Optimal Client Sampling in Federated Learning with Client-Level Heterogeneous Differential Privacy

Jiahao Xu
University of Nevada, Reno
jiahaox@unr.edu

Rui Hu
University of Nevada, Reno
ruihu@unr.edu

Olivera Kotevska
Oak Ridge National Laboratory
kotevskao@ornl.gov

Abstract

Federated Learning with client-level differential privacy (DP) provides a promising framework for collaboratively training models while rigorously protecting clients' privacy. However, classic approaches like DP-FedAvg struggle when clients have heterogeneous privacy requirements, as they must uniformly enforce the strictest privacy level across clients, leading to excessive DP noise and significant model utility degradation. Existing methods to improve the model utility in such heterogeneous privacy settings often assume a trusted server and are largely heuristic, resulting in suboptimal performance and lacking strong theoretical underpinnings. In this work, we address these challenges under a practical attack model where both clients and the server are honest-but-curious. We propose GDPFed, which partitions clients into groups based on their privacy budgets and achieves client-level DP within each group to reduce the privacy budget waste and hence improve the model utility. Based on the privacy and convergence analysis of GDPFed, we find that the magnitude of DP noise depends on both model dimensionality and the per-group client sampling ratios. To further improve the performance of GDPFed, we introduce GDPFed$^+$, which integrates model sparsification to eliminate unnecessary noise and optimizes per-group client sampling ratios to minimize convergence error. Extensive empirical evaluations on multiple benchmark datasets demonstrate the effectiveness of GDPFed$^+$, showing substantial performance gains compared with state-of-the-art methods.

1 Introduction

Traditional centralized Machine Learning (ML) frameworks require collecting all training data at a single node (e.g., a central server), raising significant privacy concerns. To mitigate this issue, Federated Learning (FL) [37] has emerged as a distributed ML paradigm that enables model training directly on decentralized data sources without transferring raw data. In FL, multiple local clients (e.g., edge devices) collaboratively train a shared global model under the coordination of a central server. Specifically, in each training round, the server sends the global model to a subset of clients, who update it using their private data. These model updates are then transmitted to the server, which aggregates them to refine the global model. This process continues until the global model converges.

Although the FL paradigm keeps sensitive training data on clients, recent studies have shown that adversaries can still infer private information through well-crafted inference attacks [17, 39, 43, 51, 61]. To mitigate privacy risks, differential privacy (DP) [14], a widely adopted standard for incorporating formal privacy guarantees, has been integrated into the FL algorithm [38]. In the context of FL, DP can be applied at two distinct protection levels: *record-level DP*, which protects individual data points within a client's dataset, and *client-level DP*, which protects the participation of a client (i.e., the client's entire dataset). This work focuses on achieving client-level differentially private FL (DPFL), as it typically yields better model utility than its record-level counterpart in

Preprint. Under review.

cross-device settings [25]. In the literature, client-level DPFL is implemented using the Gaussian mechanism [14], where each client's local model update is perturbed by adding calibrated Gaussian noise scaled according to a *uniform* privacy budget ϵ across all clients [5, 7, 25, 29, 50]. A smaller ϵ provides stronger privacy guarantees but requires injecting larger noise, which consequently leads to more severe model utility degradation. These perturbed model updates are typically aggregated using secure aggregation (e.g., [8]), which cryptographically ensures that the server can only access their sum without observing individual contributions. This dual protection yields a differentially private aggregated model update that prevents client-level privacy inference even with an adversarial server.

However, in practice, clients often have heterogeneous privacy preferences, necessitating support for heterogeneous DP (HDP) [26, 33]. In the literature, Liu et al. [32] formally introduced the problem of FL with heterogeneous DP (HDPFL), where each client naturally has an individual privacy budget reflecting their privacy needs. In this setting, ensuring record-level HDP is relatively straightforward, and numerous studies have proposed to improve model utility [7, 33, 35, 36, 48, 58]. In contrast, client-level HDPFL remains under-explored. To achieve client-level DP with heterogeneous privacy requirements, conventional approaches such as DP-FedAvg [38] must satisfy the most stringent privacy budget among all clients, which severely limits overall model utility. A more practical alternative partitions clients into groups and enforces client-level DP at the group level. To improve the model utility in this scenario, recent efforts include manually adjusting per-group client sampling ratios [29], adjusting training rounds per group [11], and mitigating the influence of noisy per-group updates [32]. However, these approaches assume a fully trusted server, which is often unrealistic in settings that are vulnerable to privacy inference attacks. Moreover, they primarily rely on heuristic methods without rigorous theoretical analysis to optimize the privacy-utility trade-off.

In this work, we aim to optimize the model utility in client-level HDPFL under a strong attack model where both the clients and the server are adversaries. We propose GDPFed, a novel client-level HDPFL approach in which clients are grouped by their privacy budgets, with client-level DP achieved at a group-wise level using each group's minimum privacy budget rather than the global minimum. This design enables higher model utility while respecting heterogeneous privacy preferences. Building on this, we theoretically investigate how to maximize model utility in GDPFed while maintaining rigorous privacy guarantees. Through privacy and convergence analysis, we identify two key factors that influence convergence errors under fixed privacy budgets: (1) model dimensionality, as DP noise must be added to each model parameter, increasing total noise with model size; and (2) per-group client sampling ratios, which has a privacy amplification effect on the guarantees. To reduce dimensionality-induced noise, we incorporate model sparsification into GDPFed, which eliminates less significant model parameters for each group with minimal utility drop. We then optimize the per-group client sampling ratios towards minimizing the convergence error, which extends GDPFed to GDPFed+ with improved model utility. In summary, we make the following contributions:

- We propose GDPFed, a novel *client-level* DPFL algorithm for environments where both server and clients are *honest-but-curious*. GDPFed is specifically designed to improve model utility when clients have heterogeneous privacy preferences. By achieving client-level DP at a group-wise level, our approach mitigates privacy budget waste inherent in HDP settings, improving the model utility. GDPFed builds upon FedAvg framework, enabling seamless integration into existing FL systems.
- To further improve the model utility, while preserving the privacy guarantees, we propose GDPFed$^+$, which integrates per-group model sparsification into GDPFed and optimizes the per-group client sampling ratios to minimize the impact of DP noise on the model utility. *To the best of our knowledge, this is the first work that optimizes client sampling ratios to enhance the privacy-utility trade-off in client-level HDP settings.*
- We conduct extensive empirical evaluations on multiple benchmark datasets of DPFL, thoroughly comparing our methods against state-of-the-art baseline methods. The results consistently demonstrate that GDPFed outperforms DP-FedAvg in HDP settings, while GDPFed$^+$ further improves the model utility under the same privacy guarantee.

2 Preliminary and Related Work

Attack Model. To achieve client-level DP, the literature typically assumes that the adversary is either *honest-but-curious* clients [7, 29, 32, 38] or, in a stronger setting, both the clients and the server [16, 25, 27, 50]. The adversary follows the prescribed FL protocol but remains curious about a

target client's private data and attempts to infer it from shared messages. In this work, we consider the latter, more challenging one.

Federated Learning and FedAvg. In a typical FL system, a set of n clients aim to collaboratively train a shared global model $\theta \in \mathbb{R}^d$ in an iterative manner under the coordination of a central server. Generally, the FL problem can be formulated as $\min_\theta (1/n) \sum_{i=1}^{n} f_i(\theta)$, where $f_i(\theta) = \mathbb{E}_{(z,y)\in D_i} l(\theta; z, y)$ represents the local learning objective of client i. Here, $l(\cdot)$ is the loss function, and (z, y) is a datapoint sampled from the local dataset D_i of client i. The classic method to solve the FL problem is known as Federated Averaging (FedAvg) [37]. Specifically, in each training round t, the server randomly selects a set of r clients $\mathcal{S}^t$ with a client sampling ratio $q \in (0, 1]$ without replacement to participate in the local training. Each client $i \in \mathcal{S}^t$ then downloads the latest global model θ^{t-1} from the server, refines the model for τ iterations towards optimizing its local objective to obtain an updated local model θ_i^t and then sends its local model updates $\Delta_i^t = \theta_i^t - \theta^{t-1}$ back to the server. The server refines the global model by averaging the local updates as $\theta^t = \theta^{t-1} + (1/r) \sum_{i\in\mathcal{S}^t} \Delta_i^t$. This process repeats for enough T rounds to ensure that the global model converges. Since the server receives individual model updates from clients in each round, it poses a significant privacy risk, as a curious server can infer sensitive information from these updates.

Differential Privacy. The DP mechanism [14, 41], especially the Gaussian mechanism (see the formal definition in Lemma 6), has been employed as a rigorous approach for mitigating privacy threats in FL [13, 25, 50]. We give the formal definition of classic (ϵ, δ)-DP in Definition 1.

Definition 1 ((ϵ, δ)-DP [14])**.** *Given privacy budget $\epsilon > 0$ and failure parameter $0 \leq \delta < 1$, a randomized mechanism $\mathcal{M}$ satisfies (ϵ, δ)-DP if for any two adjacent datasets D, D', any subset of outputs $O \subseteq$ range$(\mathcal{M})$ satisfies $\Pr[\mathcal{M}(D) \in O] \leq e^\epsilon \Pr[\mathcal{M}(D') \in O] + \delta$.*

In this work, as we consider client-level DP, we define the *adjacent datasets* by adding or removing the *entire* local dataset of a client in FL. The privacy budget ϵ defines the upper bound on privacy loss in a DP mechanism. A smaller ϵ indicates stronger privacy protection but requires injecting more intense noise into the learning process, which can significantly impact model performance. Additionally, the failure parameter δ quantifies the probability that the DP guarantee may be violated. When $\delta = 0$, the formulation (ϵ, δ)-DP simplifies to pure DP.

The standard (ϵ, δ)-DP provides a relatively loose composition bound, making it unsuitable for accurately tracking the cumulative privacy loss in complex iterative algorithms. Therefore, in this work, we adopt Rényi DP (RDP) [41], a relaxed variant of (ϵ, δ)-DP, to better quantify privacy loss over multiple rounds in DPFL. We provide the formal definition of RDP and its related properties used in this work in Appendix A.

Client-level DP-FedAvg. Compared with record-level DP [2, 33, 34, 54], which aims to protect every individual record in a client's dataset, client-level DP hides a single client's overall contribution. To achieve client-level DP under our attack mdoel, one can use DP-FedAvg [38]: before transmitting the local model update Δ_i^t to the server at round t, each selected client clips its model update with a clipping threshold C, and adds small amount of DP noise drawn from $\mathcal{N}\left(0, C^2\sigma^2/r \cdot \mathbf{I}^d\right)$, where σ^2 is the noise multiplier. Notably, the noise multiplier σ^2 must be carefully calibrated to ensure that DP-FedAvg satisfies (ϵ, δ)-DP after T training rounds. Theoretical analyses establish the relationship $\sigma^2 = \Omega(q^2/\epsilon)$ [1, 42], implying that satisfying a smaller privacy budget ϵ necessitates injecting larger noise. Furthermore, DP-FedAvg benefits from *privacy amplification* via client subsampling [6], where in each client is independently selected with probability q in every training round.

