizing cluster round time. Rewritten into the format of our metrics selection, the original Benchmark B can be expressed as:

$$\mu' \leq \mu,\ \mu = \text{skew}_{c_1} + \text{skew}_{c_2}$$

Feature	Privacy	Generation Complexity	Size	Pre-training
Original ϕ	+	++	++	yes
skew	-	-	-	no
ϕ^{crop}	++	++	+	yes
$\mathcal{F}$	+	+	-	no

TABLE 5: Comparison of substitutive features.

5.4.4 Feature Substitution

The aforementioned benchmarks do not compute or have limited capacity estimating the round time of the formed cluster due to the absent of D2D link condition in the optimization targets. However, with the inspiration of Benchmark B, we also derive substitutive features for determining the alignment of optimization direction in Algorithm 2. The substitutive features might have advantages in some aspects such as privacy preservation and low computation costs, but might have limited/unreliable information to indicate the optimization direction. In the later experiment section, we compare the performance of the following substitutive features:

Label Skew: the angle metric θ between features ϕ is replaced by skew in Algorithm 2 and Section 5.3. We append the suffix **skew** to denote this variant.

Partial Information of ϕ: the original feature vector ϕ is of the same structure as the model $\mathbf{x}$. Using ϕ to compute angles in the swap-search algorithm can be prohibitive in complexity when the number of parameters in ϕ is high. Apart from the common techniques in compressing model information, a subset of the information in ϕ might also serve as an indicator of optimization direction. To this end, we can select a few layers in ϕ where the number of parameters are small in quantity, and denote the selected feature as ϕ^{crop} and use it to compute θ^{crop} to replace θ in Algorithm 2. We add the suffix **crop** to our solution number to denote such feature variation.

Random Model Prediction: as experimented in [7], the inference results of clients using the identical random model can be indicative of clients' data distribution. Specifically, with forward function denoted as ψ, the prediction results of any client averaged over randomly sampled data points:

$$\mathcal{F}_i = \frac{1}{|D_i|}\sum_{b=1}^{\lfloor\frac{|D_i|}{B}\rfloor}\left(\sum_{j=1}^{B}\psi(\mathbf{x}, d_j)\right), d_j \sim D_i$$

are used to compute the similarity between clients based on the cosine distance: $\cos\left(\mathcal{F}_i, \mathcal{F}_{i'}\right)$. In our algorithm, the clients hold identical model after initialization ($\mathbf{x}^0$) and also after each global aggregation round p. Therefore, we use each $\mathbf{x}^p$ to compute $\mathcal{F}$ on each client. We further derive $\mathcal{F}_c = \sum_{i\in C_c}|D_i|\cdot\mathcal{F}_i/|D_c|$ and $\mathcal{F}_g = \sum_{i\in\{N\}}|D_i|\cdot\mathcal{F}_i/|D|$ to serve as the alternative feature and use $\theta^{\mathcal{F}}_c = \arccos\frac{\mathcal{F}_c\cdot\mathcal{F}_g}{\|\mathcal{F}_c\|\cdot\|\mathcal{F}_g\|}$ to replace θ in Algorithm 2. Suffix **pred** is added to our solution number to refer this solution.

We compare the features in terms of their privacy preservation and computation overhead in Table 5.

Data Distr.	N	Benchmark A	Benchmark A+	Solution 1.1	Solution 1.2
$\sigma = 750$ $I_c = 4$	20	42.1	44.0 (+ 4.5%)	44.7 (+6.3%)	**44.9** (+6.7 %)
$\sigma = 750$ $I_c = 7$	20	47.7	49.2 (+ 3.2%)	**49.6** (+3.8%)	49.5 (+3.7 %)
$\sigma = 1250$ $I_c = 4$	20	41.9	44.5 (+ 6.2%)	44.5 (+6.1%)	**44.7** (+6.6 %)
$\sigma = 1250$ $I_c = 7$	20	48.3	**49.3** (+ 2.2%)	49.2 (+2.0%)	**49.3** (+2.2 %)
Dir(0.1)	30	41.1	**42.8** (+ 4.3%)	**42.8** (+4.3%)	42.7 (+4.0 %)
Dir(0.5)	30	49.8	**50.5** (+ 1.3%)	50.4 (+1.2%)	50.3 (+0.9 %)

TABLE 6: Test accuracy after 2400 steps on CIFAR10 under different data distributions with **uniform** D2D connections. Percent change from Benchmark A is shown in parentheses.

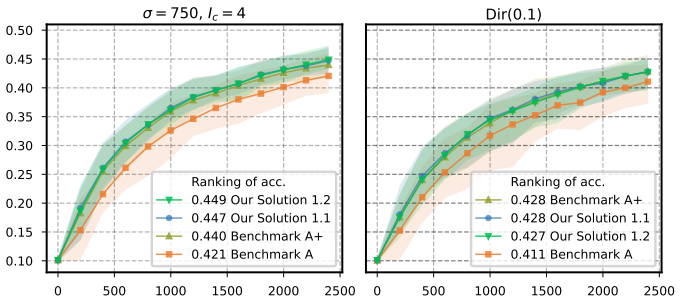


Fig. 5: Accuracy evolution of Our Solution 1 vs. Benchmark A on the CIFAR-10 dataset with various partitions, averaged over 10 trials. Shaded areas indicate the range across trials.

5.5 Results

5.5.1 Comparison with Benchmark A and A+

Data Distr.	N	Bench-mark A	Benchmark A+	Solution 1.1	Solution 1.2
$\sigma = 750$ $I_c = 4$	20	42.0	43.7 (+ 4.2%)	45.0 (+7.2%)	**45.0** (+7.3 %)
$\sigma = 750$ $I_c = 7$	20	47.6	48.8 (+ 2.6%)	49.9 (+4.8%)	**49.9** (+4.9 %)
$\sigma = 1250$ $I_c = 4$	20	41.8	43.5 (+ 3.9%)	44.5 (+6.3%)	**44.6** (+6.7 %)
$\sigma = 1250$ $I_c = 7$	20	47.8	48.8 (+ 2.2%)	**49.5** (+3.6%)	**49.5** (+3.6 %)
Dir(0.1)	30	40.8	42.6 (+ 4.4%)	**42.8** (+4.8%)	42.7 (+4.6 %)
Dir(0.5)	30	49.5	49.9 (+ 0.9%)	**50.6** (+2.3%)	50.5 (+2.1 %)

TABLE 7: Test accuracy after 2400 steps on CIFAR10 under different data distributions with **heterogeneous** D2D connections. Percent change from Benchmark A is shown in parentheses.

In Table 6, we compare the performance of our solutions 1.1 and 1.2 against Benchmarks A and A+ in the categorical learning task of CIFAR10, $t_g = 120$ and $t_\phi = 80$. The communication link throughput between each pair of devices is assumed to be uniform. Clusters' round time is then upper-bounded by the member device which needs the longest time to finish computing. The modified Benchmark A+ and both of our solutions are particularly advantageous when data is unbalanced ($I_c = 4$), with more than 6% improvement in terms of final accuracy while taking 500 fewer steps (25% less time) to reach the test accuracy of 0.4. For the case of relatively balanced data distribution, our solution reaches 80% accuracy of the centrally trained model. We show the evolution of test accuracy under two data distributions in Figure 5.

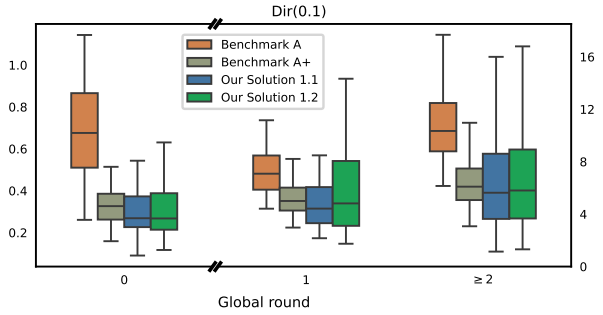
The difference between optimizing metrics of individual clusters (solution 1.2) and optimizing metrics of a pair of clusters (solution 1.1) is small, with solution 1.1 exhibits more fluctuation in the evolution of test accuracy.

Benchmark A and its variation A+ do not have the capacity to deal with heterogeneous link conditions between clients. To verify this, we conduct the experiments with non-equal link conditions (as Set 1b settings in Table 4) across D2D links and have Benchmark A and A+ estimating the round time with computation time only. We present the results in Table 7 and observe the general degradation of performance of Benchmark A and A+ compared to the results in Table 6 and the maintenance of performance of our solutions.

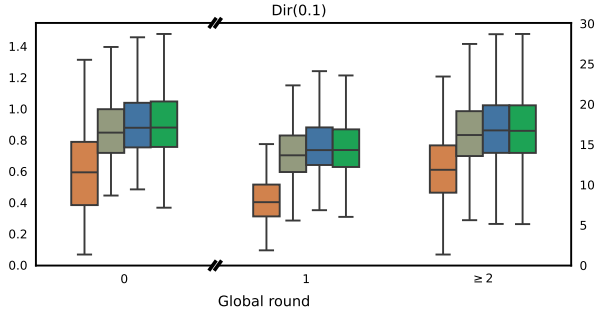
Data Distr.	Benchmark A	Benchmark A+	Solution 1.1	Solution 1.2
$\sigma = 750$ $I_c = 4$	1.59	1.74	1.74	1.76
$\sigma = 750$ $I_c = 7$	1.98	2.05	2.02	2.03
$\sigma = 1250$ $I_c = 4$	1.49	1.71	1.75	1.75
$\sigma = 1250$ $I_c = 7$	1.94	2.01	2.02	2.01
Dir(0.1)	1.58	1.65	1.74	1.75
Dir(0.5)	2.19	2.11	2.13	2.16

TABLE 8: Tested $\frac{1}{P}\sum_{p=1}^{P}\frac{2}{w^p}\left[F\left(\mathbf{x}^{p-1}\right) - F\left(\mathbf{x}^{p}\right)\right]$ scaled by 10^4 using different methods learning CIFAR10

To verify the convergence analysis in Section 4.2, we log and compute the values of $\frac{1}{P}\sum_{p=1}^{P}\frac{2}{w^p}\left[F\left(\mathbf{x}^{p-1}\right) - F\left(\mathbf{x}^{p}\right)\right]$ and show them in Table 8 for the experiments conducted with uniform D2D links. We noticed that the better performing methods generally have higher values in Table 8. This is because the loss is reduced more with better performing methods, and therefore resulting in higher values for each $F\left(\mathbf{x}^{p-1}\right) - F\left(\mathbf{x}^{p}\right)$, despite increasing w^p.



(a) Distribution of $\|\nabla F_c(\mathbf{x}^p) - \nabla F(\mathbf{x}^p)\|$



(b) Distribution of $\|\nabla F_i(\mathbf{x}^p) - \nabla F_c(\mathbf{x}^p)\|$

Fig. 6: Scales of gradient mismatch of different method when learning CIFAR10 under uniform D2D link condition and Dir(0.1) data distribution.