After perturbing their updates locally, clients encrypt these noisy updates using a secure aggregation protocol (e.g., [8]) and send them to the server. Secure aggregation is a commonly used practice in client-level DPFL [16, 25, 27, 50], ensuring that a curious server only observes the aggregated sum of clients' updates, without access to individual contributions. In this setting, the aggregated model update received by the server is already perturbed with Gaussian noise $\mathcal{N}\left(0, C^2\sigma^2 \cdot \mathbf{I}^d\right)$. Finally, the global model is refined with the perturbed aggregated updates. If the server is assumed to be trusted [11, 29, 32], these model clipping and perturbation operations can be directly applied to the aggregated model update on the server side to prevent clients from inferring private information. We present the detailed DP-FedAvg algorithm in Algorithm 2 in Appendix B.3.

The noise applied to model updates inherently reduces the utility of the global model. To mitigate this issue, numerous methods have been proposed, including model update regularization [5, 13, 50] to ensure more robust local updates, optimized client sampling [12, 47, 52] to select more informative

clients, and sparsification [13, 25] to remove unnecessary noise. However, these methods consider a homogeneous DP setting, where all clients share the same privacy preference. In contrast, an HDP setting where clients have heterogeneous privacy preferences is more realistic and better aligned with practical deployment scenarios.

FL with Heterogeneous Privacy Preferences. In practice, clients often have diverse privacy requirements due to varying policies or individual preferences, making it essential to consider FL under HDP [26]. Liu et al. [32] first formalized the problem of HDPFL, allowing each client to specify a unique privacy budget that reflects their preferences. In this setting, record-level HDP is straightforward to implement by calibrating the DP noise individually per client [7, 32, 33, 35, 48, 58]. For example, Boenisch et al. [7] proposed IDP-FedAvg, which assigns data sampling ratios and clipping thresholds based on each record's privacy budget. However, achieving client-level HDP, where the goal is to protect a single client's contribution from being inferred, poses greater challenges.

Standard approaches such as DP-FedAvg [38] in this heterogeneous setting have to calibrate noise to satisfy the most stringent privacy requirement among clients, leading to excessive noise for clients with more relaxed privacy preferences and thus poor model utility [29]. A more privacy-efficient approach is to partition clients into groups based on their privacy budgets and ensure client-level DP within each group [11, 29, 32]. For instance, Kiani et al. [29] proposed a dynamic HDPFL framework where clients in different groups consume less privacy budget in early training rounds. Note that this method also proposes the formulation of client sampling ratio optimization, but does not address it in its design. Instead, they manually tune each group's sampling ratio, limiting the method's theoretical rigor. Another related method, Projected Federated Averaging (PFA) [32], retains updates from groups with high privacy budgets while projecting updates from low-budget groups onto the principal subspace learned from the high-budget group. Compared to PFA, our method improves the privacy-utility trade-off through both theoretical analysis and optimization techniques.

3 Federated Learning with Heterogeneous Group Client-Level DP

Problem Formulation. In this work, we consider an HDPFL setting where each client has its own privacy budget $\epsilon_i,\ \forall i \in [n]$. The objective is to collaboratively train a global model with satisfactory utility while respecting each client's privacy preference. To achieve this, our proposed method, GDPFed, partitions all clients into M groups $\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_M$ based on their privacy budgets. Note that the FL problem now is formalized as $\min_\theta \sum_{m\in[M]} \omega_m \sum_{i\in\mathcal{G}_m} f_{m,i}(\theta)$, where $f_{m,i}(\theta)$ is the local learning objective of client i in $\mathcal{G}_m$ and ω_m is a reweighting parameter for each group. In each training round t of GDPFed, the server samples a subset of r_m clients $\mathcal{S}_m^t$ from each group $m \in [M]$ where the number of sampled clients r_m in group m is determined by the client sampling ratio q_m and calculated as $r_m = q_m|\mathcal{G}_m|$. To achieve client-level DP within each group, every local model update in group m is perturbed by adding Gaussian noise drawn from $\mathcal{N}(0, C^2\sigma_m^2/r_m \cdot \mathbf{I}^d)$ after clipping with clipping threshold C. Note that the noise multiplier σ_m^2 is set to satisfy the minimum privacy budget within each group, denoted by $\epsilon_m = \min\{\epsilon_{m,i}\}_{i\in\mathcal{G}_m}$, to ensure that clients' privacy losses are smaller than their budgets. Consequently, selected clients send the perturbed local updates via secure aggregation. One can follow the approach in [9, 25] to implement secure aggregation, and we note that designing a novel secure aggregation protocol is beyond the scope of this paper. The server receives the model update summation from each group and aggregates them with reweighting parameters to refine the global model. This process will repeat for T rounds to ensure that the global model achieves sufficient utility.

Privacy Analysis of GDPFed. We provide per-group privacy guarantees of GDPFed in Theorem 1.

Theorem 1 (Per-Group Privacy Guarantees of GDPFed)**.** *Suppose clients in group m are sampled without replacement with probability q_m at each round. For any $\epsilon_m < 2\log(1/\delta)$ and $\delta \in (0,1)$, GDPFed satisfies (ϵ_m, δ)-DP for clients in group m after T rounds if $\sigma_m^2 \geq 7q_m^2 T(\epsilon_m + 2\log(1/\delta))/\epsilon_m^2$.*

Proof. The detailed proof is provided in Appendix C. □

Remark 1. *This relation helps quantify the required magnitude of DP noise with key parameters to maintain the desired privacy guarantee. Notably, σ_m^2 exhibits a negative correlation with the privacy budget ϵ_m: as ϵ_m increases, the acceptable privacy leakage tolerance grows, thereby reducing the required noise variance. Conversely, σ_m^2 is quadratically and positively correlated with the client*

sampling rate q_m as a higher sampling ratio increases a client's participation frequency, thereby elevating the risk of privacy leakage and necessitating stronger noise injection. The noise level also grows linearly with the number of rounds T, reflecting the cumulative privacy loss over time. In practice, one may choose the exact lower bound value that minimizes the magnitude of DP noise.

In addition to the per-group privacy guarantees provided by GDPFed, we also establish its overall privacy guarantee. To this end, we first present the principle of parallel composition for DP mechanisms, as stated in Lemma 1.

Lemma 1 (Parallel Composition of DP [32, 60])**.** *Let $\mathcal{M}_m : \mathcal{D}_m \to \mathbb{R}^d$ be a randomized mechanism that satisfies (ϵ_m, δ)-DP, where $\{\mathcal{D}_m\}_{m\in[M]}$ are disjoint subsets of the domain $\mathcal{D}$. Then, any randomized function applied to the sequence $\{\mathcal{M}_m\}_{m\in[M]}$ satisfies $(\max_{m\in[M]} \epsilon_m, \delta)$-DP.*

That is, if the input domain is partitioned into disjoint subsets independently of the actual data, and each subset is protected using a DP mechanism, the overall privacy guarantee is determined solely by the *weakest* guarantee (i.e., the highest privacy budget) among the individual mechanisms. Using Lemma 1, we can easily establish the overall privacy guarantee of GDPFed as in Theorem 2.

Theorem 2 (Privacy Guarantee of GDPFed)**.** *If each group $m \in [M]$ in GDPFed selects the noise multiplier σ_m^2 satisfies Theorem 1, then after T training rounds, the GDPFed satisfies $\{(\epsilon_m, \delta)\}_{m\in[M]}$ heterogeneous group-wise DP and $(\max_{m\in[M]} \epsilon_m, \delta)$-DP.*

Remark 2. *It is clear that $\min_{i\in[n]} \epsilon_i \leq \max_{m\in[M]} \min_{i\in\mathcal{G}_m} \epsilon_i \leq \max_{i\in[n]} \epsilon_i$ where the two equalities hold under the homogeneous DP setting. Compared with DP-FedAvg, which guarantees a $(\min_{i\in[n]} \epsilon_i, \delta)$-DP for each client in the system, GDPFed achieves a weaker guarantee. Nevertheless, both approaches ensure that any client's privacy budget is not violated. Importantly, GDPFed relaxes the guarantee for clients with looser privacy requirements, potentially improving the utility of the resulting global model.*

Analyzing DP Noise. Building upon the privacy analysis of GDPFed, we now conduct a detailed investigation of the factors that influence the magnitude of the DP noise applied to the model updates, aiming to derive further insights for improving model utility. In GDPFed, we leverage the Gaussian mechanism to impose noise for each group, drawn from the distribution $\mathcal{N}(0, (C^2\sigma_m^2/r_m) \cdot \mathbf{I}^d)$, thereby ensuring (ϵ_m, δ)-DP. The expected squared ℓ_2-norm of the total noise applied to aggregated model updates (denoted as Λ_m) received by the server is $\Lambda_m = d \cdot C^2\sigma_m^2$, for group m. Substituting σ_m^2 with its lower bound from Theorem 1, we obtain $\Lambda_m = 7dq_m^2T(\epsilon_m + 2\log(1/\delta))C^2/\epsilon_m^2$. We focus on analyzing the influence of two critical parameters, d and q_m, on the magnitude of DP noise, as other parameters are typically fixed in a given HDPFL system. Specifically, properly adjusting d and q_m can effectively reduce the amount of noise under the same privacy guarantee. If model utility is preserved in the process, this can potentially lead to improved overall performance.

a) Reducing d. Modern neural network architectures (e.g., ResNet [21]) are typically designed with millions of parameters to ensure strong generalization capability. This results in a large model dimensionality d, which in turn significantly increases the magnitude of DP noise. To reduce d, existing works consider low-rank decomposition [18, 59, 62] or structured pruning [22, 23]. However, these methods suffer from significant utility loss [25]. Moreover, they alter the model architecture, which poses challenges for model aggregation in FL. A more effective approach is to retain the original architecture while reducing the number of active parameters, a technique known as *model sparsification* [31] (also known as unstructured pruning). This strategy selectively eliminates a subset of model parameters, which directly reduces DP noise while preserving both the original network architecture and model performance, leveraging the natural redundancy present in DNNs.

b) Adjusting q_m. Regarding q_m, directly reducing it leads to a smaller magnitude of DP noise injected into model updates for group m. Intuitively, it is desirable to reduce the sampling probability for groups with tighter privacy requirements (i.e., smaller ϵ_m), as these groups demand lower privacy loss. In practice, privacy-sensitive clients indeed prefer to participate less frequently in training, reducing their exposure to potential inference attacks [33]. However, this approach degrades global model performance if insufficient clients participate in local training. Assuming a minimum participation ratio q is required (i.e., in expectation, qn clients are selected for local training in each round of GDPFed), clients with larger privacy budgets should participate more frequently, as their model updates contain less noise. Yet, excessive participation frequency also increases DP noise under the same privacy guarantee. Consequently, there exists an *optimal set of client sampling ratios* that balances these competing factors while satisfying both participation and privacy constraints.

4 Sparsification-Amplified GDPFed with Optimal Client Sampling

In this section, we further improve the model utility of GDPFed by integrating sparsification techniques and deriving the optimal per-group client sampling ratios. This improved version of GDPFed is referred to as GDPFed$^+$, as detailed in Algorithm 1.

GDPFed with Per-group Sparsification. To achieve client-level DP under our attack model, sampled clients in each group add a small amount of noise to the model updates (line 13 in Algorithm 1) and send them to the server via secure aggregation (line 14). The secure aggregation ensures the server only receives the sum of model updates from each group, as well as the summed noise (line 16). Here, per-group DP perturbation is applied over the entire parameter space (i.e., $\mathbb{R}^d$) of model updates. In other words, all parameters are subjected to perturbation regardless of their importance. However, prior studies have shown that neural networks typically exhibit substantial parameter redundancy, with many parameters contributing negligibly to the task [19, 20, 31]. Under DP settings, perturbing unimportant parameters introduces redundant noise, unnecessarily degrading model utility.

A practical remedy is model sparsification, which removes unimportant parameters from model updates along with their associated noise. Specifically, a top-k sparsifier, denoted as $\mathrm{Top}_k(\cdot)$, is applied to retain only the $k \in [0, d]$ most important parameters. Note that $k = 0$ corresponds to eliminating all parameters, while $k = d$ indicates no sparsification. The detailed algorithm of $\mathrm{Top}_k(\cdot)$ is provided in Algorithm 3 in Appendix B.3. In this work, we adopt a widely-used and straightforward criterion for identifying important parameters—their absolute magnitude [19, 25, 56, 57].

Algorithm 1 GDPFed$^+$: Sparsification-Amplified GDPFed with Optimal Client Sampling

Require: Optimal client sampling ratio $\{q_m\}_{m=1}^M$; training rounds T; local iteration τ; local learning rate η; clipping threshold C; noise multipliers $\{\sigma_m^2\}_{m=1}^M$; reweighting parameters $\{\omega_m\}_{m=1}^M$; top-k parameter $\{k_m\}_{m=1}^M$;
Ensure: Global model θ^T
1: **Initialization:** Randomly initialize $\theta^0 \in \mathbb{R}^d$
2: **for** $t = 0$ to $T-1$ **do**
3: **for** group $m = 1$ to M **do**
4: Sample $r_m = q_m|\mathcal{G}_m|$ clients $\mathcal{S}_m^t$ from $\mathcal{G}_m$
5: Broadcast θ^t to all clients in $\mathcal{S}_m^t$
6: **for** client $i \in \mathcal{S}_m^t$ **in parallel do**
7: **for** $s = 0$ to $\tau-1$ **do**
8: Compute a mini-batch gradient $g_{m,i}^{t,s}$
9: $\theta_{m,i}^{t,s+1} \leftarrow \theta_{m,i}^{t,s} - \eta g_{m,i}^{t,s}$
10: **end for**
11: $\hat{\Delta}_{m,i}^t \leftarrow \theta_{m,i}^{t,\tau} - \theta^t$
12: $\bar{\Delta}_{m,i}^t \leftarrow \hat{\Delta}_{m,i}^t \times \min(1, C/\|\hat{\Delta}_{m,i}^t\|_2)$
13: $\Delta_{m,i}^t \leftarrow \bar{\Delta}_{m,i}^t + \mathcal{N}(0, (C^2\sigma_m^2/r_m) \cdot \mathbf{I}^d)$
14: $\mathbf{y}_{m,i}^t \leftarrow \mathrm{Encrypt}(\Delta_{m,i}^t)$ via secure aggregation and send $\mathbf{y}_{m,i}^t$ to the server
15: **end for**
16: $\bar{\mathbf{y}}_m^t \leftarrow \sum_{i \in \mathcal{S}_m^t} \mathbf{y}_{m,i}^t$
17: $\tilde{\mathbf{y}}_m^t \leftarrow \mathrm{Top}_k(\bar{\mathbf{y}}_m^t, k_m)$
18: **end for**
19: $\theta^{t+1} \leftarrow \theta^t + \sum_{m \in [M]} \omega_m \tilde{\mathbf{y}}_m^t$
20: **end for**
21: **return** θ^T

It should be noted that sparsification must be applied after DP perturbation to preserve the desired (ϵ, δ)-DP guarantee, as ensured by the post-processing property of DP given as in Lemma 2.

Lemma 2 (Post-Processing of DP [14])**.** *Let $\mathcal{M}$ be a randomized mechanism that satisfies (ϵ, δ)-DP. Then, for any (possibly randomized) mapping g, the composed function $g \circ \mathcal{M}$ also satisfies (ϵ, δ)-DP.*

Technically, in training round t, the server applies the $\mathrm{Top}_k(\cdot)$ sparsifier to the per-group model updates summation $\bar{\mathbf{y}}_m^t$ using a group-specific sparsification parameter k_m, resulting in sparsified updates $\tilde{\mathbf{y}}_m^t$ (line 17). These sparsified updates are then aggregated with reweighting parameters to refine the global model (line 19).

Bounded Sparsification Error. To reflect varying privacy preferences, it is desirable to assign distinct sparsification parameters (i.e., $k_1, k_2, \ldots, k_M$) to different groups. Intuitively, groups with stricter privacy requirements should be assigned more aggressive sparsification to mitigate the larger DP noise added to their updates. However, $\mathrm{Top}_k(\cdot)$ is not without cost since using a smaller k_m means that more parameters are removed, which can potentially lead to a non-negligible loss in utility. To formally quantify this relationship, we introduce Lemma 3, which characterizes the approximation error introduced by the $\mathrm{Top}_k(\cdot)$ sparsifier.

Lemma 3 (Bounded Sparsification)**.** *Given a vector $x \in \mathbb{R}^d$ and a sparsification parameter $k \in [d]$. The $\mathrm{Top}_k(\cdot)$ holds that $\mathbb{E}\|\mathrm{Top}_k(x) - x\|^2 \leq \phi\|x\|^2$, where ϕ is a sparsification error coefficient.*

It is evident that a smaller k results in a larger ϕ, thereby leading to a greater sparsification error. Therefore, k_m should be carefully selected in order to successfully leverage its benefit. In the

literature, ϕ is typically set to $1-k/d$ [25, 57] or $(1-k/d)^2$ [49] to measure the sparsification error. In this work, we choose $\phi=(1-k/d)^2$ as it provides a tighter bound.

Convergence Analysis of GDPFed. We now present a detailed convergence analysis of the sparsification-amplified GDPFed. In the following, we present several important assumptions that help us conduct the convergence analysis.

Assumption 1 (L-Smoothness)**.** *The local objective $f_{m,i}(\cdot)$ of each client $i\in\mathcal{G}_m$ in any group $m\in[M]$, is L-smooth with constant $L>0$; i.e., for all $x,y\in\mathbb{R}^d$, $\|\nabla f_{m,i}(x)-\nabla f_{m,i}(y)\|\leq L\|x-y\|$, which implies $f_{m,i}(x)-f_{m,i}(y)\leq\nabla f_{m,i}(x)^\top(y-x)+(L/2)\|x-y\|^2$.*

Assumption 2 (Unbiased Gradient and Bounded Variance)**.** *For each client $i\in\mathcal{G}_m$ in any group $m\in[M]$, the stochastic gradient $g_{m,i}(x)\in\mathbb{R}^d$ satisfies: $\mathbb{E}[g_{m,i}(x)]=\nabla f_{m,i}(x)$ and $\mathbb{E}\|[g_{m,i}(x)]_j-[\nabla f_{m,i}(x)]_j\|^2\leq\zeta_{m,i}^2,\forall j\in[d]$, where the expectation is over mini-batch sampling.*

Assumption 3 (Bounded Dissimilarity)**.** *There exist two constants $\beta^2\geq 1$ and $\kappa^2\geq 0$ such that $\sum_{m=1}^M\omega_m\sum_{i\in\mathcal{G}_m}\|\nabla f_{m,i}(x)\|^2\leq\beta^2\|\sum_{m=1}^M\omega_m\sum_{i\in\mathcal{G}_m}\nabla f_{m,i}(x)\|^2+\kappa^2$. If all local objective functions are identical, the inequality holds with $\beta^2=1$ and $\kappa^2=0$.*

Note that Assumption 1 and Assumption 2 are commonly used in the theoretical analysis of distributed learning systems [25, 44, 57]. In particular, Assumption 2 bounds the coordinate-wise variance of local gradients [24]. Meanwhile, Assumption 3 captures inter-client heterogeneity in FL [4, 15, 28]. With the above assumptions, we provide the convergence result of GDPFed under the general non-convex setting in Theorem 3.

Theorem 3 (Convergence Result of GDPFed)**.** *Let θ^0 be the initial point and f^* be the optimal objective value. Assume the learning rate satisfies $\eta\leq\min\{1/\left(4L\beta^2\left(\tau+1\right)+8L\tau\beta^2\right),1/(16\tau L)\}$, then the sequence of outputs θ^t generated by GDPFed satisfies:*

$$\frac{1}{T}\sum_{t=0}^{T-1}\left\|\nabla f(\theta^t)\right\|^2\leq\frac{8\left(f(\theta^0)-f^*\right)}{\eta T\tau}+\mu_1\kappa^2+\mu_2\sum_{m=1}^M\omega_m(\phi_m+1)d\zeta_m^2+\mu_3\sum_{m=1}^M\frac{k_m\omega_m^2C^2\sigma_m^2}{r_mq_m},$$

where $\mu_1=4L\eta\tau+4L\eta+64L$, $\mu_2=32L\eta\tau+L\eta+L\eta/\tau$, $\mu_3=4L/\eta\tau$, $\phi_m=(1-k_m/d)^2$, and $\zeta_m^2=(1/|\mathcal{G}_m|)\sum_{i\in\mathcal{G}_m}\zeta_{m,i}^2$.

Proof. The detailed proof is given in Appendix D. □

Remark 3. *If $\phi_m=0,\ \forall m\in[M]$, meaning no sparsification is applied, the first three terms on the right-hand side of the convergence bound correspond to the optimization error of FedAvg. In particular, the third term captures group-wise heterogeneity in model updates, which are influenced by the group-wise sparsification parameters k_m. Specifically, applying more aggressive sparsification (i.e., smaller k_m) increases the heterogeneity among per-group model updates. However, as reflected in the final term of the bound, a smaller k_m reduces the privacy error introduced by DP, confirming our analysis in Section 3. This highlights a fundamental trade-off: selecting an appropriate k_m is crucial for balancing sparsification and privacy errors, thereby minimizing the overall convergence error. Hence, by directly minimizing the errors in the third and last terms that are related to k_m, we obtain we obtain a coarse closed-form expression for the optimal sparsification level for the group m: $k_m^*/d=1-2\omega_m\sigma_m^2/(\eta\tau\mu_4r_m^2),\ \forall m\in[M]$, where $\mu_4=32\eta\tau+\eta+\eta/\tau$. At this case, ϕ_m^* is given by $\phi_m^*=4\omega_m^2\sigma_m^4/(\eta\tau\mu_4r_m^2)^2$ (see the sketch of the derivation in Appendix E). This yields a tighter upper bound for the convergence error.*

Optimal Client Sampling Ratios. Building on the convergence analysis of sparsification-amplified GDPFed, we now discuss how to determine the optimal client sampling ratio for each group. To ensure that the global model trained by GDPFed converges to a better optimum, it is desirable to minimize the true gradient of the objective function (i.e., the left-hand side of the convergence result). However, directly minimizing this function is typically infeasible in practice, as $\nabla f(\theta^t)$ is a high-dimensional, non-convex function. An alternative approach is optimizing its upper bound (i.e., the right-hand side of the convergence bound), which approximates optimizing the objective function. Notably, only the third and last terms in the bound are influenced by the client sampling ratios. This leads to the constrained minimization problem formulated in Problem 1.