We further verify that the bound for $\frac{1}{P}\sum_{p=1}^{P}\left\|\nabla F\left(\mathbf{x}^{p-1}\right)\right\|^2$ is lower due to our methods effectively reducing cluster-global divergence. It is worth

Data Distr.	Benchmark A	Benchmark A+	Solution 1.1	Solution 1.2
	Average of $\|\nabla F_c(\mathbf{x}^p) - \nabla F(\mathbf{x}^p)\|$			
$\sigma = 750$ $I_c = 4$	7.63	4.25 *(0.56)*	3.92 *(0.51)*	4.02 *(0.53)*
$\sigma = 750$ $I_c = 7$	3.84	2.36 *(0.62)*	2.06 *(0.54)*	2.30 *(0.60)*
$\sigma = 1250$ $I_c = 4$	7.81	4.48 *(0.57)*	5.13 *(0.66)*	4.98 *(0.64)*
$\sigma = 1250$ $I_c = 7$	3.92	2.54 *(0.65)*	2.66 *(0.68)*	2.69 *(0.69)*
Dir(0.1)	9.67	5.78 *(0.60)*	5.81 *(0.60)*	5.98 *(0.62)*
Dir(0.5)	4.16	2.72 *(0.65)*	2.09 *(0.50)*	2.26 *(0.54)*
	Average of $\|\nabla F_i(\mathbf{x}^p) - \nabla F_c(\mathbf{x}^p)\|$			
$\sigma = 750$ $I_c = 4$	8.34	11.78 *(1.41)*	12.18 *(1.46)*	12.07 *(1.45)*
$\sigma = 750$ $I_c = 7$	4.69	6.12 *(1.31)*	6.36 *(1.36)*	6.26 *(1.34)*
$\sigma = 1250$ $I_c = 4$	8.08	11.76 *(1.46)*	11.84 *(1.47)*	11.78 *(1.46)*
$\sigma = 1250$ $I_c = 7$	4.57	6.05 *(1.32)*	6.20 *(1.36)*	6.14 *(1.34)*
Dir(0.1)	10.75	14.97 *(1.39)*	15.43 *(1.44)*	15.36 *(1.43)*
Dir(0.5)	5.63	7.03 *(1.25)*	7.28 *(1.29)*	7.23 *(1.29)*

TABLE 9: Gradient mismatch on CIFAR10 with different data distributions (relative to Benchmark A).

noting that $\frac{1}{2\gamma L}\epsilon_{i,c}^2 + \frac{r_{\max}+1}{4\gamma L}\epsilon_{c,g}^2$ in the upper bound of Equation (11) is a rough estimation of the cluster-client gradient mismatch error and cluster-global gradient mismatch error. In practice, each $\|\nabla F_i(\mathbf{x}^p) - \nabla F_c(\mathbf{x}^p)\|$ and $\|\nabla F_c(\mathbf{x}^p) - \nabla F(\mathbf{x}^p)\|$ contributes to this type of error cumulatively. To this end, at the beginning of each global round p with global model $\mathbf{x}^p$, $\nabla F_i(\mathbf{x}^p)$ is tested with all data samples of client i and then averaged with corresponding weight to get $\nabla F_c(\mathbf{x}^p)$ and $\nabla F(\mathbf{x}^p)$. We show the statistical distribution of gradient mismatch under the condition of the uniform D2D link Figure 6. Furthermore, we list all the average of gradient mismatch of experiments in various data distribution in Table 9. Our method, along with Benchmark A+, achieves a greater reduction in the cluster-to-global gradient mismatch $\overline{\nabla F_c(\mathbf{x}^p) - \nabla F(\mathbf{x}^p)\|}$, than the corresponding increase in client-to-cluster gradient deviation, $\overline{\|\nabla F_i(\mathbf{x}^p) - \nabla F_c(\mathbf{x}^p)\|}$. Notably, for $r_{\max} \geq 4$, our solution and Benchmark A+ effectively reduce the combined upper bound term $\frac{r_{\max}+1}{4}\overline{\|\nabla F_i(\mathbf{x}^p) - \nabla F_c(\mathbf{x}^p)\|}^2 + \frac{1}{2}\overline{\|\nabla F_c(\mathbf{x}^p) - \nabla F(\mathbf{x}^p)\|}^2$. We also notice that the gradient mismatch error is in the scale of ≥ 100, which is significantly larger than $\frac{1}{P}\sum_{p=1}^{P}\frac{2}{w^p}\left[F\left(\mathbf{x}^0\right) - F\left(\mathbf{x}^P\right)\right]$. Therefore, despite the increase value of loss gap, our methods effectively reduced the gradient mismatch, which is directly associated with improved performance, as reflected in the higher test accuracy reported in Table 6.

In Figure 7, we show the position and orientation error of the regression task, with models trained over $N = 24$ devices and $t_g = t_\phi = 80$. Devices are equipped with magnet signals collected from a certain area corresponding to the virtual label. We observe that to achieve any target error below 5 mm and 6°, Benchmark A requires around two additional global rounds (320 more steps) compared to our solutions. Consequently, our solutions reduce the time to reach the target position error (5 mm) by at least 18% and the target orientation error (6°) by at least 16.7%, respectively.

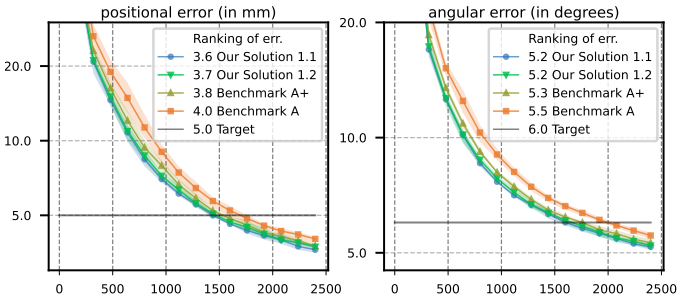


Fig. 7: Error evolution of our solution 1 vs Benchmark A tested on regression dataset with $I_c = 3$, $\sigma = 12500$.

5.5.2 *Criteria for Co-Optimization*

In the following set of experiments, the throughput of D2D links is varied after each global learning round, following the configurations provided in Table 4 for experiment set 2. In practical applications, a global learning round is typically defined such that D2D links remain stable during the round. To reflect this, we examine the applicability of our solutions under different global round durations when training on CIFAR10. We compare Solutions 1.1, 2, and 3, presenting their overall learning progress in Figure 8 and highlighting representative cases in Figure 8.

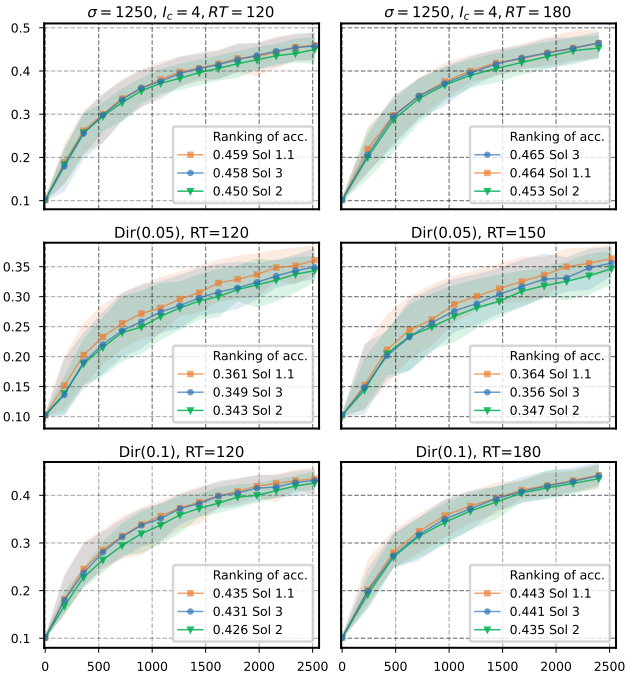


Fig. 8: Accuracy evolution of our solution 1.1, solution 2, and solution 3 under different global round time, tested on CIFAR10 dataset selected partition.

We observe that optimizing the metric θ/T tends to favor configurations with a longer round time T. Since

Data Distr.	RT=90 1.1	RT=90 2	RT=90 3	RT=120 1.1	RT=120 2	RT=120 3	RT=150 1.1	RT=150 2	RT=150 3	RT=180 1.1	RT=180 2	RT=180 3
σ=750 I_c=4	45.4	43.6 (-3.9)	44.5 (-2.0)	46.8	44.6 (-4.7)	46.1 (-1.6)	47.2	44.7 (-5.4)	46.7 (-1.2)	47.1	45.1 (-4.3)	46.4 (-1.4)
σ=750 I_c=7	49.9	47.9 (-3.9)	49.2 (-1.3)	51.4	49.4 (-4.0)	50.7 (-1.5)	52.0	50.0 (-3.9)	51.1 (-1.8)	52.1	50.1 (-3.9)	51.0 (-2.1)
σ=1250 I_c=4	44.8	43.7 (-2.3)	44.6 (-0.4)	45.9	45.0 (-2.1)	45.8 (-0.3)	46.6	45.2 (-3.0)	46.5 (-0.2)	46.4	45.3 (-2.4)	46.5 (+0.2)
σ=1250 I_c=7	49.7	48.5 (-2.4)	49.0 (-1.4)	51.3	50.1 (-2.3)	50.9 (-0.6)	51.8	50.3 (-2.9)	51.4 (-0.7)	51.5	50.3 (-2.4)	51.3 (-0.4)
σ=1600 I_c=2	39.2	37.6 (-4.1)	38.3 (-2.3)	39.8	38.7 (-2.8)	39.2 (-1.5)	40.7	37.7 (-7.2)	38.9 (-4.3)	40.3	39.1 (-2.9)	39.6 (-1.6)
Dir(0.05)	34.8	32.7 (-6.2)	33.7 (-3.3)	36.1	34.3 (-4.8)	34.9 (-3.2)	36.4	34.7 (-4.7)	35.6 (-2.1)	35.4	33.5 (-5.6)	34.0 (-4.0)
Dir(0.1)	42.7	41.1 (-3.6)	42.0 (-1.5)	43.5	42.6 (-2.1)	43.1 (-0.8)	44.1	42.7 (-3.2)	44.0 (-0.4)	44.3	43.5 (-1.7)	44.1 (-0.3)
Dir(0.5)	49.5	47.4 (-4.2)	48.4 (-2.4)	50.9	48.6 (-4.4)	49.7 (-2.3)	51.4	49.2 (-4.2)	50.1 (-2.4)	51.5	49.1 (-4.7)	50.3 (-2.3)

TABLE 10: CIFAR10 test accuracy across various data distributions, with varying D2D connections.

Solution 2 does not explicitly constrain T, it results in low learning efficiency due to fewer cluster learning rounds and consequently fewer optimization steps. By contrast, Solution 3 aims to encourage shorter round times, while also regulates the learning speed of diverging clusters. This can be particularly beneficial under extreme data imbalance, as it leads to smoother and faster overall progress. However, Solution 3 only outperforms Solution 1.1 in terms of the final test accuracy under the specific setting where $I_c = 4$, $\sigma = 1250$, and the global round time is 180 steps. Moreover, as shown in Figure 8, the accuracy upper bound of Solution 1.1 remains consistently higher than that of Solution 3 after 2000 steps. This suggests that the intended role of Solution 3 – balancing learning-direction alignment with the number of cluster rounds – has not been fully achieved.

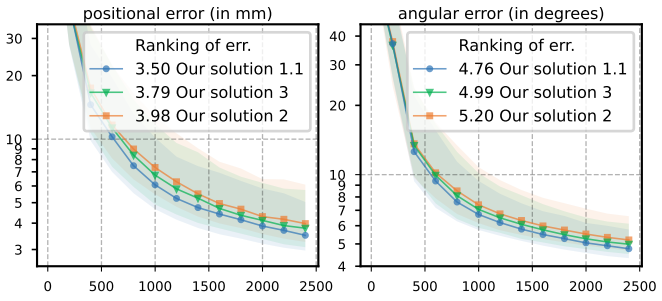


Fig. 9: Error evolution of our solutions 1.1, 2, and 3, tested on regression dataset with $I_c = 2$, $\sigma = 20000$.

In the regression tasks (Figure 9), Solution 3 similarly fails to surpass Solution 1.1. This may be due to insufficient imbalance in the regression dataset, as the virtual position labels used to denote heterogeneity are inherently inaccurate. We further discuss these observations and their implications in Section 5.5.4.