Problem 1 (Optimal Sampling Ratios for GDPFed)**.** *The optimal per-group sampling ratios* $\{q_m\}_{m\in[M]}$ *for GDPFed are obtained by solving the following constrained optimization problem:*

$$\min_{\{q_m\}_{m\in[M]}} \quad \sum_{m\in[M]} \omega_m \left(\mu_4(1+\phi_m^*) + \mu_5 \frac{\left(1-\sqrt{\phi_m^*}\right)\omega_m \sigma_m^2}{r_m^2} \right)$$
$$s.t. \quad r_m = q_m|\mathcal{G}_m|, \ \sum_{m\in[M]} r_m = qn,$$

where $\mu_4 = 32\eta\tau + \eta + \eta/\tau$ *and* $\mu_5 = 4/(\eta\tau)$*. The formulation sketch is provided in Appendix F.*

Remark 4. *The optimal sparsification error coefficient* ϕ_m^* *is defined as given in Remark 3. If there is no sparsification applied, then* $\phi_m^* = 0$, $\forall m \in [M]$*. The noise multiplier* σ_m^2 *required to satisfy the* (ϵ_m, δ)*-DP for group* m *is derived in Theorem 1. All parameters in Problem 1 are now the settings of the system, except for the decision variables. Therefore, the optimal client sampling ratios for each group in GDPFed can be efficiently obtained by solving this minimization problem. As Problem 1 is a non-convex optimization problem, one can resort to existing solvers in practice, such as optimization libraries in Python (e.g., scikit-learn [46]), to obtain a feasible solution.*

With the optimal client sampling ratios derived from solving Problem 1, which minimizes the convergence upper bound in Theorem 3, GDPFed$^+$ converges to a better minimum than GDPFed, thereby enhancing model utility. Importantly, GDPFed$^+$ still satisfies the per-group privacy guarantees in Theorem 1, the overall privacy guarantee in Theorem 2, and the convergence bound in Theorem 3.

5 Empirical Evaluation

Datasets and Settings. Our evaluation covers four benchmark datasets for DPFL: Fashion MNIST (FMNIST) [55], SVHN [45], CIFAR-10 [30], and Shakespeare [10]. Correspondingly, we adopt a 2-layer CNN for FMNIST, a 3-layer CNN for SVHN, a ResNet-18 [21] for CIFAR-10, and an LSTM model for Shakespeare. We conduct experiments in *cross-device* FL settings with n clients. Datasets, excluding Shakespeare, are evenly partitioned (i.e., IID) across clients; Shakespeare is used in its natural non-IID form. Following prior works [3, 7, 29] that simulate heterogeneous privacy requirements, clients are assigned to one of three groups, each associated with a distinct minimum privacy budget $(\epsilon_1, \epsilon_2, \epsilon_3)$. By default, clients are evenly distributed among three groups. For GDPFed$^+$, the optimal client sampling ratios (q_1, q_2, q_3) derived by solving Problem 1. By default, the sparsification levels k_m/d for each group are $(0.7, 0.8, 0.9)$. We provide detailed empirical observations on the effects of varying sparsification levels and offer practical suggestions for selecting appropriate sparsification configurations in Appendix B.4. We summarize the system configurations in Table 1. Detailed experimental settings are provided in Appendix B.2. *All experiments are repeated* 3 *times with different seeds.*

Table 1: Configurations for each dataset.

Dataset	n	T	C	$(\epsilon_1, \epsilon_2, \epsilon_3)$	(q_1, q_2, q_3)-q (%)
FMNIST	6,000	50	1.5	(0.5, 1.5, 3.0)	(0.69, 1.89, 3.42)-2
SVHN	6,000	100	1.0	(0.5, 1.5, 3.0)	(1.66, 4.67, 8.68)-5
Shakespeare	714	50	1.0	(0.5, 1.5, 3.0)	(4.79, 9.83, 15.21)-10
CIFAR-10	600	100	1.5	(2.0, 6.0, 12.0)	(3.61, 9.62, 16.77)-10

Baselines. We compare against four baselines to demonstrate the effectiveness of GDPFed$^+$. Specifically, we include two important baselines: Pure FedAvg (P-FedAvg), a non-private FL case that represents the upper bound of model utility, and client-level DP-FedAvg (DP-FedAvg), which enforces the strictest privacy requirement across all clients. Moreover, our comparisons include IDP-FedAvg [7] and PFA [32] (see the rationale behind selecting these methods in Appendix B.2).

Table 2: Test accuracy of baselines, GDPFed, and GDPFed$^+$ on each dataset. Results are shown in percentages (%).

Method	FMNIST	SVHN	CIFAR-10	Shakespeare	Avg.
P-FedAvg	$78.96_{\pm 0.90}$	$84.13_{\pm 0.18}$	$56.40_{\pm 0.34}$	$60.62_{\pm 0.76}$	70.03
DP-FedAvg	$71.88_{\pm 0.15}$	$40.80_{\pm 1.73}$	$32.10_{\pm 0.34}$	$34.97_{\pm 0.76}$	44.94
GDPFed	$73.97_{\pm 0.21}$	$59.11_{\pm 1.72}$	$34.00_{\pm 0.44}$	$37.63_{\pm 0.29}$	51.18
PFA	$73.67_{\pm 0.28}$	$67.51_{\pm 0.84}$	$37.17_{\pm 0.98}$	$36.69_{\pm 0.21}$	53.76
IDP-FedAvg	$74.80_{\pm 0.19}$	$66.46_{\pm 0.98}$	$37.49_{\pm 0.761}$	$37.26_{\pm 0.55}$	54.00
GDPFed$^+$	$\mathbf{75.83_{\pm 0.47}}$	$\mathbf{71.10_{\pm 0.58}}$	$\mathbf{38.78_{\pm 0.36}}$	$\mathbf{42.00_{\pm 0.45}}$	**56.93**

Experimental Results. We begin by presenting the convergence curves of representative methods in Figure 1 and the corresponding test accuracies in Table 2. As shown in Figure 1, DP-FedAvg suffers

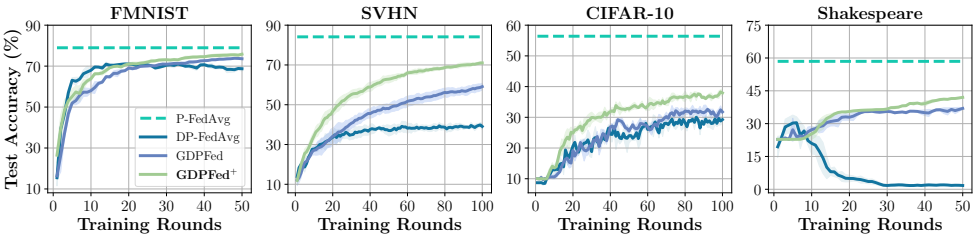

Figure 1: Convergence comparison between P-FedAvg, DP-FedAvg, GDPFed, and GDPFed$^+$.

from substantial performance degradation compared to P-FedAvg, due to applying the strictest privacy level uniformly across all clients. In contrast, GDPFed, which enforces group-level DP guarantees, consistently converges to better optima across all datasets by reducing the noise magnitude for clients with looser privacy requirements. Building upon this, GDPFed$^+$ further integrates model sparsification with optimal per-group client sampling ratios, resulting in enhanced performance and achieving the highest test accuracies on all datasets. Moreover, GDPFed$^+$ exhibits a more stable convergence process compared to other baselines. As shown in Table 2, GDPFed$^+$ outperforms two competitive baselines, PFA and IDP-FedAvg, with average accuracy gains of $+3.17\%$ and $+2.93\%$, respectively. These results demonstrate the empirical effectiveness of GDPFed$^+$ in enhancing model utility while preserving client-level HDP guarantees.

Noise Analysis. Then, we analyze why our proposed methods achieve better performance. Specifically, we compute the total amount of noise (measured as the expectation of the squared ℓ_2-norm and denoted by Λ) added to the global model updates. In Table 3, we report the noise multipliers and corresponding Λ values for DP-FedAvg, GDPFed, GDPFed-op (GDPFed with only optimized client sampling ratios), and GDPFed$^+$ on the FMNIST dataset as an example. Comprehensive results are provided in Table 5 in Appendix B.4. As shown in the table, GDPFed reduces the total noise by nearly half compared to DP-FedAvg, as it relaxes the privacy constraints for clients with looser requirements. GDPFed-op further significantly decreases Λ by adjusting the noise multipliers based on optimized client sampling ratios. Finally, GDPFed$^+$ achieves the smallest Λ by *additionally* applying model sparsification to eliminate noise associated with less informative model parameters. These results highlight the effectiveness of our design in reducing DP noise to improve the privacy-utility trade-off.

Table 3: Detailed noise multipliers and Λ for different methods.

Method	σ^2 / $(\sigma_1^2, \sigma_2^2, \sigma_3^2)$	Λ
DP-FedAvg	2.26	5.09
GDPFed	$(2.26, 0.90, 0.53)$	2.77
GDPFed-op	$(1.42, 0.87, 0.70)$	0.57
GDPFed$^+$	$\mathbf{(1.42, 0.87, 0.70)}$	**0.49**

Additional Results and Discussions. We present more results in Appendix B.4. Specifically, we illustrate how the optimal client sampling ratios vary with different privacy budget settings (Figure 2). We also report the performance of our methods under various degrees of non-IIDness (Table 7). Moreover, we examine the impact of different distributions of client privacy preferences (Table 8). In addition, we analyze how the key DP-related parameters such as ϵ (Table 9) and C (Table 10) affect model performance. Finally, we discuss the broader impact of our work in Appendix B.5.

6 Conclusion and Future Work

In this work, we explore the challenges of achieving client-level DP in heterogeneous privacy settings. Unlike classic methods that must satisfy the strictest privacy requirements across all clients, we propose GDPFed, which partitions clients into groups to ensure group-level DP guarantees. To further enhance the utility of GDPFed, we introduce GDPFed$^+$, which integrates model sparsification and optimal client sampling ratios. GDPFed$^+$ preserves the same privacy guarantees as GDPFed while achieving significant utility improvements, as demonstrated both theoretically and empirically.

We discuss some promising directions for future research. First, it would be desirable to incorporate the degree of data heterogeneity within each group into Problem 1. While we empirically demonstrate that GDPFed and GDPFed$^+$ perform well under non-IID data, a biased sampling of clients that over- or under-represents certain privacy groups could potentially exacerbate model bias in non-IID settings. Second, although we derive the theoretically optimal sparsification levels in Remark 3, using this closed-form solution in practice can be challenging due to its dependence on unknown parameters and Lemma 3 does not precisely capture the sparsification error of $\mathrm{Top}_k(\cdot)$. Co-designing the optimal sparsification levels and client sampling ratios is an important direction for future work.

References

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016. 3

[2] Naman Agarwal, Ananda Theertha Suresh, Felix Xinnan X Yu, Sanjiv Kumar, and Brendan McMahan. cpsgd: Communication-efficient and differentially-private distributed sgd. *Advances in Neural Information Processing Systems*, 31, 2018. 3

[3] Mohammad Alaggan, Sébastien Gambs, and Anne-Marie Kermarrec. Heterogeneous differential privacy. *arXiv preprint arXiv:1504.06998*, 2015. 8

[4] Youssef Allouah, Sadegh Farhadkhani, Rachid Guerraoui, Nirupam Gupta, Rafaël Pinot, and John Stephan. Fixing by mixing: A recipe for optimal byzantine ml under heterogeneity. In *International Conference on Artificial Intelligence and Statistics*, pages 1232–1300. PMLR, 2023. 7

[5] Galen Andrew, Om Thakkar, Brendan McMahan, and Swaroop Ramaswamy. Differentially private learning with adaptive clipping. *Advances in Neural Information Processing Systems*, 34:17455–17466, 2021. 2, 3

[6] Amos Beimel, Hai Brenner, Shiva Prasad Kasiviswanathan, and Kobbi Nissim. Bounds on the sample complexity for private learning and private data release. *Machine learning*, 94:401–437, 2014. 3

[7] Franziska Boenisch, Christopher Mühl, Adam Dziedzic, Roy Rinberg, and Nicolas Papernot. Have it your way: Individualized privacy assignment for dp-sgd. *Advances in Neural Information Processing Systems*, 36:19073–19103, 2023. 2, 4, 8, 15

[8] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191, 2017. 2, 3

[9] Keith Bonawitz, Fariborz Salehi, Jakub Konečnỳ, Brendan McMahan, and Marco Gruteser. Federated learning with autotuned communication-efficient secure aggregation. In *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, pages 1222–1226. IEEE, 2019. 4

[10] Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečnỳ, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018. 8

[11] Ajesh Koyatan Chathoth, Abhyuday Jagannatha, and Stephen Lee. Federated intrusion detection for iot with heterogeneous cohort privacy. *arXiv preprint arXiv:2101.09878*, 2021. 2, 3, 4

[12] Wenlin Chen, Samuel Horvath, and Peter Richtarik. Optimal client sampling for federated learning. *arXiv preprint arXiv:2010.13723*, 2020. 3

[13] Anda Cheng, Peisong Wang, Xi Sheryl Zhang, and Jian Cheng. Differentially private federated learning with local regularization and sparsification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10122–10131, 2022. 3, 4

[14] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014. 1, 2, 3, 6, 14

[15] El Mahdi El-Mhamdi, Sadegh Farhadkhani, Rachid Guerraoui, Arsany Guirguis, Lê-Nguyên Hoang, and Sébastien Rouault. Collaborative learning in the jungle (decentralized, byzantine, heterogeneous, asynchronous and nonconvex learning). *Advances in neural information processing systems*, 34:25044–25057, 2021. 7

[16] Hossein Fereidooni, Samuel Marchal, Markus Miettinen, Azalia Mirhoseini, Helen Möllering, Thien Duc Nguyen, Phillip Rieger, Ahmad-Reza Sadeghi, Thomas Schneider, Hossein Yalame, et al. Safelearn: Secure aggregation for private federated learning. In *2021 IEEE Security and Privacy Workshops (SPW)*, pages 56–62. IEEE, 2021. 2, 3

[17] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1322–1333, 2015. 1

[18] Yuanxiong Guo, Rui Hu, and Yanmin Gong. Agent-level differentially private federated learning via compressed model perturbation. In *2022 IEEE Conference on Communications and Network Security (CNS)*, pages 127–135. IEEE, 2022. 5

[19] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015. 6

[20] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015. 6

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5, 8

[22] Yang He and Lingao Xiao. Structured pruning for deep convolutional neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 46(5):2900–2919, 2023. 5

[23] Yang He, Ping Liu, Ziwei Wang, Zhilan Hu, and Yi Yang. Filter pruning via geometric median for deep convolutional neural networks acceleration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4340–4349, 2019. 5

[24] Rui Hu, Yanmin Gong, and Yuanxiong Guo. Federated learning with sparsification-amplified privacy and adaptive optimization. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, 2021. 7

[25] Rui Hu, Yuanxiong Guo, and Yanmin Gong. Federated learning with sparsified model perturbation: Improving accuracy under client-level differential privacy. *IEEE Transactions on Mobile Computing*, 2023. 2, 3, 4, 5, 6, 7, 16, 21

[26] Zach Jorgensen, Ting Yu, and Graham Cormode. Conservative or liberal? personalized differential privacy. In *2015 IEEE 31St international conference on data engineering*, pages 1023–1034. IEEE, 2015. 2, 4

[27] Swanand Kadhe, Nived Rajaraman, O Ozan Koyluoglu, and Kannan Ramchandran. Fastsecagg: Scalable secure aggregation for privacy-preserving federated learning. *arXiv preprint arXiv:2009.11248*, 2020. 2, 3

[28] Sai Praneeth Karimireddy, Lie He, and Martin Jaggi. Byzantine-robust learning on heterogeneous datasets via bucketing. *arXiv preprint arXiv:2006.09365*, 2020. 7

[29] Shahrzad Kiani, Nupur Kulkarni, Adam Dziedzic, Stark Draper, and Franziska Boenisch. Differentially private federated learning with time-adaptive privacy spending. In *The Thirteenth International Conference on Learning Representations*, 2025. 2, 3, 4, 8, 15

[30] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. *University of Toronto*, 2009. 8

[31] Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989. 5, 6

[32] Junxu Liu, Jian Lou, Li Xiong, Jinfei Liu, and Xiaofeng Meng. Projected federated averaging with heterogeneous differential privacy. *Proceedings of the VLDB Endowment*, 15(4):828–840, 2021. 2, 3, 4, 5, 8, 15

[33] Junxu Liu, Jian Lou, Li Xiong, Jinfei Liu, and Xiaofeng Meng. Cross-silo federated learning with record-level personalized differential privacy. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, page 303–317, New York, NY, USA, 2024. Association for Computing Machinery. 2, 3, 4, 5

[34] Ken Liu, Shengyuan Hu, Steven Z Wu, and Virginia Smith. On privacy and personalization in cross-silo federated learning. *Advances in neural information processing systems*, 35:5925–5940, 2022. 3

[35] Jiating Ma, Yipeng Zhou, Qi Li, Quan Z Sheng, Laizhong Cui, and Jiangchuan Liu. The power of bias: Optimizing client selection in federated learning with heterogeneous differential privacy. *arXiv preprint arXiv:2408.08642*, 2024. 2, 4

[36] Saber Malekmohammadi, Yaoliang Yu, and Yang Cao. Noise-aware algorithm for heterogeneous differentially private federated learning. In *Proceedings of the 41st International Conference on Machine Learning*, pages 34461–34498. PMLR, 2024. 2

[37] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017. 1, 3, 16

[38] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *International Conference on Learning Representations*, 2018. 1, 2, 3, 4, 16

[39] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting unintended feature leakage in collaborative learning. In *2019 IEEE symposium on security and privacy (SP)*, pages 691–706. IEEE, 2019. 1

[40] Thomas Minka. Estimating a dirichlet distribution, 2000. 18

[41] Ilya Mironov. Rényi differential privacy. In *2017 IEEE 30th computer security foundations symposium (CSF)*, pages 263–275. IEEE, 2017. 3, 14

[42] Ilya Mironov, Kunal Talwar, and Li Zhang. Rényi differential privacy of the sampled gaussian mechanism. *arXiv preprint arXiv:1908.10530*, 2019. 3

[43] Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *2019 IEEE symposium on security and privacy (SP)*, pages 739–753. IEEE, 2019. 1

[44] Yurii Nesterov et al. *Lectures on convex optimization*. Springer, 2018. 7

[45] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, page 4. Granada, 2011. 8

[46] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. 8

[47] Monica Ribero and Haris Vikalo. Communication-efficient federated learning via optimal client sampling. *arXiv preprint arXiv:2007.15197*, 2020. 3

[48] Xiaoying Shen, Hang Jiang, Yange Chen, Baocang Wang, and Le Gao. Pldp-fl: Federated learning with personalized local differential privacy. *Entropy*, 25(3):485, 2023. 2, 4

[49] Shaohuai Shi, Xiaowen Chu, Ka Chun Cheung, and Simon See. Understanding top-k sparsification in distributed deep learning. *arXiv preprint arXiv:1911.08772*, 2019. 7

[50] Yifan Shi, Yingqi Liu, Kang Wei, Li Shen, Xueqian Wang, and Dacheng Tao. Make landscape flatter in differentially private federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24552–24562, 2023. 2, 3

[51] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017. 1

[52] Lin Wang, YongXin Guo, Tao Lin, and Xiaoying Tang. DELTA: Diverse client sampling for fasting federated learning. *Advances in Neural Information Processing Systems*, 36, 2024. 3

[53] Yu-Xiang Wang, Borja Balle, and Shiva Prasad Kasiviswanathan. Subsampled rényi differential privacy and analytical moments accountant. In *The 22nd international conference on artificial intelligence and statistics*, pages 1226–1235. PMLR, 2019. 14

[54] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H Vincent Poor. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE transactions on information forensics and security*, 15:3454–3469, 2020. 3

[55] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017. 8

[56] Jiahao Xu, Zikai Zhang, and Rui Hu. Detecting backdoor attacks in federated learning via direction alignment inspection. *arXiv preprint arXiv:2503.07978*, 2025. 6

[57] Jiahao Xu, Zikai Zhang, and Rui Hu. Achieving byzantine-resilient federated learning via layer-adaptive sparsified model aggregation. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 1508–1517, 2025. 6, 7

[58] Ge Yang, Shaowei Wang, and Haijie Wang. Federated learning with personalized local differential privacy. In *2021 IEEE 6th International Conference on Computer and Communication Systems (ICCCS)*, pages 484–489. IEEE, 2021. 2, 4

[59] Da Yu, Huishuai Zhang, Wei Chen, and Tie-Yan Liu. Do not let privacy overbill utility: Gradient embedding perturbation for private learning. In *International Conference on Learning Representations*, 2021. 5

[60] Lei Yu, Ling Liu, Calton Pu, Mehmet Emre Gursoy, and Stacey Truex. Differentially private model publishing for deep learning. In *2019 IEEE symposium on security and privacy (SP)*, pages 332–349. IEEE, 2019. 5

[61] Lin Zhang, Li Shen, Liang Ding, Dacheng Tao, and Ling-Yu Duan. Fine-tuning global model via data-free knowledge distillation for non-iid federated learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10174–10183, 2022. 1

[62] Yingxue Zhou, Steven Wu, and Arindam Banerjee. Bypassing the ambient dimension: Private {sgd} with gradient subspace identification. In *International Conference on Learning Representations*, 2021. 5

A Additional Background

In this section, we provide additional background which are related to our paper. First, we introduce the formal definition of RDP as in Definition 2.