5.5.3 Comparison with Benchmark B and Substitutive Features

Directly sharing label distribution as in Benchmark B raises potential privacy concerns. Despite this, we tolerate it for evaluation purposes to compare the performance of our solution with Benchmark B. We also show the variation of our solution using skew to replace θ. We also evaluate a

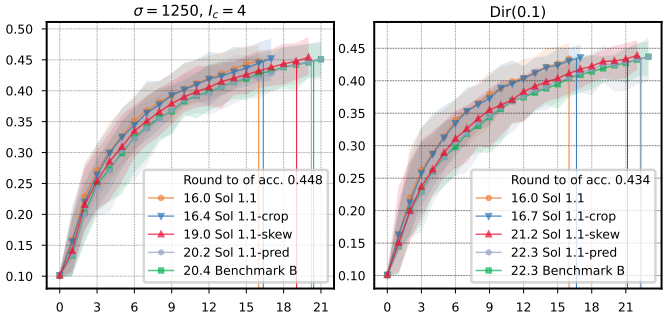


Fig. 10: Accuracy evolution of our Solution 1.1 with various features vs. Benchmark B, tested on CIFAR10 dataset with selected data partitions.

variant of our method where the skew metric replaces θ, as well as two other feature substitutes, ϕ^{crop} and $\mathcal{F}$, described in Section 5.4.4. Based on the evidence in Section 5.5.2 that swap search with simple summed metrics (Solution 1.1 with metric $\theta_{c_1} + \theta_{c_2}$, $T_{c_1} + T_{c_2}$) outperforms the alternatives, we focus on the variations of Solution 1.1, namely Solution 1.1-crop, Solution 1.1-skew, and Solution 1.1-pred. In addition to modeling the dynamics of D2D links, we vary the computational resources of each FL client as detailed in Table 4 for experiment set 3.

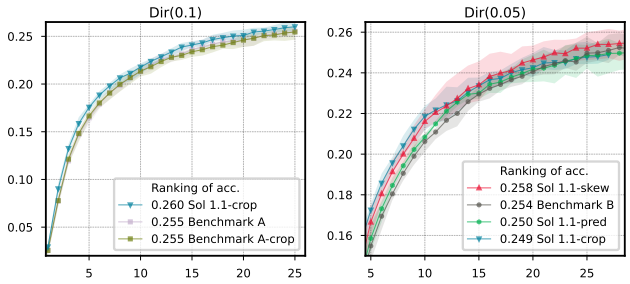
Since feature collection time in Benchmark B and our solutions with feature variants may differ in practice, we now compare the accuracy performance of the methods with respect to the number of global learning rounds ($t_g = 100$ steps).

In the CIFAR10 classification task, all D2D links are kept active, with a minimum link throughput greater than zero. Our solution is run for $R^{\phi} = 16$ global rounds, and its final accuracy is used as the target. Other methods are allowed to run for additional global rounds if necessary to reach or surpass this target accuracy. To estimate the number of global rounds required for each method (R_{method}, where method $\in$ 1.1-pred, 1.1-skew, 1.1-crop, Benchmark B) to achieve the target, we use Piecewise Cubic Hermite Interpolating Polynomial (PCHIP) interpolation [62].

With the computed number of rounds, we further incorporate the inter-cluster synchronization time $t_{c \to g}$. Al-

Data partition	Target acc.	$R_{\phi^{\text{crop}}}$	$t_\phi = 50$ outperform when	cannot outperform when
Dir(0.1)	43.4	16.7	$t_{\phi^{\text{crop}}} \leq 43.0$	$t_\phi < 5.3$
Dir(0.5)	51.1	16.8	$t_{\phi^{\text{crop}}} \leq 41.9$	$t_\phi < 6.1$
$\sigma = 750$, $I_c = 4$	45.2	17.4	$t_{\phi^{\text{crop}}} \leq 36.0$	$t_\phi < 11.0$
$\sigma = 750$, $I_c = 7$	50.1	16.7	$t_{\phi^{\text{crop}}} \leq 42.7$	$t_\phi < 5.5$
$\sigma = 1250$, $I_c = 4$	44.8	16.4	$t_{\phi^{\text{crop}}} \leq 46.0$	$t_\phi < 2.9$
$\sigma = 1250$, $I_c = 7$	49.6	16.1	$t_{\phi^{\text{crop}}} \leq 48.6$	$t_\phi < 1.0$

TABLE 11: Conditions for Solution 1.1-crop to outperform the original solution 1.1 when $t_g = 100$, $t_{c\to g} = 30$, and $R_\phi = 16$



(a) Compare with Benchmark A, using 100 clients **(b)** Compare with Benchmark B, using 80 clients

Fig. 11: Accuracy evolution of Solution 1.1 with different features vs. two benchmarks on CIFAR100.

though not explicitly simulated, it occurs at the end of each global round for all methods. We introduce $t_{c\to g}$ to assess the feasibility of specific t_ϕ values. Since both $t_{c\to g}$ and t_ϕ involve transmitting models from devices to the server and are therefore comparable in magnitude. For example, when $t_{c\to g}$ is set to 30 steps, making synchronized device-server FL prohibitively expensive, t_ϕ should be slightly higher (e.g. 50 steps) because it also includes local optimization in addition to all devices sending information to the server.

The condition for another method to outperform our solution can then be expressed as:

$$R_\phi \cdot (t_\phi + t_g + t_{c\to g}) > R_{\text{method}} \cdot (t_{ft.} + t_g + t_{c\to g}),$$

where $t_{ft.}$ is the time required for clients to compute their features and send them to the central server. Specifically, we have $t_{\phi^{\text{crop}}}$ for Solution 1.1-crop, t_{skew} for Benchmark B and Solution 1.1-skew, and $t_{\mathcal{F}}$ for Solution 1.1-pred.

Moreover, if t_ϕ is not fixed, we can compute its value such that the outperforming condition becomes:

$$t_{ft.} < \frac{R_\phi}{R_{\text{method}}} \cdot (t_\phi + t_g + t_{c\to g}) - (t_g + t_{c\to g}) < 0,$$

which rules out the possibility of the other method surpassing our solution under the given settings.

Since ϕ^{crop} requires pre-training to generate while other features do not, we report the corresponding threshold values in Table 11 and Table 12. In experiments listed in Table 11, the model updates corresponding to the smallest convolutional layer of the CNN are cropped out to form the feature ϕ^{crop}. We observe that using ϕ^{crop} generally requires approximately one additional global round to achieve performance comparable to the original feature. The corresponding outperforming time threshold ($t_{\phi^{\text{crop}}}$) is easily attainable, given that ϕ^{crop} is only 0.7% of the size of the full feature vector ϕ.

In experiments listed in Table 12, Benchmark B performs the worst among all methods, requiring roughly five additional global rounds to reach the target accuracy. This is because Benchmark B forms clusters without considering link conditions, resulting in clusters that may have aligned optimization directions but long cluster round times. By contrast, incorporating link and computational conditions for joint optimization, as in Solution 1.1-skew, accelerates learning and reduces the number of global rounds needed to reach the target accuracy. We also observe that Dirichlet-distributed data poses a greater challenge for substitutive features that require no pre-training (skew and $\mathcal{F}$), resulting in their slower learning progress compared to other data partitions. Using $\mathcal{F}$ as a clustering feature is generally less effective than using skew, on average requiring one additional global round to reach the target accuracy, although it has the advantage of preserving client privacy. Both skew and $\mathcal{F}$ are lightweight features; however, generating $\mathcal{F}$ requires multiple rounds of inference, making it more challenging than using skew to meet the time threshold needed to outperform the original Solution 1.1.

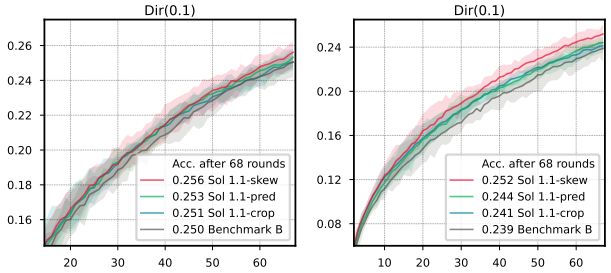
We further evaluate the methods on CIFAR100 and the magnetic trace dataset. In Figure 11 and Figure 12, we present results on CIFAR-100 under challenging heterogeneous settings, where each class is distributed across 80/100/200 devices following a Dirichlet distribution with $\alpha = 0.05$ and 0.1. Due to the prohibitive size of the full feature vector ϕ, we exclude it from swap-search and instead use its cropped version ϕ^{crop}, obtained from the smallest four convolutional layers (1.086% of full layers in ResNet-18, 0.063% in SqueezeNet). Accuracy is evaluated at each global round.

In Figure 11a, all solutions use the same feature collection time. Here, ϕ^{crop} proves effective: Benchmark A with ϕ^{crop} performs equivalently to the original Benchmark A, while Solution 1.1-crop, which integrates ϕ^{crop} into the swap search algorithm, achieves superior performance. However, in Figure 11b, Solution 1.1-crop's early advantage from training to obtain ϕ^{crop} diminishes once the global model reaches an accuracy of 0.2. Under such highly non-IID conditions, the label distribution (used to compute skew) provides a more reliable optimization signal than either $\mathcal{F}$ or ϕ^{crop}, as Benchmark B eventually surpasses both Solution 1.1-pred and Solution 1.1-crop. This conclusion is further supported in Figure 12. When D2D links are always available (Figure 12a), Benchmark B eventually catches up with Solution 1.1-crop despite ignoring link topology. In contrast, when D2D links are intermittently unavailable, co-optimizing learning direction and link topology becomes critical, with variants of our proposed solutions clearly outperforming Benchmark B in this setting.

For regression tasks, the label skew in Benchmark B was originally designed for classification. To adapt it, we incorporate virtual position labels to form unbalanced distributions for clustering in Solution 1.1-skew. As shown in Figure 13, our original solution significantly outperforms

Data partition	Target acc.	R_{B}	if $t_\phi = 50$ outperf. when	cannot outperf. when	$R_{\text{1.1-pred}}$	if $t_\phi = 50$ outperf. when	cannot outperf. when	$R_{\text{1.1-skew}}$	if $t_\phi = 50$ outperf. when	cannot outperf. when
Dir(0.1)	43.4	22.3	/	t_ϕ <51.4	22.3	/	t_ϕ <51.0	21.2	t_{skew} ≤6.2	t_ϕ <41.9
Dir(0.5)	51.1	22.6	/	t_ϕ <53.3	21.1	$t_{\mathcal{F}}$ ≤6.6	t_ϕ <41.3	21.0	t_{skew} ≤7.2	t_ϕ <40.5
$\sigma = 750$, $I_c = 4$	45.2	21.2	t_{skew} ≤5.7	t_ϕ <42.4	20.8	$t_{\mathcal{F}}$ ≤8.6	t_ϕ <38.9	19.2	t_{skew} ≤20.3	t_ϕ <25.7
$\sigma = 750$, $I_c = 7$	50.1	21.6	t_{skew} ≤3.5	t_ϕ <45.3	20.6	$t_{\mathcal{F}}$ ≤9.8	t_ϕ <37.4	19.9	t_{skew} ≤14.8	t_ϕ <31.6
$\sigma = 1250$, $I_c = 4$	44.8	20.4	t_{skew} ≤11.1	t_ϕ <35.9	20.2	$t_{\mathcal{F}}$ ≤12.4	t_ϕ <34.3	19.0	t_{skew} ≤21.2	t_ϕ <24.7
$\sigma = 1250$, $I_c = 7$	49.6	20.9	t_{skew} ≤7.7	t_ϕ <40.0	20.1	$t_{\mathcal{F}}$ ≤13.6	t_ϕ <32.9	19.3	t_{skew} ≤18.9	t_ϕ <27.1

TABLE 12: Outperforming conditions for methods with no-training features vs. Solution 1.1 when $t_g = 100$, $t_{c\to g} = 30$, and R_ϕ



(a) D2D links always available **(b)** intermittent connectivity

Fig. 12: Accuracy of Solution 1.1 with different features vs. Benchmark B on CIFAR100 with Squeezenet across 200 clients.