Definition 2 ((α, ρ)-RDP [41]). *Given controlling parameter $\alpha > 1$ and privacy parameter $\rho \geq 0$, a randomized mechanism $\mathcal{M}$ satisfies (α, ρ)-RDP if for any two adjacent datasets D, D', the Rényi α-divergence between $\mathcal{M}(D)$ and $\mathcal{M}(D')$ satisfies $D_\alpha[\mathcal{M}(D)\|\mathcal{M}(D')] := (1/(\alpha-1)) \log \mathbb{E}\left[(\mathcal{M}(D)/\mathcal{M}(D'))^\alpha\right] \leq \rho(\alpha)$.*

Useful Definitions and Lemmas. We present several key definitions and lemmas related to DP and RDP, which will be helpful in deriving our main results later in the paper. We begin by introducing the transition from RDP to (ϵ, δ)-DP for any $\delta > 0$ can be achieved using the following Lemma 4.

Lemma 4 (Converting RDP to (ϵ, δ)-DP [41]). *If the randomized mechanism $\mathcal{M}$ satisfies $(\alpha, \rho(\alpha))$-RDP, then it also satisfies $(\rho(\alpha) + \log(1/\delta)/(\alpha-1), \delta)$-DP.*

Next, we present the composability property of RDP as in Lemma 5.

Lemma 5 (RDP Composition [41]). *For two randomized mechanisms $\mathcal{M}_1$ satisfies (α, ρ_1)-RDP and $\mathcal{M}_2$ satisfies (α, ρ_2)-RDP, their composition $\mathcal{M}_1 \circ \mathcal{M}_2$ satisfies $(\alpha, \rho_1 + \rho_2)$-RDP.*

Then, we introduce the concept of ℓ_2-sensitivity (as defined in Definition 3), which quantifies the maximum impact that an individual's data can have on a query function $h(\cdot)$ in the worst-case scenario.

Definition 3 (ℓ_2-Sensitivity [14]). *Let $h : \mathcal{D} \to \mathbb{R}^d$ be a query function over a dataset. The ℓ_2-sensitivity of h is defined as $\psi(h) := \sup_{D, D' \in \mathcal{D}} \|h(D) - h(D')\|_2$ where D and D' are two adjacent datasets.*

With ℓ_2-sensitivity, we introduce the Gaussian Mechanism (as in Lemma 6) which is used for DP mechanisms in our work.

Lemma 6 (Gaussian Mechanism [41]). *Let $h : \mathcal{D} \to \mathbb{R}^d$ be a query function with ℓ_2-sensitivity $\psi(h)$. The Gaussian mechanism $\mathcal{M} = h(D) + \mathcal{N}(0, \sigma^2 \psi(h)^2 \cdot \mathbf{I}^d)$ satisfies $(\alpha, \alpha/2\sigma^2)$-RDP.*

Finally, we discuss how RDP behaves when combined with a subsampling mechanism, as described in Lemma 7.

Lemma 7 (RDP for Subsampling Mechanism [53]). *For a Gaussian mechanism $\mathcal{M}$ and any m-datapoints dataset D, define $\mathcal{M} \circ$ SUBSAMPLE as 1) subsample without replacement B datapoints from the dataset (denote $q = B/m$ as the sampling ratio); and 2) apply $\mathcal{M}$ on the subsampled dataset as input. Then if $\mathcal{M}$ satisfies $(\alpha, \rho(\alpha))$-RDP with respect to the subsampled dataset for all integers $\alpha \geq 2$, then the new randomized mechanism $\mathcal{M} \circ$ SUBSAMPLE satisfies $(\alpha, \rho'(\alpha))$-RDP with respect to D, where*

$$\rho'(\alpha) \leq \frac{1}{\alpha - 1} \log \left(1 + q^2 \binom{\alpha}{2} \min\{4(e^{\rho(2)} - 1), 2e^{\rho(2)}\} + \sum_{j=3}^{\alpha} q^j \binom{\alpha}{j} 2e^{(j-1)\rho(j)}\right).$$

If $\sigma^2 \geq 0.7$ and $\alpha \leq (2/3)\sigma^2 \psi^2(h) \log(1/q\alpha(1+\sigma^2)) + 1$, $\mathcal{M} \circ$ SUBSAMPLE satisfies $(\alpha, 3.5q^2\alpha/\sigma^2)$-RDP.

B Additional Contents

B.1 Notation Table

Table 4: Notation Table

Symbol	Description
n	Total number of clients in the FL system
q/q_m	Global client sampling ratio / client sampling ratios for m
r/r_m	Total selected clients for local training / total selected clients for local training for group m
$\mathcal{S}^t/\mathcal{S}_m^t$	The set of selected clients at round t / the set of selected clients at round t for group m
D_i	Local dataset of client i
$l(\cdot)$	Loss function used for training
d	Model dimensionality
θ^t	Global model parameters at round t
$\theta_i^t/\theta_{m,i}^t$	Local model of client i at round t / local model of client i from group m at round t
$\Delta_i^t/\Delta_{m,i}^t$	Local model update of client i at round t / local model update of client i from group m at round t
T	Total number of training rounds
τ	Local iterations
η	Local learning rate
ϵ/ϵ_m	Privacy budget / group-level privacy budget for group m
δ	Failure parameter
C	Clipping threshold
σ^2/σ_m^2	Noise multiplier / noise multiplier for group m
ω_m	Reweighting parameter for group m
M	Toal number of groups
$\mathcal{G}_m$	Client group m
Λ/Λ_m	The expected squared ℓ_2-norm of noise / the expected squared ℓ_2-norm of noise for group m
k_m	Sparsification parameter for group m
ϕ_m	Sparsification error parameter for group m
ϕ_m^*	Optimal sparsification error parameter for group m as given in Remark 3
L	Smoothness parameter as given in Assumption 1
$\zeta_{m,i}^2$	Coordinate-wise variance as given in Assumption 2
β^2 and κ^2	Dissimilarity parameters as given in Assumption 3

B.2 Additional Experimental Settings

Hardware Settings. All experiments were conducted on a Linux-based internal compute cluster equipped with 8 NVIDIA RTX A6000 GPUs (each with 49 GB of memory) and AMD EPYC 7763 64-core CPUs. The system runs Ubuntu 20.04.6 LTS. Model training was primarily GPU-accelerated. The cluster is self-hosted and not based on any commercial cloud provider. Overall, the experiments consumed approximately one week of cumulative GPU time.

Detailed Baseline Settings. We identify three works that are closely related to our study: IDP-FedAvg [7], PFA [32], and Time Adaptive HDPFL [29]. Among these, we include IDP-FedAvg and PFA in our comparisons. To ensure a fair comparison, we re-implement both methods on top of our GDPFed framework.

- For IDP-FedAvg, originally designed for record-level HDPFL, we adapt it to the client-level HDPFL setting by following the methodology in [29]. Notably, while [29] only implements the Scale variant of IDP-FedAvg, we implement both the Scale and Sample variants, as described in the original IDP-FedAvg paper. We also follow the original codebase to calculate the per-group clipping thresholds and client sampling ratios. Specifically, for FMNIST, SVHN, CIFAR-10, and Shakespeare, the per-group clipping thresholds used are $(1.15, 1.51, 2.12)$, $(0.35, 0.53, 0.81)$, $(0.72, 1.03, 1.55)$, and $(0.73, 1.02, 1.52)$, respectively. The corresponding client sampling ratios $(\%)$ are $(0.82, 2.80, 2.38)$, $(4.07, 4.93, 6.00)$, $(6.09, 11.11, 12.80)$, and $(10.51, 7.49, 12.00)$.
- For PFA, the original implementation considers only two groups. Since our setting involves three privacy groups, we adapt it accordingly: we treat the group with the highest privacy budget as the public group (as in PFA), the group with the lowest privacy budget as the private group (whose

updates are projected onto the public subspace), and retain the median privacy group as-is for direct aggregation.

- Time-Adaptive HDPFL considers a setting in which clients dynamically control their privacy budget expenditure across training rounds. Specifically, clients are constrained to spend less privacy budget in earlier rounds and more in later rounds. The authors also formulate an optimization problem to determine per-group client sampling ratios that minimize the error between the perturbed global model update and an unbiased estimate of the true global model update. However, they do not explicitly solve this optimization problem. Instead, they manually tune the per-group sampling ratios based on empirical observations, which limits the theoretical rigor of their approach. Furthermore, as their experiments are conducted under a different system configuration from ours, we cannot directly adapt their fine-tuned per-group client sampling ratios. Therefore, we are unable to include Time-Adaptive HDPFL in our evaluation.

More Training Settings. In all experiments, local clients use stochastic gradient descent (SGD) as the optimizer with a learning rate of $\eta = 0.1$ and a decay ratio of 0.99. For the FMNIST, SVHN, CIFAR-10, and Shakespeare datasets, the momentum values are set to 0.0, 0.0, 0.5, and 0.9, the local training iterations τ are 5, 25, 5, and 30, and the batch sizes are 10, 10, 50, and 4, respectively. We set the uniform DP failure parameter as $\delta = 1/n^{1.1}$, following the recommendation in [25, 38]. To reweight the per-group model updates, we set the reweighting parameter $\omega_m = (1/qn) \cdot r_m^2 / \sum_{m \in [M]} r_m^2$ for all $m \in [M]$. This reweighting strategy prioritizes groups with higher expected client participation and helps reduce the total noise added to the aggregated model. Note that to ensure a fair comparison, we adopt this reweighting parameter for all group-based methods, including PFA, IDP-FedAvg, GDPFed, and GDPFed$^+$. For additional implementation details, please refer to our released code.

B.3 Details of Algorithms

Algorithm 2 DP-FedAvg

Require: Model dimension d, client sampling ratio q, number of training rounds T, local iteration τ, local learning rate η, clipping threshold C, noise multiplier σ^2
Ensure: Global model θ^T

Initialization: Randomly initialize $\theta^0 \in \mathbb{R}^d$
for $t = 0$ to $T-1$ **do**
  Sample $r = qn$ clients $\mathcal{S}^t$ at random without replacement
  Broadcast θ^t to all clients in $\mathcal{S}^t$
  for each client $i \in \mathcal{S}^t$ **in parallel do**
    for $s = 0$ to $\tau-1$ **do**
      Compute mini-batch gradient $g_i^{t,s}$
      $\theta_i^{t,s+1} \leftarrow \theta_i^{t,s} - \eta \mathbf{g}_i^{t,s}$
    end for
    $\hat{\Delta}_i^t \leftarrow \theta_i^{t,\tau} - \theta^t$
    $\bar{\Delta}_i^t \leftarrow \hat{\Delta}_i^t \times \min(1, C/\|\hat{\Delta}_i^t\|_2)$
    $\Delta_i^t \leftarrow \bar{\Delta}_i^t + \mathcal{N}(0, (C^2\sigma^2/r) \cdot \mathbf{I}^d)$
    $\mathbf{y}_i^t \leftarrow \mathrm{Encrypt}(\Delta_i^t)$ via secure aggregation and send $\mathbf{y}_i^t$ to the server
  end for
  $\bar{\mathbf{y}}^t \leftarrow (1/r) \sum_{i \in \mathcal{S}^t} \mathbf{y}_i^t$
  $\theta^{t+1} \leftarrow \theta^t + \bar{\mathbf{y}}^t$
end for
return θ^T

Algorithm 3 Top-k Sparsifier $\mathrm{Top}_k(\cdot)$

Require: Vector $x \in \mathbb{R}^d$, top-k parameter $k \in [1, d]$
Ensure: Binary mask $\mathbf{mk} \in \{0,1\}^d$

Initialization: Initialize $\mathbf{mk} \leftarrow \mathbf{0} \in \{0,1\}^d$
Compute absolute values: $a_j \leftarrow |x_j|$ for all $j \in [1, d]$
Sort indices π such that $a_{\pi(1)} \geq a_{\pi(2)} \geq \cdots \geq a_{\pi(d)}$
for $j = 1$ **to** k **do**
  $[\mathbf{mk}]_{\pi(j)} \leftarrow 1$
end for
return $\mathbf{mk} \odot x$

We present the classical DP-FedAvg algorithm [38] in Algorithm 2. DP-FedAvg is a differentially private variant of the standard FedAvg algorithm [37], designed to incorporate DP guarantees into the FL process. In our work, we adopt secure aggregation as the default setting for transmitting clients' model updates to the server, ensuring that individual updates remain confidential.

We also provide the detailed $\mathrm{Top}_k(\cdot)$ procedure in Algorithm 3. Specifically, given an input vector, $\mathrm{Top}_k(\cdot)$ first sorts all elements by their absolute values and retains only the top-k elements. The remaining entries are set to zero, resulting in a sparsified output vector.

Table 5: Noise multipliers (σ^2 or $(\sigma_1^2, \sigma_2^2, \sigma_3^2)$) and noise magnitude ($\Lambda$) across different datasets and methods.

Method	FMNIST	SVHN	CIFAR-10	Shakespeare
DP-FedAvg	2.26 / 5.09	13.20 / 13.20	3.52 / 7.91	17.14 / 17.14
GDPFed	$(2.26, 0.90, 0.53)$ / 2.77	$(13.20, 2.50, 1.16)$ / 5.62	$(3.52, 0.95, 0.49)$ / 3.72	$(17.14, 3.26, 1.41)$ / 7.27
GDPFed-op	$(1.42, 0.87, 0.70)$ / 0.57	$(2.38, 2.29, 2.23)$ / 0.75	$(0.98, 0.91, 0.83)$ / 0.64	$(4.45, 3.18, 2.46)$ / 0.93
GDPFed$^+$	$\mathbf{(1.42, 0.87, 0.70)}$ / **0.49**	$\mathbf{(2.38, 2.29, 2.23)}$ / **0.65**	$\mathbf{(0.98, 0.91, 0.83)}$ / **0.56**	$\mathbf{(4.45, 3.18, 2.46)}$ / **0.79**

B.4 Additional Empirical Results

Dynamics of Client Sampling Ratios. We demonstrate how the optimal client sampling ratios dynamically adjust in response to varying privacy budgets, as dictated by our optimization formulation in Problem 1. Specifically, we conduct test on the FMNIST and CIFAR-10 datasets. In both cases, we fix the privacy budgets for Group 2 and Group 3 as $\epsilon_2 = 1.5$ and $\epsilon_3 = 3.0$ for FMNIST and $\epsilon_2 = 6.0$ and $\epsilon_3 = 12.0$ for CIFAR-10, respectively, and vary ϵ_1 to examine how the optimal sampling ratios evolve. The resulting trends are illustrated in Figure 2.

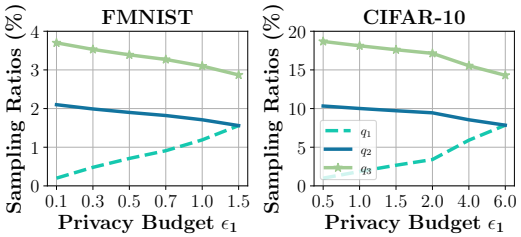

Figure 2: Optimal client sampling ratios under varying ϵ_1. Ratios adjust dynamically to satisfy the global constraint with fixed ϵ_2 and ϵ_3.

Across both datasets, we observe a consistent pattern governed by the optimization objective. When ϵ_1 is small, the corresponding sampling ratio q_1 decreases to accommodate the stronger noise required for stricter privacy. As ϵ_1 increases, q_1 rises accordingly, while q_2 and q_3 adjust downward to maintain the global constraint $\sum_{m\in[M]} r_m = qn$. Notably, when $\epsilon_1 = \epsilon_2 = 1.5$ for FMNIST and $\epsilon_1 = \epsilon_2 = 6.0$ for CIFAR-10, Groups 1 and 2 yield nearly identical optimal sampling ratios.

More Results of Noise Multipliers and Total Amount of Noise. We present the detailed noise multipliers σ^2 or $(\sigma_1^2, \sigma_2^2, \sigma_3^2)$ and the total noise magnitude Λ added to the global model update in Table 5. As expected, DP-FedAvg applies the most stringent noise due to the need to satisfy the strictest client-level privacy constraint, resulting in a large noise multiplier and a correspondingly high total noise, which severely degrades model utility. In contrast, our proposed method, GDPFed, reduces the noise multipliers for groups with looser privacy requirements, thereby decreasing the overall noise. By further optimizing client sampling ratios, GDPFed-op achieves an even lower noise magnitude. Finally, GDPFed$^+$, which additionally incorporates sparsification, achieves the smallest total noise among all methods, offering the best trade-off between privacy and utility.

Optimal Sparsification Levels. We conduct extensive experiments to investigate how different sparsification levels affect model utility. Specifically, we report the test accuracies of GDPFed-op (with the full sparsification level $(1.0, 1.0, 1.0)$ or one can say no sparsification is applied) and GDPFed$^+$ under various sparsification configurations in Table 6. GDPFed-op is used as the baseline, and the relative differences in accuracy for each configuration are highlighted—positive values are shown in blue to indicate improvements, while negative values are shown in red to reflect degradations.

Our results reveal that moderate sparsification levels, such as $(0.9, 0.9, 0.9)$ and $(0.7, 0.8, 0.9)$, can lead to performance improvements across multiple datasets. In contrast, overly aggressive sparsification (e.g., $(0.1, 0.1, 0.1)$) significantly degrades performance, particularly on complex datasets such as SVHN and CIFAR-10. Notably, GDPFed$^+$ with the sparsification level $(0.1, 0.3, 0.5)$ yields only a minor performance drop of -0.30% compared to GDPFed-op, and clearly outperforms configurations like $(0.3, 0.3, 0.3)$ and $(0.1, 0.1, 0.1)$. This supports our intuition that groups with stricter privacy requirements should adopt more aggressive sparsification, while groups with looser privacy constraints can retain more parameters.

Based on these results, we observe that the optimal performance is achieved when GDPFed$^+$ uses the sparsification level $(0.7, 0.8, 0.9)$; therefore, we adopt it as the default configuration in our subsequent experiments. For other datasets not evaluated in this work, we recommend starting with $(0.7, 0.8, 0.9)$ and gradually adjusting the sparsification levels to balance utility and privacy based on task-specific characteristics.

Table 6: Test accuracy (%) of GDPFed$^+$ under various sparsification levels. The configuration $(1.0, 1.0, 1.0)$ (GDPFed-op) is used as the baseline. The relative accuracy difference compared to the baseline is shown in parentheses: blue indicates improvement, and red indicates degradation.

Sparsification Level	FMNIST	SVHN	CIFAR-10	Shakespeare	Average
$(1.0, 1.0, 1.0)$	$75.65_{\pm 0.53}$	$70.85_{\pm 0.57}$	$38.04_{\pm 0.46}$	$41.68_{\pm 0.44}$	56.56
$(0.9, 0.9, 0.9)$	$75.89_{\pm 0.53}$ $(+0.24)$	$71.04_{\pm 0.60}$ $(+0.19)$	$38.69_{\pm 0.33}$ $(+0.65)$	$41.94_{\pm 0.42}$ $(+0.26)$	56.89 $(+0.33)$
$(0.7, 0.7, 0.7)$	$75.79_{\pm 0.44}$ $(+0.14)$	$70.79_{\pm 0.41}$ (-0.06)	$38.40_{\pm 0.42}$ $(+0.36)$	$41.90_{\pm 0.34}$ $(+0.22)$	56.72 $(+0.16)$
$(0.5, 0.5, 0.5)$	$75.37_{\pm 0.41}$ (-0.28)	$70.39_{\pm 0.65}$ (-0.46)	$38.53_{\pm 0.74}$ $(+0.49)$	$41.40_{\pm 0.26}$ (-0.28)	56.42 (-0.14)
$(0.3, 0.3, 0.3)$	$74.92_{\pm 0.02}$ (-0.73)	$68.94_{\pm 1.29}$ (-1.91)	$37.50_{\pm 0.05}$ (-0.54)	$40.33_{\pm 0.73}$ (-1.35)	55.42 (-1.14)
$(0.1, 0.1, 0.1)$	$72.37_{\pm 0.35}$ (-3.28)	$61.65_{\pm 2.10}$ (-9.20)	$35.19_{\pm 1.58}$ (-2.85)	$36.13_{\pm 0.12}$ (-5.55)	51.34 (-5.22)
$(0.7, 0.8, 0.9)$	$\mathbf{75.83_{\pm 0.47}}$ $(+0.18)$	$\mathbf{71.10_{\pm 0.58}}$ $(+0.25)$	$\mathbf{38.78_{\pm 0.36}}$ $(+0.74)$	$\mathbf{42.00_{\pm 0.45}}$ $(+0.32)$	**56.93** $(+0.37)$
$(0.5, 0.7, 0.9)$	$75.79_{\pm 0.49}$ $(+0.14)$	$71.17_{\pm 0.58}$ $(+0.32)$	$38.63_{\pm 0.28}$ $(+0.59)$	$41.98_{\pm 0.42}$ $(+0.30)$	56.89 $(+0.33)$
$(0.3, 0.5, 0.7)$	$75.62_{\pm 0.43}$ (-0.03)	$70.83_{\pm 0.45}$ (-0.02)	$38.69_{\pm 0.18}$ $(+0.65)$	$41.83_{\pm 0.30}$ $(+0.15)$	56.74 $(+0.18)$
$(0.1, 0.3, 0.5)$	$75.52_{\pm 0.27}$ (-0.13)	$70.14_{\pm 0.76}$ (-0.71)	$38.38_{\pm 0.77}$ $(+0.34)$	$40.98_{\pm 0.38}$ (-0.70)	56.26 (-0.30)

Table 7: Test accuracy (%) under different heterogeneity levels (α). The last column shows the average accuracy across all data settings.