Solution 1.1-skew, requiring substantially fewer rounds to reach the target error. This highlights that the feature ϕ provides a more reliable indicator of learning direction than synthetic labels derived from regression targets.

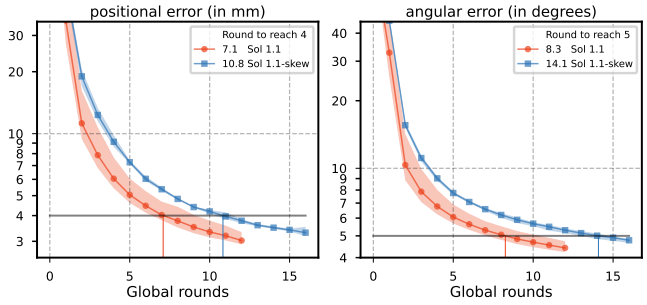


Fig. 13: Error evolution of our solution 1.1 vs solution 1.1-**skew**, tested on regression dataset with $I_c = 3$, $\sigma = 12500$.

5.5.4 *Discussion*

Reasons for Solution 3's Suboptimal Performance Out of all the experiments in Table 10, our solution 3 only outperforms solution 1.1 in one case ($\sigma = 1250$, $I_c = 4$, $RT = 180$). Moreover, in the regression task presented in Figure 9, solution 3 also exhibits an clear disadvantage. This all shows that we need to better scheme to control the error in cluster-global gradient divergence. In Equation (10), we notice that the accumulated error is not only proportional to the number of cluster rounds r, but also the number of optimizations h performed by the clients inside a cluster. Furthermore, the cluster-global divergence $\epsilon_{c,g}$ is the scale of the vector difference and squared, while in our solution the difference is represented with angle difference $\theta = \arccos \frac{\phi_1 \phi_2}{|\phi_1||\phi_2|}$. The disparity of the representation of cluster-global divergence in theory and in practice might be the reason for the sub-optimal performance of our solution trying to control the accumulated error originated from cluster-global divergence.

Choice of Substitutive Features: When using other features in categorical tasks, although achieving $t_{\text{skew}} < 20$ is feasible due to the much smaller data size compared to model transmission, it remains challenging when the device-server link is primarily limited by access delay — an expected difficulty since all devices must communicate with the server. The feature collection time for $\mathcal{F}$ is expected to be longer than t_{skew} since it involves model inference, therefore making it harder to outperform the original solution. However, if privacy is a critical concern and the client-server link is also severely limited, using $\mathcal{F}$ as the feature is a good candidate since it preserves privacy and is also light in data size. For simpler tasks where the optimization progress in a couple of layers can reasonably indicate the general optimization direction, ϕ^{crop} is also a nice substitute to preserve privacy as well as facilitate feature collection.

Complexity of Swap Search is theoretically $O(N^2)$ based on Algorithm 3. For the case of extremely large N, we suppose the size of the cluster also grows along N. According to the law of large numbers, when the number of clients in a cluster is large, the data distribution of the cluster also gets close to the global data distribution. Therefore, we just need a few more rounds of swap search to avoid slow links in some clusters. Moreover, we can also start with the result of reverse-spectral-clustering (clustering dissimilar clients together), then use swap search method to adjust cluster configuration to avoid slow links inside a cluster. The swap search is also facilitated by using smaller substitutive features such as ϕ^{crop} and $\mathcal{F}$. Taken together, the combination of substitutive features, heuristic initial clusters, and limited swaps to solve bottleneck links ensures that the swap-search algorithm remains feasible in both computation and communication..

Method against Byzantine attacks: Common techniques against model poisoning can be implemented along with our solutions. As for the clients reporting false information about the D2D link throughput, the issue can be addressed by comparing if the link throughput between a pair of clients are identical in both directions. In other words, if the matrix of transmission time between clients are not symmetrical diagonally, the false link information reported by malicious clients can be detected. As for clients reporting

true link information, but later limit their bandwidth in intra-cluster synchronization, the head of the cluster can also detect and report this to the server and then excluding the malicious clients in later rounds.

6 Conclusion

In this paper, we proposed a co-optimization scheme to address learning devices' heterogeneity of HFL in D2D networks. Our solution forms clusters of devices that achieve fast synchronization while ensuring their optimization direction remains aligned with the global learning direction. To support our method, we provide a convergence analysis that highlights the need to address the misalignment between cluster learning directions and the overall global learning process. Extensive experiments on both classification and regression tasks confirm the effectiveness of our approach. Under uniform D2D links, our method improves classification accuracy by around 6% and reduces the time to reach target error in regression tasks by at least 16% compared with Benchmark A. In realistic heterogeneous-link scenarios, where Benchmark A cannot be applied, we investigated multiple optimization metrics and substitutive feature designs and identified methods that are privacy preserving and meet the possible network constraints. These results establish our method as an efficient and generalizable framework for scalable HFL in D2D networks.

Future research directions include refining clustering algorithms with stage-aware strategies to different phases of learning, extending the framework to parameter-server and edge-assisted HFL architectures through bandwidth-aware virtual clustering or intra-cluster client selection, and enabling hopping/client-selection within clusters to enhance efficiency and robustness. Additional directions include developing defenses against malicious manipulations of link or resource information, and integrating real-world measurements and hardware evaluations to validate performance under practical deployment conditions.

References

[1] Q. Yang, Y. Liu, T. Chen, et al., "Federated Machine Learning: Concept and Applications," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 2, pp. 1–19, Jan. 2019.

[2] A. Reisizadeh, I. Tziotis, H. Hassani, et al., "Straggler-Resilient Federated Learning: Leveraging the Interplay Between Statistical Accuracy and System Heterogeneity," *IEEE Journal on Selected Areas in Information Theory*, vol. 3, no. 2, pp. 197–205, Jun. 2022.

[3] L. Liu, J. Zhang, S. Song, et al., "Client-Edge-Cloud Hierarchical Federated Learning," in *IEEE International Conference on Communications (ICC 2020)*, Virtual Conference: IEEE, Jun. 2020, pp. 1–6.

[4] A. Ghosh, J. Chung, D. Yin, et al., "An Efficient Framework for Clustered Federated Learning," in *34th Conference on Neural Information Processing Systems (NeurIPS 2020)*, H. Larochelle, M. Ranzato, R. Hadsell, et al., Eds., vol. 33, Virtual Conference: Curran Associates Inc., Dec. 2020, pp. 19 586–19 597.

[5] C. Briggs, Z. Fan, and P. Andras, "Federated learning with hierarchical clustering of local updates to improve training on non-IID data," in *2020 International Joint Conference on Neural Networks (IJCNN 2020)*, Glasgow, United Kingdom: IEEE, Jul. 2020, pp. 1–9.

[6] Y. Zhang, D. Liu, M. Duan, et al., "FedMDS: An Efficient Model Discrepancy-Aware Semi-Asynchronous Clustered Federated Learning Framework," *IEEE Transactions on Parallel and Distributed Systems*, vol. 34, no. 3, pp. 1007–1019, Mar. 2023.

[7] G. Lao, X. Zhang, Y. Li, et al., "Lightweight Clustered Federated Learning via Feature Extraction," in *50th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2025)*, Hyderabad, India: IEEE, Apr. 2025, pp. 1–5.

[8] N. Mhaisen, A. A. Abdellatif, A. Mohamed, et al., "Optimal User-Edge Assignment in Hierarchical Federated Learning Based on Statistical Properties and Network Topology Constraints," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 1, pp. 55–66, Jan. 2022.

[9] M. Wu, M. Boban, and F. Dressler, "Flexible Training and Uploading Strategy for Asynchronous Federated Learning in Dynamic Environments," *IEEE Transactions on Mobile Computing*, vol. 23, no. 12, pp. 12 907–12 921, Dec. 2024.

[10] A. Khaled, K. Mishchenko, and P. Richtárik, "First Analysis of Local GD on Heterogeneous Data," arXiv, cs.IT 1909.04715, Sep. 2019.

[11] Q. Li, M. Zhang, T. Sun, et al., "DFedGFM: Pursuing global consistency for Decentralized Federated Learning via global flatness and global momentum," *Neural Networks*, vol. 184, p. 107 084, Apr. 2025.

[12] P. Foret, A. Kleiner, H. Mobahi, et al., "Sharpness-Aware Minimization for Efficiently Improving Generalization," in *9th International Conference on Learning Representations (ICLR 2021)*, Virtual Conference: OpenReview.net, May 2021.

[13] Z. Qu, X. Li, R. Duan, et al., "Generalized Federated Learning via Sharpness Aware Minimization," in *39th International Conference on Machine Learning (ICML 2022)*, K. Chaudhuri, S. Jegelka, L. Song, et al., Eds., vol. 162, Baltimore, MD: PMLR, Jul. 2022, pp. 18 250–18 280.

[14] X. Xing, Q. Zhan, X. Xie, et al., "Flexible Sharpness-Aware Personalized Federated Learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, no. 20, pp. 21 707–21 715, Apr. 2025.

[15] M. Hu, P. Zhou, Z. Yue, et al., "FedCross: Towards Accurate Federated Learning via Multi-Model Cross-Aggregation," in *IEEE 40th International Conference on Data Engineering (ICDE 2024)*, Utrecht, Netherlands: IEEE, May 2024, pp. 2137–2150.

[16] M. Hu, Z. Yue, X. Xie, et al., "Is Aggregation the Only Choice? Federated Learning via Layer-wise Model Recombination," in *30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD 2024)*, Barcelona, Spain: ACM, Aug. 2024, pp. 1096–1107.

[17] Z. Zheng, Y. Deng, X. Liu, et al., "Asynchronous Federated Learning via Over-the-Air Computation," in *IEEE Global Communications Conference (GLOBECOM 2023)*, Kuala Lumpur, Malaysia: IEEE, Dec. 2023, pp. 1345–1350.

[18] A. Koloskova, S. Stich, and M. Jaggi, "Sharper Convergence Guarantees for Asynchronous SGD for Distributed and Federated Learning," in *36th Conference on Neural Information Processing Systems (NeurIPS 2022)*, S. Koyejo, S. Mohamed, A. Agarwal, et al., Eds., vol. 35, New Orleans, LA: Curran Associates Inc., Nov. 2022, pp. 17 202–17 215.

[19] Y. Chen, Y. Ning, M. Slawski, et al., "Asynchronous Online Federated Learning for Edge Devices with Non-IID Data," in *IEEE International Conference on Big Data (BigData 2020)*, Virtual Conference: IEEE, Dec. 2020, pp. 15–24.

[20] T. Li, A. K. Sahu, M. Zaheer, et al., "Federated optimization in heterogeneous networks," in *3rd Conference on Machine Learning and Systems (MLSys 2020)*, Austin, TX, Mar. 2020.

[21] A. Bellet, A.-M. Kermarrec, and E. Lavoie, "D-Cliques: Compensating for Data Heterogeneity with Topology in Decentralized Federated Learning," in *41st International Symposium on Reliable Distributed Systems (SRDS 2022)*, Vienna, Austria: IEEE, Sep. 2022, pp. 1–11.

[22] J. Chen, P. Zhang, J. Chen, et al., "Heterogeneity-Aware Clustering and Intra-Cluster Uniform Data Sampling for Federated Learning," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 8, pp. 1–12, Dec. 2024.

[23] B. C. Tedeschini, S. Savazzi, and M. Nicoli, "Weighted Average Consensus Algorithms in Distributed and Federated Learning," *IEEE Transactions on Network Science and Engineering*, vol. 12, no. 1, pp. 1–13, Jan. 2025.

[24] X. Cao, M. Fang, J. Liu, et al., "Fltrust: Byzantine-robust federated learning via trust bootstrapping," in *28th Annual Network and Distributed System Security Symposium (NDSS 2021)*, Virtual Conference, Feb. 2021.

[25] J. Zhang, Y. Han, X. Jing, et al., "DegaFL: Decentralized Gradient Aggregation for Cross-Silo Federated Learning," *IEEE Transactions on Parallel and Distributed Systems*, vol. 36, no. 2, pp. 212–225, Feb. 2025.

[26] T. Nishio and R. Yonetani, "Client Selection for Federated Learning with Heterogeneous Resources in Mobile Edge," in *IEEE*

International Conference on Communications (ICC 2019), Shanghai, China: IEEE, May 2019.

[27] J. Xu and H. Wang, "Client Selection and Bandwidth Allocation in Wireless Federated Learning Networks: A Long-Term Perspective," *IEEE Transactions on Wireless Communications*, vol. 20, no. 2, pp. 1188–1200, Feb. 2021.

[28] Y. Wang, J. Zhu, Y. Mao, et al., "Hierarchical Federated Edge Learning over Space-Air-Ground Integrated Networks," in *IEEE Global Communications Conference (GLOBECOM 2023)*, Kuala Lumpur, Malaysia: IEEE, Dec. 2023, pp. 190–196.

[29] E. Diao, J. Ding, and V. Tarokh, "HeteroFL: Computation and Communication Efficient Federated Learning for Heterogeneous Clients," in *9th International Conference on Learning Representations (ICLR 2021), Poster Session*, Virtual Conference: OpenReview.net, May 2021.

[30] M. Kim, S. Yu, S. Kim, et al., "Depthfl: Depthwise federated learning for heterogeneous clients," in *11th International Conference on Learning Representations (ICLR 2023), Poster Session*, Kigali, Rwanda: OpenReview.net, May 2023.

[31] K. Pfeiffer, M. Rapp, R. Khalili, et al., "CoCoFL: Communication- and Computation-Aware Federated Learning via Partial NN Freezing and Quantization," *Transactions on Machine Learning Research*, Jun. 2023.

[32] Z. Chen, C. Jia, M. Hu, et al., "FlexFL: Heterogeneous Federated Learning via APoZ-Guided Flexible Pruning in Uncertain Scenarios," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 43, no. 11, pp. 4069–4080, Nov. 2024.

[33] L. Cui, X. Su, Y. Zhou, et al., "Toward Optimized Federated Learning With Compressed Communications by Rate Adaption," *IEEE Transactions on Networking*, vol. 33, pp. 1–16, 2025.

[34] B. Xie, Y. Sun, S. Zhou, et al., "MOB-FL: Mobility-Aware Federated Learning for Intelligent Connected Vehicles," in *IEEE International Conference on Communications (ICC 2023)*, Rome, Italy: IEEE, May 2023, pp. 3951–3957.

[35] W. Wu, L. He, W. Lin, et al., "SAFA: A Semi-Asynchronous Protocol for Fast Federated Learning With Low Overhead," *IEEE Transactions on Computers*, vol. 70, no. 5, pp. 655–668, May 2021.

[36] Z. Wang, Z. Zhang, Y. Tian, et al., "Asynchronous Federated Learning Over Wireless Communication Networks," *IEEE Transactions on Wireless Communications*, vol. 21, no. 9, pp. 6961–6978, Sep. 2022.

[37] X. He, H. Huang, B. Xie, et al., "HiveFL: GAN-Empowered Semi-Asynchronous Federated Learning With Self-Determining Clients," *IEEE Transactions on Cognitive Communications and Networking*, vol. 11, 2025.

[38] J. Nguyen, K. Malik, H. Zhan, et al., "Federated Learning with Buffered Asynchronous Aggregation," in *25th International Conference on Artificial Intelligence and Statistics (AISTATS 2022)*, G. Camps-Valls, F. Ruiz, and I. Valera, Eds., vol. 151, Valencia, Spain: PMLR, Mar. 2022, pp. 3581–3607.

[39] Q. Ma, Y. Xu, H. Xu, et al., "FedSA: A Semi-Asynchronous Federated Learning Mechanism in Heterogeneous Edge Computing," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3654–3672, Dec. 2021.

[40] Z. Xia, M. Hu, D. Yan, et al., "CaBaFL: Asynchronous Federated Learning via Hierarchical Cache and Feature Balance," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 43, no. 11, pp. 4057–4068, Nov. 2024.

[41] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous Federated Optimization," arXiv, cs.DC 1903.03934, Mar. 2019.

[42] K. Mishchenko, F. Bach, M. Even, et al., "Asynchronous SGD Beats Minibatch SGD Under Arbitrary Delays," in *36th Conference on Neural Information Processing Systems (NeurIPS 2022)*, S. Koyejo, S. Mohamed, A. Agarwal, et al., Eds., vol. 35, New Orleans, LA: Curran Associates Inc., Nov. 2022, pp. 420–433.

[43] Y. Arjevani, O. Shamir, and N. Srebro, "A Tight Convergence Analysis for Stochastic Gradient Descent with Delayed Updates," in *31st International Conference on Algorithmic Learning Theory (ALT 2020)*, A. Kontorovich and G. Neu, Eds., vol. 117, San Diego, CA: PMLR, Feb. 2020, pp. 111–132.

[44] M. F. Pervej, R. Jin, and H. Dai, "Hierarchical Federated Learning in Wireless Networks: Pruning Tackles Bandwidth Scarcity and System Heterogeneity," *IEEE Transactions on Wireless Communications*, vol. 23, no. 9, pp. 11 417–11 432, Sep. 2024.

[45] C. Feng, H. H. Yang, D. Hu, et al., "Mobility-Aware Cluster Federated Learning in Hierarchical Wireless Networks," *IEEE Transactions on Wireless Communications*, vol. 21, no. 10, pp. 8441–8458, Oct. 2022.

[46] Z. Yang, S. Fu, W. Bao, et al., "Hierarchical Federated Learning With Momentum Acceleration in Multi-Tier Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 34, no. 10, pp. 2629–2641, Oct. 2023.

[47] J. Yang, W. Jiang, and L. Nie, "Hypernetworks-Based Hierarchical Federated Learning on Hybrid Non-IID Datasets for Digital Twin in Industrial IoT," *IEEE Transactions on Network Science and Engineering*, vol. 11, no. 2, pp. 1413–1423, Mar. 2024.

[48] A. Shamsian, A. Navon, E. Fetaya, et al., "Personalized Federated Learning using Hypernetworks," in *38th International Conference on Machine Learning (ICML 2021)*, M. Meila and T. Zhang, Eds., vol. 139, Virtual Conference: PMLR, Jul. 2021, pp. 9489–9502.

[49] Z. Wang, H. Xu, J. Liu, et al., "Resource-Efficient Federated Learning with Hierarchical Aggregation in Edge Computing," in *40th IEEE International Conference on Computer Communications (INFOCOM 2021)*, Virtual Conference: IEEE, May 2021, pp. 1–10.

[50] B. Wang, J. Fang, H. Li, et al., "Confederated Learning: Federated Learning With Decentralized Edge Servers," *IEEE Transactions on Signal Processing*, vol. 71, pp. 248–263, Apr. 2023.

[51] W. Zhang, D. Yang, W. Wu, et al., "Optimizing Federated Learning in Distributed Industrial IoT: A Multi-Agent Approach," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3688–3703, Dec. 2021.

[52] S. Luo, X. Chen, Q. Wu, et al., "HFEL: Joint Edge Association and Resource Allocation for Cost-Efficient Hierarchical Federated Edge Learning," *IEEE Transactions on Wireless Communications*, vol. 19, no. 10, pp. 6535–6548, Oct. 2020.

[53] S. Liu, G. Yu, X. Chen, et al., "Joint User Association and Resource Allocation for Wireless Hierarchical Federated Learning With IID and Non-IID Data," *IEEE Transactions on Wireless Communications*, vol. 21, no. 10, pp. 7852–7866, Oct. 2022.

[54] X. Liu, S. Wang, Y. Deng, et al., "Adaptive Federated Pruning in Hierarchical Wireless Networks," *IEEE Transactions on Wireless Communications*, vol. 23, no. 6, pp. 5985–5999, Jun. 2024.

[55] L. Su, R. Zhou, N. Wang, et al., "Low-Latency Hierarchical Federated Learning in Wireless Edge Networks," *IEEE Internet of Things Journal*, vol. 11, no. 4, pp. 6943–6960, Feb. 2024.

[56] E. Hellinger, "Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen.," *Journal für die reine und angewandte Mathematik*, vol. 136, pp. 210–271, 1909.

[57] A. A. Abdellatif, N. Mhaisen, A. Mohamed, et al., "Communication-efficient hierarchical federated learning for IoT heterogeneous systems with imbalanced data," *Elsevier Future Generation Computer Systems*, vol. 128, pp. 406–419, Mar. 2022.

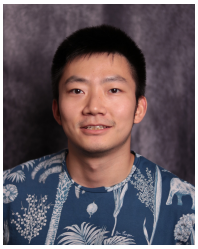
[58] A. Krizhevsky, "Learning multiple layers of features from tiny images," University of Toronto, Toronto, Canada, Technical Report TR-2009, Apr. 2009.

[59] M. Wu, T. Langerak, O. Hilliges, et al., "Using Synthetic Data in Supervised Learning for Robust 5-DoF Magnetic Marker Localization," *IEEE Transactions on Magnetics*, vol. 60, no. 8, pp. 1–11, Aug. 2024.

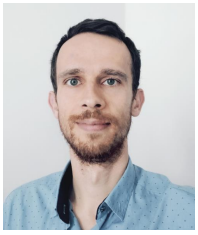
[60] F. Iandola, S. Han, M. Moskewicz, et al., "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size," DeepScale, arXiv 1602.07360, Nov. 2016.

[61] K. He, X. Zhang, S. Ren, et al., "Deep Residual Learning for Image Recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016)*, Las Vegas, NV: IEEE, Jun. 2016.

[62] F. Fritsch and R. Carlson, "Monotone piecewise cubic interpolation," *SIAM Journal on Numerical Analysis*, vol. 17, no. 2, pp. 238–246, Apr. 1980.



Mengfan Wu received a bachelor degree in telecommunications from Tongji University, Shanghai, in 2018 and received his M.Sc. degree in Electrical Engineering and Information Technology from ETH Zurich in 2021. He is a doctoral student at Huawei Munich Research Center and TU Berlin. His main research interests are federated learning algorithms and machine learning solutions for high-frequency telecommunications.



Mate Boban received the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, USA, in 2012. He is a technical expert with Huawei Technologies, Munich Research Center, Germany. He is currently a Co-Chair of WG1 (Radio Channels) in COST action CA20120 INTERACT and a Vice-Chair of the ETSI ISG on THz communications. His current research interests include machine learning applied to wireless communications and channel modeling.

Falko Dressler (Fellow, IEEE) is full professor and Chair for Telecommunication Networks at the School of Electrical Engineering and Computer Science, TU Berlin. He received his M.Sc. and Ph.D. degrees from the Dept. of Computer Science, University of Erlangen in 1998 and 2003, respectively. Dr. Dressler has been associate editor-in-chief for IEEE Trans. on Network Science and Engineering, IEEE Trans. on Mobile Computing and Elsevier Computer Communications as well as an editor for journals such as IEEE/ACM Trans. on Networking, Elsevier Ad Hoc Networks, and Elsevier Nano Communication Networks. He has been chairing conferences such as IEEE INFOCOM, ACM MobiSys, ACM MobiHoc, IEEE VNC, IEEE GLOBECOM. He authored the textbooks Self-Organization in Sensor and Actor Networks published by Wiley & Sons and Vehicular Networking published by Cambridge University Press. He has been an IEEE Distinguished Lecturer as well as an ACM Distinguished Speaker. Dr. Dressler is an IEEE Fellow, an ACM Fellow, and an AAIA Fellow. He is a member of the German National Academy of Science and Engineering (acatech). He has been serving on the IEEE COMSOC Conference Council and the ACM SIGMOBILE Executive Committee. His research objectives include next generation wireless communication systems in combination with distributed machine learning and edge computing for improved resiliency. Application domains include the internet of things, cyber-physical systems, and the internet of bio-nano-things.

APPENDIX A
DETAILS OF PROOF OF CONVERGENCE ANALYSIS

A.1 Bounding Difference between Global Model $\mathbf{x}^{p-1}$ and Client's Intermediate Model $\mathbf{x}_{c,\,i}^{s,\,t}$ in Equation (9)

$$
\begin{aligned}
&\left\|\nabla F\left(\mathbf{x}^{p-1}\right)-g_i\left(\mathbf{x}_{c,i}^{s,t}\right)\right\|^2\\
=&\left\|\nabla F\left(\mathbf{x}_{c,i}^{0,0}\right)-g_i\left(\mathbf{x}_{c,i}^{s,t}\right)\right\|^2\\
=&\left\|\nabla F\left(\mathbf{x}_{c,i}^{0,0}\right)-\nabla F_c\left(\mathbf{x}_{c,i}^{s,0}\right)+\nabla F_c\left(\mathbf{x}_{c,i}^{s,0}\right)-g_i\left(\mathbf{x}_{c,i}^{s,t}\right)\right\|^2\\
=&\left\|\nabla F\left(\mathbf{x}_{c,i}^{0,0}\right)-\nabla F_c\left(\mathbf{x}_{c,i}^{s,0}\right)+\sum_{\tau=0}^{t-1}\left[\nabla F_c\left(\mathbf{x}_{c,i}^{s,\tau}\right)-\nabla F_c\left(\mathbf{x}_{c,i}^{s,\tau+1}\right)\right]+\nabla F_c\left(\mathbf{x}_{c,i}^{s,t}\right)-\nabla F_i\left(\mathbf{x}_{c,i}^{s,t}\right)+\nabla F_i\left(\mathbf{x}_{c,i}^{s,t}\right)-g_i\left(\mathbf{x}_{c,i}^{s,t}\right)\right\|^2\\
\leq&(t+3)\cdot\sum_{\tau=0}^{t-1}\left\|\nabla F_c\left(\mathbf{x}_{c,i}^{s,\tau}\right)-\nabla F_c\left(\mathbf{x}_{c,i}^{s,\tau+1}\right)\right\|^2\\
+&(t+3)\cdot\left[\left\|\nabla F\left(\mathbf{x}_{c,i}^{0,0}\right)-\nabla F_c\left(\mathbf{x}_{c,i}^{s,0}\right)\right\|^2+\left\|\nabla F_c\left(\mathbf{x}_{c,i}^{s,t}\right)-\nabla F_i\left(\mathbf{x}_{c,i}^{s,t}\right)\right\|^2+\left\|\nabla F_i\left(\mathbf{x}_{c,i}^{s,t}\right)-g_i\left(\mathbf{x}_{c,i}^{s,t}\right)\right\|^2\right]\\
\leq&(t+3)\cdot\left[\sum_{\tau=0}^{t-1}L_c^2\left\|\mathbf{x}_{c,i}^{s,\tau}-\mathbf{x}_{c,i}^{s,\tau+1}\right\|^2+\left\|\nabla F\left(\mathbf{x}_{c,i}^{0,0}\right)-\nabla F_c\left(\mathbf{x}_{c,i}^{s,0}\right)\right\|^2+\epsilon_{i,c}^2+\eta^2\right]
\end{aligned}
\tag{12}
$$

Expand the second term further, we have

$$
\begin{aligned}
&\left\|\nabla F\left(\mathbf{x}_{c,i}^{0,0}\right)-\nabla F_c\left(\mathbf{x}_{c,i}^{s,0}\right)\right\|^2\\
=&\left\|\nabla F\left(\mathbf{x}_{c,i}^{0,0}\right)-\nabla F_c\left(\mathbf{x}_{c,i}^{0,0}\right)+\nabla F_c\left(\mathbf{x}_{c,i}^{0,0}\right)-\nabla F_c\left(\mathbf{x}_{c,i}^{s,0}\right)\right\|^2\\
=&\left\|\nabla F\left(\mathbf{x}_{c,i}^{0,0}\right)-\nabla F_c\left(\mathbf{x}_{c,i}^{0,0}\right)+\sum_{\sigma=0}^{s-1}\left[\nabla F_c\left(\mathbf{x}_{c,i}^{\sigma,0}\right)-\nabla F_c\left(\mathbf{x}_{c,i}^{\sigma+1,0}\right)\right]\right\|^2\\
\leq&(s+1)\cdot\left[\sum_{\sigma=0}^{s-1}\left\|\nabla F_c\left(\mathbf{x}_{c,i}^{\sigma,0}\right)-\nabla F_c\left(\mathbf{x}_{c,i}^{\sigma+1,0}\right)\right\|^2+\left\|\nabla F\left(\mathbf{x}_{c,i}^{0,0}\right)-\nabla F_c\left(\mathbf{x}_c^{0,0}\right)\right\|^2\right]\\
\leq&(s+1)\cdot\left[\sum_{\sigma=0}^{s-1}L_c^2\left\|\mathbf{x}_{c,i}^{\sigma,0}-\mathbf{x}_{c,i}^{\sigma+1,0}\right\|^2+\epsilon_{c,g}^2\right]
\end{aligned}
\tag{13}
$$

Decomposing the difference between the models of two neighboring cluster rounds, based on $\sum_{i'\in N_c}\pi_{i'}=1$ and using $\left\|\sum_{i'\in N_c}\pi_{i'}k_{i'}\right\|^2\leq\sum_{i'\in N_c}\pi_{i'}\|k_{i'}\|^2$:

$$
\begin{aligned}
&\left\|\mathbf{x}_{c,i}^{\sigma,0}-\mathbf{x}_{c,i}^{\sigma+1,0}\right\|^2\\
=&\left\|-\sum_{i'\in N_c}\pi_{i'}\cdot\gamma\sum_{\tau'=0}^{h_{i'}-1}\nabla g_{i'}\left(\mathbf{x}_{c,i'}^{\sigma,\tau'}\right)\right\|^2\\
\leq&\gamma^2\sum_{i'\in N_c}\pi_{i'}\left\|\sum_{\tau'=0}^{h_{i'}-1}\nabla g_{i'}\left(\mathbf{x}_{c,i'}^{\sigma,\tau'}\right)\right\|^2\\
\leq&\gamma^2\sum_{i'\in N_c}\pi_{i'}h_{i'}\sum_{\tau'=0}^{h_{i'}-1}\left\|\nabla g_{i'}\left(\mathbf{x}_{c,i'}^{\sigma,\tau'}\right)\right\|
\end{aligned}
\tag{14}
$$

Put Equation (14) back into Equation (13), we get:

$$
\begin{aligned}
&\left\|\nabla F\left(\mathbf{x}_{c,i}^{0,0}\right)-\nabla F_c\left(\mathbf{x}_{c,i}^{s,0}\right)\right\|^2\\
\leq&(s+1)\cdot\left[\sum_{\sigma=0}^{s-1}L_c^2\left\|\mathbf{x}_{c,i}^{\sigma,0}-\mathbf{x}_{c,i}^{\sigma+1,0}\right\|^2+\epsilon_{c,g}^2\right]\\
\leq&(s+1)\cdot\left[L_c^2\gamma^2\sum_{\sigma=0}^{s-1}\sum_{i'\in N_c}\pi_{i'}h_{i'}\sum_{\tau'=0}^{h_{i'}-1}\left\|\nabla g_{i'}\left(\mathbf{x}_{c,i'}^{\sigma,\tau'}\right)\right\|^2+\epsilon_{c,g}^2\right]
\end{aligned}
\tag{15}
$$

Integrate Equation (15) into Equation (12), and based on $\left\|\mathbf{x}_{c,i}^{s,\tau}-\mathbf{x}_{c,i}^{s,\tau+1}\right\|^2=\gamma^2\cdot\left\|\nabla g_i\left(\mathbf{x}_{c,i}^{s,\tau}\right)\right\|^2$, we can bound $\left\|\nabla F\left(\mathbf{x}^{p-1}\right)-g_i\left(\mathbf{x}_{c,i}^{s,t}\right)\right\|^2$ by:

$$
\begin{aligned}
&\left\|\nabla F\left(\mathbf{x}^{p-1}\right)-g_i\left(\mathbf{x}_{c,i}^{s,t}\right)\right\|^2\\
\leq&(t+3)\cdot\left[\sum_{\tau=0}^{t-1}L_c^2\left\|\mathbf{x}_{c,i}^{s,\tau}-\mathbf{x}_{c,i}^{s,\tau+1}\right\|^2+\left\|\nabla F\left(\mathbf{x}_{c,i}^{0,0}\right)-\nabla F_c\left(\mathbf{x}_{c,i}^{s,0}\right)\right\|^2+\epsilon_{i,c}^2+\eta^2\right]\\
\leq&(t+3)\cdot\left[\sum_{\tau=0}^{t-1}L_c^2\gamma^2\cdot\left\|\nabla g_i\left(\mathbf{x}_{c,i}^{s,\tau}\right)\right\|^2+\epsilon_{c,i}^2+\eta^2\right]+(t+3)(s+1)\cdot\left[L_c^2\gamma^2\sum_{\sigma=0}^{s-1}\sum_{i'\in N_c}\pi_{i'}h_{i'}\sum_{\tau'=0}^{h_{i'}-1}\left\|\nabla g_{i'}\left(\mathbf{x}_{c,i'}^{\sigma,\tau'}\right)\right\|^2+\epsilon_{c,g}^2\right]
\end{aligned}
\tag{16}
$$

A.2 Bounding Difference between $F\left(\mathbf{x}^p\right)$ and $F\left(\mathbf{x}^{p-1}\right)$ in Equation (7)

Decomposing the superpositioned terms in the vector product, we get:

$$
\begin{aligned}
&F\left(\mathbf{x}^p\right)-F\left(\mathbf{x}^{p-1}\right)\\
\leq&-\gamma\cdot\sum_{c\in C^p}\rho_c\sum_{s=0}^{r_c-1}\sum_{i\in N_c}\pi_i\sum_{t=0}^{h_i-1}\left\langle\nabla F\left(\mathbf{x}^{p-1}\right),g_i\left(\mathbf{x}_{c,i}^{s,t}\right)\right\rangle+\frac{L}{2}\left\|\gamma\cdot\sum_{c\in C^p}\rho_c\sum_{s=0}^{r_c-1}\sum_{i\in N_c}\pi_i\sum_{t=0}^{h_i-1}g_i\left(\mathbf{x}_{c,i}^{s,t}\right)\right\|^2
\end{aligned}
\tag{17}
$$

Based on $\sum_{i'\in N_c}\pi_{i'}=1$ and $\sum_{c\in C^p}\rho_c=1$, using $\left\|\sum_{i'\in N_c}\pi_{i'}k_{i'}\right\|^2\leq\sum_{i'\in N_c}\pi_{i'}\|k_{i'}\|^2$ and $\left\|\sum_{c\in C^p}\rho_c k_c\right\|^2\leq\sum_{c\in C^p}\rho_c\|k_c\|^2$, we rearrage the last term into:

$$
\begin{aligned}
&\frac{L}{2}\left\|\gamma\cdot\sum_{c\in C}\rho_c\sum_{s=0}^{r_c-1}\sum_{i\in N_c}\pi_i\sum_{t=0}^{h_i-1}g_i\left(\mathbf{x}_c^{s,t}\right)\right\|^2\\
\leq&\frac{L\gamma^2}{2}\sum_{c\in C}\rho_c\left\|\sum_{i\in N_c}\pi_i\sum_{s=0}^{r_c-1}\sum_{t=0}^{h_i-1}g_i\left(\mathbf{x}_c^{s,t}\right)\right\|^2\\
\leq&\frac{L\gamma^2}{2}\sum_{c\in C}\rho_c\sum_{i\in N_c}\pi_i\left\|\sum_{s=0}^{r_c-1}\sum_{t=0}^{h_i-1}g_i\left(\mathbf{x}_{c,i}^{s,t}\right)\right\|^2\\
\leq&\frac{L\gamma^2}{2}\sum_{c\in C}\rho_c\sum_{i\in N_c}\pi_i\left\|\sum_{s=0}^{r_c-1}\sum_{t=0}^{h_i-1}g_i\left(\mathbf{x}_{c,i}^{s,t}\right)\right\|^2
\end{aligned}
\tag{18}
$$

Integrate Equation (8) and Equation (18) into Equation (17), we get:

$$
\begin{aligned}
&F\left(\mathbf{x}^p\right)-F\left(\mathbf{x}^{p-1}\right)\\
\leq&\frac{\gamma}{2}\sum_{c\in C^p}\rho_c\sum_{s=0}^{r_c-1}\sum_{i\in N_c}\pi_i\sum_{t=0}^{h_i-1}\left[\left\|\nabla F\left(\mathbf{x}^{p-1}\right)-g_i\left(\mathbf{x}_{c,i}^{s,t}\right)\right\|^2-\left\|\nabla F\left(\mathbf{x}^{p-1}\right)\right\|^2-\left\|g_i\left(\mathbf{x}_{c,i}^{s,t}\right)\right\|^2\right]\\
+&\frac{L\gamma^2}{2}\sum_{c\in C^p}\rho_c\sum_{i\in N_c}\pi_i\left\|\sum_{s=0}^{r_c-1}\sum_{t=0}^{h_i-1}g_i\left(\mathbf{x}_{c,i}^{s,t}\right)\right\|^2\\
\leq&\frac{\gamma}{2}\sum_{c\in C^p}\rho_c\sum_{s=0}^{r_c-1}\sum_{i\in N_c}\pi_i\sum_{t=0}^{h_i-1}\left[\left\|\nabla F\left(\mathbf{x}^{p-1}\right)-g_i\left(\mathbf{x}_{c,i}^{s,t}\right)\right\|^2-\left\|\nabla F\left(\mathbf{x}^{p-1}\right)\right\|^2\right]\\
+&\frac{\gamma}{2}\sum_{c\in C^p}\rho_c\sum_{i\in N_c}\pi_i\left[-\sum_{s=0}^{r_c-1}\sum_{t=0}^{h_i-1}\left\|g_i\left(\mathbf{x}_{c,i}^{s,t}\right)\right\|^2+\gamma L\left\|\sum_{s=0}^{r_c-1}\sum_{t=0}^{h_i-1}g_i\left(\mathbf{x}_{c,i}^{s,t}\right)\right\|^2\right]
\end{aligned}
\tag{19}
$$

To eliminate the second term, we choose γ so that $\gamma L r_c h_i\leq 0$ is always satisfied, namely $\gamma\leq\frac{1}{L(r_c h_i)_{\max}}$.
Plug the result of Equation (16) into Equation (19) and use Assumption 4.3:

$$
\begin{aligned}
&F\left(\mathbf{x}^{p}\right)-F\left(\mathbf{x}^{p-1}\right)\\
\leq&\frac{\gamma}{2}\sum_{c\in C^{p}}\rho_{c}\sum_{s=0}^{r_{c}-1}\sum_{i\in N_{c}}\pi_{i}\sum_{t=0}^{h_{i}-1}\left[\left\|\nabla F\left(\mathbf{x}^{p-1}\right)-g_{i}\left(\mathbf{x}_{c,i}^{s,t}\right)\right\|^{2}-\left\|\nabla F\left(\mathbf{x}^{p-1}\right)\right\|^{2}\right]\\
\leq&\frac{\gamma}{2}\sum_{c\in C^{p}}\rho_{c}\sum_{i\in N_{c}}\pi_{i}\sum_{s=0}^{r_{c}-1}\sum_{t=0}^{h_{i}-1}(t+3)\cdot\left[\sum_{\tau=0}^{t-1}L_{c}^{2}\gamma^{2}\cdot G^{2}+\epsilon_{c,i}^{2}+\eta^{2}\right]\\
+&\frac{\gamma}{2}\sum_{c\in C^{p}}\rho_{c}\sum_{i\in N_{c}}\pi_{i}\sum_{s=0}^{r_{c}-1}\sum_{t=0}^{h_{i}-1}(t+3)(s+1)\cdot\left[L_{c}^{2}\gamma^{2}\sum_{\sigma=0}^{s-1}\sum_{i'\in C}\pi_{i'}h_{i'}\sum_{\tau'=0}^{h_{i'}-1}G^{2}+\epsilon_{c,g}^{2}\right]\\
-&\frac{\gamma}{2}\sum_{c\in C^{p}}\rho_{c}\sum_{s=0}^{r_{c}-1}\sum_{i\in N_{c}}\pi_{i}\sum_{t=0}^{h_{i}-1}\left\|\nabla F\left(\mathbf{x}^{p-1}\right)\right\|^{2}
\end{aligned}
\tag{20}
$$

For simplicity expression, set $w^{p}=\gamma\sum_{c\in C^{p}}\rho_{c}\sum_{i\in N_{c}}\pi_{i}r_{c}h_{i}$
The first group of coefficients for G^{2} can be computed as:

$$
\begin{aligned}
&\frac{\gamma}{2}\sum_{c\in C^{p}}\rho_{c}\sum_{i\in N_{c}}\pi_{i}\sum_{s=0}^{r_{c}-1}\sum_{t=0}^{h_{i}-1}(t+3)\cdot\sum_{\tau=0}^{t-1}L_{c}^{2}\gamma^{2} &(21)\\
=&\frac{\gamma}{2}\sum_{c\in C^{p}}\rho_{c}\sum_{i\in N_{c}}\pi_{i}L_{c}^{2}\gamma^{2}\cdot\frac{1}{3}h_{i}(h_{i}-1)(h_{i}+4)r_{c}\\
=&\frac{w^{p}}{2}L_{c}^{2}\gamma^{2}\cdot\frac{1}{3}(h_{i}-1)(h_{i}+4) &(22)
\end{aligned}
$$

The second group of coefficients for G^{2} can be computed as:

$$
\begin{aligned}
&\frac{\gamma}{2}\sum_{c\in C}\rho_{c}\sum_{i\in N_{c}}\pi_{i}\sum_{s=0}^{r_{c}-1}\sum_{t=0}^{h_{i}-1}(t+3)(s+1)\cdot\left[L_{c}^{2}\gamma^{2}\sum_{\sigma=0}^{s-1}\sum_{i'\in C}\pi_{i'}h_{i'}\sum_{\tau'=0}^{h_{i'}-1}1\right]\\
=&\frac{\gamma}{2}\sum_{c\in C}\rho_{c}\sum_{i\in N_{c}}\pi_{i}\sum_{s=0}^{r_{c}-1}\sum_{t=0}^{h_{i}-1}(t+3)(s+1)\cdot L_{c}^{2}\gamma^{2}\sum_{\sigma=0}^{s-1}\sum_{i'\in C}\pi_{i'}h_{i'}^{2}\\
=&\frac{\gamma}{2}\sum_{c\in C}\rho_{c}\sum_{i\in N_{c}}\pi_{i}L_{c}^{2}\gamma^{2}\left(\sum_{i'\in C}\pi_{i'}h_{i'}^{2}\right)\frac{1}{6}h_{i}(h_{i}+5)r_{c}(r_{c}-1)(r_{c}+1)\\
=&\frac{w^{p}}{2}L_{c}^{2}\gamma^{2}\left(\sum_{i'\in C}\pi_{i'}h_{i'}^{2}\right)\frac{1}{6}(h_{i}+5)(r_{c}-1)(r_{c}+1) &(23)
\end{aligned}
$$

Coefficient for $\epsilon_{c,g}^{2}$ is:

$$
\begin{aligned}
&\frac{\gamma}{2}\sum_{c\in C}\rho_{c}\sum_{i\in N_{c}}\pi_{i}\sum_{s=0}^{r_{c}-1}\sum_{t=0}^{h_{i}-1}(t+3)(s+1)\\
=&\frac{\gamma}{2}\sum_{c\in C}\rho_{c}\sum_{i\in N_{c}}\pi_{i}\frac{1}{4}h_{i}(h_{i}+5)r_{c}(r_{c}+1)\\
=&\frac{w^{p}}{2}\frac{1}{4}(h_{i}+5)(r_{c}+1) &(24)
\end{aligned}
$$

Coefficient for $\epsilon_{c,i}^{2}+\eta^{2}$ is:

$$
\begin{aligned}
&\frac{\gamma}{2}\sum_{c\in C^{p}}\rho_{c}\sum_{i\in N_{c}}\pi_{i}\sum_{s=0}^{r_{c}-1}\sum_{t=0}^{h_{i}-1}(t+3)\\
\leq&\frac{\gamma}{2}\sum_{c\in C^{p}}\rho_{c}\sum_{i\in N_{c}}\pi_{i}\frac{1}{2}h_{i}(h_{i}+5)r_{c}\\
=&\frac{w^{p}}{2}\frac{1}{2}(h_{i}+5) &(25)
\end{aligned}
$$

To simplify the first group of coefficients for G^2, choose $\gamma \leq \frac{1}{(h_{\max}+5)L_c}$, we can get

$$\begin{aligned} & L_c^2\gamma^2 \cdot \frac{1}{3}(h_i-1)(h_i+4) \\ \leq & \frac{1}{3}\frac{1}{(h_{\max}+5)^2 L_c^2} \cdot L_c^2 \cdot (h_i-1)(h_i+4) < \frac{1}{3} \end{aligned} \tag{26}$$

To simplify the second group of coefficients for G^2, with $\overline{h_c^2} = \sum_{i' \in C} \pi_{i'} h_{i'}^2$, choose $\gamma \leq \frac{\sqrt{6}}{\left(\sqrt{\overline{h_c^2}\cdot(h_i+5)}\cdot r_c\right)_{\max} L_c}$, we can get

$$\begin{aligned} & L_c^2\gamma^2\overline{h_c^2}\frac{1}{6}(h_i+5)(r_c-1)(r_c+1) \\ \leq & L_c^2\overline{h_c^2}\frac{1}{6}(h_i+5)(r_c-1)(r_c+1) \cdot \frac{6}{\left(\overline{h_c^2}\cdot(h_i+5)\cdot r_c^2\right)_{\max} L_c^2} < 1 \end{aligned} \tag{27}$$

Integrate all the simplified coefficients from Equation (21) to Equation (27) into Equation (20) and rearrage, we can get:

$$\begin{aligned} & \frac{w^p}{2}\left\|F\left(\mathbf{x}^{p-1}\right)\right\|^2 \leq F\left(\mathbf{x}^{p-1}\right) - F\left(\mathbf{x}^p\right) \\ & + \frac{w^p}{2}\left(\frac{1}{3}+1\right)G^2 + \frac{w^p}{2}\frac{\gamma(h_i+5)(r_c+1)}{4\gamma}\epsilon_{c,g}^2 + \frac{w^p}{2}\frac{\gamma(h_i+5)}{2\gamma}(\epsilon_{i,c}^2+\eta^2) \end{aligned} \tag{28}$$

A.3 Bound on the Average of Global Round Gradients

Therefore, for each global round p, we have:

$$\begin{aligned} \left\|\nabla F\left(\mathbf{x}^{p-1}\right)\right\|^2 \leq & \frac{2}{w^p}\left[F\left(\mathbf{x}^{p-1}\right) - F\left(\mathbf{x}^p\right)\right] \\ & + \frac{4}{3}G^2 + \frac{(r_{\max}+1)(h_{\max}+5)}{4}\cdot\epsilon_{c,g}^2 + \frac{(h_{\max}+5)}{2}\cdot\left(\epsilon_{c,i}^2+\eta^2\right) \end{aligned} \tag{29}$$

Adding up the bound of gradient in each round and average, the bound of the averaged gradient scale is:

$$\begin{aligned} \frac{1}{P}\sum_{p=1}^{P}\left\|\nabla F\left(\mathbf{x}^{p-1}\right)\right\|^2 \leq & \frac{1}{P}\sum_{p=1}^{P}\frac{2}{w^p}\left[F\left(\mathbf{x}^{p-1}\right) - F\left(\mathbf{x}^p\right)\right] + \frac{4}{3}G^2 \\ & + \frac{(r_{\max}+1)(h_{\max}+5)}{4}\cdot\epsilon_{c,g}^2 + \frac{(h_{\max}+5)}{2}\cdot\left(\epsilon_{c,i}^2+\eta^2\right) \end{aligned} \tag{30}$$

This does not eliminate the middle terms ($F\left(\mathbf{x}^p\right)$, $p = 1, \ldots, P-1$), but enables us to have an accurate testing of the bound for $\left\|\nabla F\left(\mathbf{x}^{p-1}\right)\right\|^2$.

Based on the setting of $\gamma \leq \frac{1}{(h_{\max}+5)L_c}$ again, we have $\gamma(h_i+5)(r_c+1) \leq \frac{(r_{\max}+1)}{L_c}$ and $\gamma(h_i+5) \leq \frac{1}{L_c}$. Using the upper and lower bound of w^p: $w^{\max} = \max_{p=1}^{P} w^p$, $w^{\min} = \min_{p=1}^{P} w^p$ to bound the right/left side of Equation (28):

$$\begin{aligned} & \frac{w^{\min}}{2}\left\|F\left(\mathbf{x}^{p-1}\right)\right\|^2 \leq F\left(\mathbf{x}^{p-1}\right) - F\left(\mathbf{x}^p\right) \\ & + \frac{w^{\max}}{2}\left(\frac{1}{3}+1\right)G^2 + \frac{w^{\max}}{2}\frac{(r_{\max}+1)}{4\gamma L_c}\epsilon_{c,g}^2 + \frac{w^{\max}}{2}\frac{1}{2\gamma L_c}(\epsilon_{i,c}^2+\eta^2) \end{aligned} \tag{31}$$

Divide both sides by $w^{\min}/2$, add up the term from $p = 1$ to P, and finally divide both sides by P, we can get:

$$\frac{1}{P}\sum_{p=1}^{P}\left\|F\left(\mathbf{x}^{p-1}\right)\right\|^2 \leq \frac{2}{Pw^{\min}}\left[F\left(\mathbf{x}^0\right) - F\left(\mathbf{x}^P\right)\right] + \frac{w^{\max}}{w^{\min}}\left[\frac{4}{3}G^2 + \frac{(r_{\max}+1)}{4\gamma L_c}\epsilon_{c,g}^2 + \frac{1}{2\gamma L_c}(\epsilon_{i,c}^2+\eta^2)\right] \tag{32}$$

APPENDIX B
MACHINE LEARNING MODEL STRUCTURE

B.1 Convolutional Neural Network for CIFAR10 Classification

TABLE 13: Model Structure for CNN used for CIFAR10 classification

Layer	Parameters	Input Layer
conv1	out_channel=6, kernel_size=5, padding=0	data
activation1	ReLU	conv1
pooling1	pool_size=2, stride=2	activation1
conv2	out_channel=16, kernel_size=5, padding=0	pooling1
activation2	ReLU	conv2
pooling2	pool_size=2, stride=2	activation2
flatten	/	pooling2
fully_connected1	output=120	flatten
activation3	ReLU	fully_connected1
fully_connected2	output=84	activation3
activation4	ReLU	fully_connected2
fully_connected3	output=10	activation4

B.2 Multi-layer Perceptron for Magnet Positioning

TABLE 14: Model Structure for MLP used for Magnet Positioning

Layer	Parameters	Input Layer
Input_transformation	Root order: 3	data (48×1)
fully_connected1	output=2048	Input_transformation
activation1	ReLU	fully_connected1
fully_connected2	output=2048	activation1
activation2	ReLU	fully_connected2
fully_connected3	output=2048	activation2
activation3	ReLU	fully_connected3
fully_connected4	output=6	activation3

APPENDIX C
HYPER-PARAMETER SEARCHING

C.1 Grid Search for Parameters in Swap Search Algorithm 2

We conduct a grid search over the parameters z and n_{swap} in the swap search algorithm for Solution 1.1 and Solution 3 under two different data partitioning schemes:

1) optimization-based partition with $I_c = 4$, $\sigma = 750$;
2) per-class partition following a Dirichlet distribution with $\alpha = 0.1$.

For each pair of (z, n_{swap}), we run experiments five times with different random seeds and corresponding data partitions. The final test accuracy averaged over these five trials is shown in the heatmap in Figure 14. The results suggest that the best-performing parameter choices are somewhat sporadic across different settings. To further evaluate stability, we also report the average test accuracy across all global rounds in Figure 15.

Based on these results, we select $z = 1.05$, $n_{\text{swap}} = 80$ for Solution 1.1, and $z = 1.8$, $n_{\text{swap}} = 80$ for Solution 3, which provide consistently good performance across both data distributions.

We omit grid search for Solution 2 since this method implicitly allows longer cluster round times, which discourages additional local optimization steps and therefore does not yield results comparable to Solution 1.1 or Solution 3. In the experiments, $z = 1.05$, $n_{\text{swap}} = 80$ is used for Solution 2.

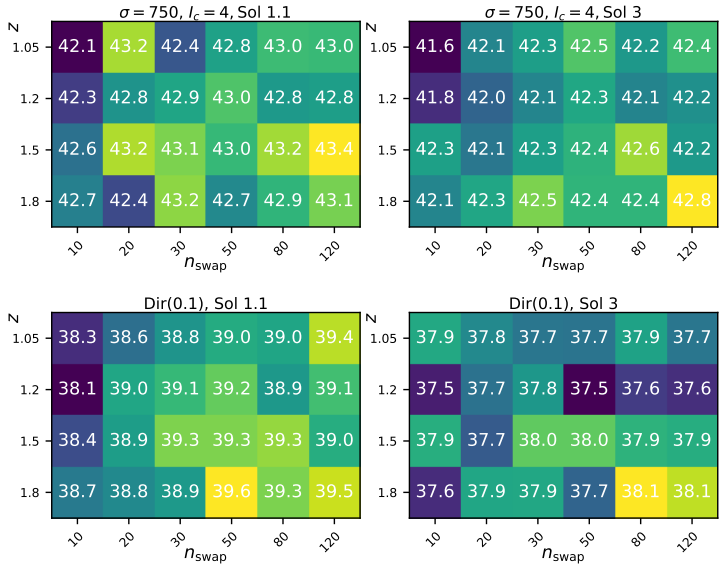


Fig. 14: Heatmap of final testing accuracy for different settings of z and n_{swap} in swap search algorithm.

C.2 Grid Search for Parameters in Spectral Clustering Section 5.4.1 and Section 5.4.2

For Benchmarks A and A+, we also conduct parameter tuning.

For Benchmark A, we search over the parameter space $\{0.5, 0.8, 1.0, 1.5, 2.0\}$ for σ. Each candidate value is evaluated over five random trials, and the averaged final test accuracies are visualized in the heatmap in Figure 16. Based on these results, we select $\sigma = 0.5$ for data distributions with $I_c = 4$, and $\sigma = 1.5$ for data distributions with $I_c = 7$ or those following a Dirichlet distribution.

For Benchmark A+, we perform a grid search over parameter pairs $(\sigma_t, \sigma_{ft.})$, again averaging results across five random trials per setting. The corresponding heatmap of final test accuracies is shown in Figure 17. From these experiments, we select $\sigma_t = 0.5, \sigma_{ft.} = 0.8$ for data distributions with $I_c = 4$; $\sigma_t = 1.0, \sigma_{ft.} = 0.8$ for those with $I_c = 7$; and $\sigma_t = 1.5, \sigma_{ft.} = 1.0$ for Dirichlet-distributed data.

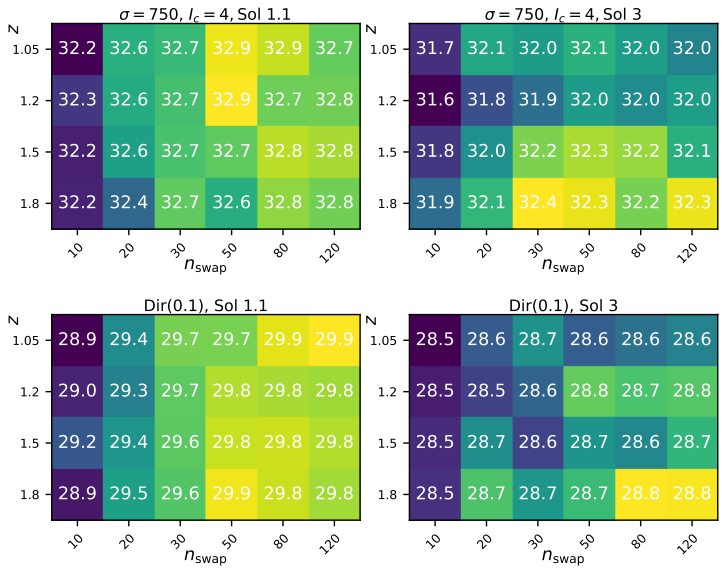


Fig. 15: Heatmap of averaged testing accuracy over all global rounds for different settings of z and n_{swap} in swap search algorithm.

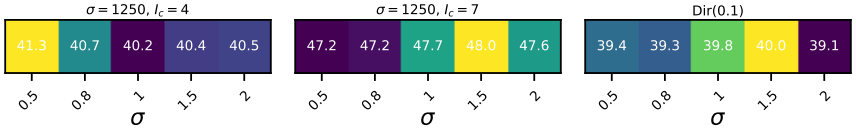


Fig. 16: Heatmap of final testing accuracy for different settings of σ in original spectral clustering solution.

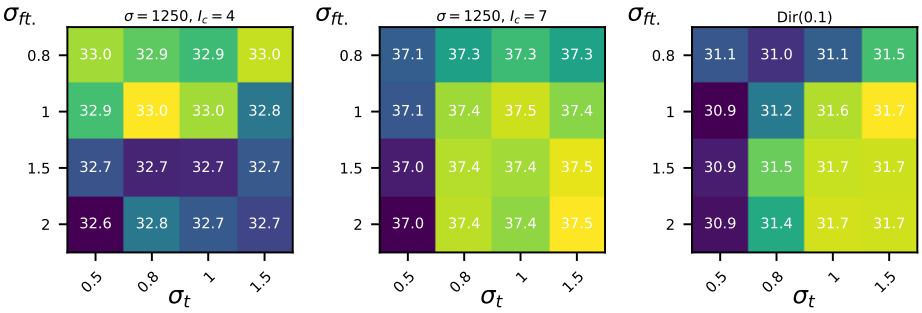


Fig. 17: Heatmap of final testing accuracy for different settings of σ_t and $\sigma_{ft.}$ in adapted spectral clustering.