Method	α=0.3	α=0.5	α=0.7	α=0.9	IID	Average
P-FedAvg	$49.17_{\pm 0.58}$	$51.14_{\pm 0.45}$	$52.55_{\pm 0.29}$	$53.35_{\pm 0.55}$	$56.40_{\pm 0.34}$	52.52
DP-FedAvg	$26.18_{\pm 1.18}$	$28.66_{\pm 1.80}$	$29.43_{\pm 0.92}$	$31.21_{\pm 0.23}$	$32.10_{\pm 0.27}$	29.52
IDP-FedAvg	$27.23_{\pm 1.78}$	$29.93_{\pm 1.27}$	$31.22_{\pm 0.46}$	$33.21_{\pm 0.43}$	$37.49_{\pm 0.76}$	31.82
GDPFed$^+$	$\mathbf{28.59_{\pm 0.71}}$	$\mathbf{32.02_{\pm 0.57}}$	$\mathbf{34.43_{\pm 0.94}}$	$\mathbf{35.58_{\pm 0.78}}$	$\mathbf{38.78_{\pm 0.36}}$	**33.88**

More Experiments on HDPFL with Heterogeneous Data. We conduct additional experiments to evaluate the performance of our method under HDPFL with heterogeneous data settings. Specifically, we consider the CIFAR-10 dataset and use the *Dirichlet distribution* [40] to simulate non-IID data across clients, controlled by the non-IIDness parameter α. A larger α corresponds to lower data heterogeneity, and vice versa. We experiment with $\alpha = 0.9, 0.7, 0.5$, and a more extreme case of $\alpha = 0.3$. The results are presented in Table 7. As α increases (i.e., the data becomes more IID), the accuracy of all methods improves accordingly. Notably, GDPFed$^+$ consistently outperforms both DP-FedAvg and IDP-FedAvg, achieving an average improvement of $+2.06\%$ across all cases. These results demonstrate the effectiveness of GDPFed$^+$ in enhancing model utility under heterogeneous data distributions in client-level HDPFL.

Impact of Privacy Preference Distribution. In our default setting, we assume a uniform client distribution across groups such that $|\mathcal{G}_1| = |\mathcal{G}_2| = |\mathcal{G}_3|$ with $\epsilon_1 < \epsilon_2 < \epsilon_3$. To further evaluate the robustness and effectiveness of our method, we examine three alternative privacy preference distributions. These scenarios vary the proportion of clients in $(\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_3)$ as follows: (1) $1:4:1$–the moderate privacy group $\mathcal{G}_2$ comprises $4/6$ of clients, while the strictest $\mathcal{G}_1$ and loosest $\mathcal{G}_3$ groups each contain $1/6$; (2) $3:2:1$–the strictest privacy group $\mathcal{G}_1$ holds the largest share with $3/6$ of clients, $\mathcal{G}_2$ contains $2/6$ and $\mathcal{G}_3$ contains $1/6$; (3) $1:2:3$–the loosest privacy group $\mathcal{G}_3$ comprises $3/6$ of clients, $\mathcal{G}_1$ contains $1/6$ and $\mathcal{G}_3$ includes $2/6$.

These distributions reflect realistic deployment scenarios where privacy needs are not evenly distributed among users. The corresponding results are shown in Table 8. Across all four datasets and under all tested distributions, GDPFed$^+$ consistently outperforms DP-FedAvg. These findings demonstrate the strong performance and adaptability of GDPFed$^+$ under varying privacy preference distributions, further supporting its practicality for real-world FL systems.

It is worth noting that GDPFed$^+$ experiences a slight performance drop under the $3:2:1$ setting, compared with other settings. It is reasonable as GDPFed is designed to reduce the privacy budget waste, and in the $3:2:1$ setting, such waste is inherently not significant. From another perspective, the performance decline is primarily due to the increased number of clients from the strictest group, which requires adding more noise. For example, on CIFAR-10, GDPFed$^+$ samples 14 clients from the strict group under the $3:2:1$ distribution, compared to only 7 under the balanced $2:2:2$ setting. This behavior is driven by the influence of r_m in the optimization objective of Problem 1. Nevertheless, even in this challenging case, GDPFed$^+$ still achieves substantial improvements over DP-FedAvg, further demonstrating its effectiveness.

Table 8: Test accuracy (%) under different privacy preference distributions. The last column shows the average accuracy across all group settings.

Dataset	Method	1 : 4 : 1	2 : 2 : 2	3 : 2 : 1	1 : 2 : 3	Average
FMNIST	DP-FedAvg	$71.59_{\pm 0.46}$	$71.88_{\pm 0.15}$	$71.75_{\pm 0.58}$	$71.89_{\pm 0.51}$	71.78
	GDPFed$^+$	$\mathbf{76.06}_{\pm \mathbf{0.38}}$	$\mathbf{75.83}_{\pm \mathbf{0.47}}$	$\mathbf{74.50}_{\pm \mathbf{0.34}}$	$\mathbf{76.77}_{\pm \mathbf{0.42}}$	**75.79**
SVHN	DP-FedAvg	$40.76_{\pm 1.58}$	$40.80_{\pm 1.73}$	$40.10_{\pm 1.59}$	$40.73_{\pm 1.43}$	40.60
	GDPFed$^+$	$\mathbf{72.02}_{\pm \mathbf{0.74}}$	$\mathbf{71.10}_{\pm \mathbf{0.58}}$	$\mathbf{65.81}_{\pm \mathbf{1.17}}$	$\mathbf{75.02}_{\pm \mathbf{0.60}}$	**71.49**
CIFAR-10	DP-FedAvg	$33.47_{\pm 0.87}$	$32.10_{\pm 0.27}$	$33.53_{\pm 0.62}$	$33.24_{\pm 0.72}$	33.09
	GDPFed$^+$	$\mathbf{38.90}_{\pm \mathbf{0.19}}$	$\mathbf{38.78}_{\pm \mathbf{0.36}}$	$\mathbf{35.95}_{\pm \mathbf{1.14}}$	$\mathbf{40.64}_{\pm \mathbf{0.33}}$	**38.57**
Shakespeare	DP-FedAvg	$31.48_{\pm 3.29}$	$31.48_{\pm 2.85}$	$31.11_{\pm 3.43}$	$30.99_{\pm 3.14}$	31.27
	GDPFed$^+$	$\mathbf{42.36}_{\pm \mathbf{0.40}}$	$\mathbf{42.00}_{\pm \mathbf{0.45}}$	$\mathbf{39.47}_{\pm \mathbf{0.60}}$	$\mathbf{43.61}_{\pm \mathbf{0.43}}$	**41.86**

Table 9: Test accuracy (%) under different privacy budget scales. The last column reports the average accuracy.

Dataset	Method	0.5×	0.75×	1.0×	1.25×	1.5×	Average
FMNIST	DP-FedAvg	$57.72_{\pm 1.83}$	$68.87_{\pm 0.55}$	$71.88_{\pm 0.15}$	$73.76_{\pm 0.39}$	$74.99_{\pm 0.29}$	69.44
	GDPFed$^+$	$\mathbf{74.87}_{\pm \mathbf{0.63}}$	$\mathbf{75.55}_{\pm \mathbf{0.41}}$	$\mathbf{75.83}_{\pm \mathbf{0.47}}$	$\mathbf{75.97}_{\pm \mathbf{0.42}}$	$\mathbf{76.01}_{\pm \mathbf{0.30}}$	**75.65**
SVHN	DP-FedAvg	$18.14_{\pm 0.31}$	$25.20_{\pm 1.95}$	$40.80_{\pm 1.73}$	$55.26_{\pm 1.27}$	$63.52_{\pm 1.19}$	40.58
	GDPFed$^+$	$\mathbf{57.93}_{\pm \mathbf{1.55}}$	$\mathbf{67.54}_{\pm \mathbf{0.94}}$	$\mathbf{71.10}_{\pm \mathbf{0.58}}$	$\mathbf{72.90}_{\pm \mathbf{0.73}}$	$\mathbf{74.03}_{\pm \mathbf{0.75}}$	**68.70**
CIFAR-10	DP-FedAvg	$24.20_{\pm 0.67}$	$29.63_{\pm 0.65}$	$32.10_{\pm 0.27}$	$35.06_{\pm 0.57}$	$36.89_{\pm 0.70}$	31.58
	GDPFed$^+$	$\mathbf{32.96}_{\pm \mathbf{0.67}}$	$\mathbf{36.84}_{\pm \mathbf{0.12}}$	$\mathbf{38.78}_{\pm \mathbf{0.36}}$	$\mathbf{39.60}_{\pm \mathbf{0.16}}$	$\mathbf{40.16}_{\pm \mathbf{0.11}}$	**37.27**
Shakespeare	DP-FedAvg	$13.12_{\pm 8.03}$	$27.10_{\pm 1.51}$	$31.48_{\pm 2.85}$	$34.73_{\pm 0.68}$	$35.98_{\pm 0.26}$	28.88
	GDPFed$^+$	$\mathbf{37.60}_{\pm \mathbf{0.57}}$	$\mathbf{41.11}_{\pm \mathbf{0.17}}$	$\mathbf{42.00}_{\pm \mathbf{0.45}}$	$\mathbf{42.27}_{\pm \mathbf{0.59}}$	$\mathbf{42.55}_{\pm \mathbf{0.63}}$	**41.51**

Experiments on Various Privacy Budgets. We investigate how varying the per-group privacy budgets affects the performance of GDPFed$^+$. Specifically, we scale the default privacy budget settings for each dataset for each group using multiplicative factors: $0.5\times$, $0.75\times$, $1.0\times$, $1.25\times$, and $1.5\times$. For example, under the $0.5\times$ setting, the privacy budgets for FMNIST become $(0.25, 0.75, 1.50)$, which is $0.5 \times (0.5, 1.5, 3.0)$. The results across all datasets are summarized in Table 9.

GDPFed$^+$ consistently outperforms DP-FedAvg across all settings, with particularly notable gains when the privacy budgets are more restrictive (e.g., $0.5\times$ and $0.75\times$). As the scale increases, both methods exhibit improved performance due to the relaxation of privacy constraints, though GDPFed$^+$ maintains a clear advantage throughout. One key observation is that the utility gap between GDPFed$^+$ and DP-FedAvg becomes smaller at larger scales. This is because higher scaling leads to larger per-group privacy budgets, which in turn require less noise to satisfy the privacy guarantees. Consequently, the model utility of GDPFed$^+$ becomes closer to that of DP-FedAvg in such cases.

Impact of Clipping Threshold on Model Utility. We now analyze the influence of the clipping threshold C on model utility. Increasing C results in less aggressive clipping of local model updates, but also amplifies the magnitude of the noise required to satisfy differential privacy constraints.

We evaluate the impact of larger clipping thresholds by scaling C using multiplicative factors: $1.25\times$ and $1.5\times$. The results, presented in Table 10, indicate that as C increases, the performance of DP-FedAvg degrades significantly, particularly on FMNIST and SVHN datasets. In contrast, our method, GDPFed$^+$, exhibits only minor fluctuations in performance under each setting, demonstrating strong robustness. Notably, on SVHN, GDPFed$^+$ achieves an average test accuracy of 70.26%, representing a substantial improvement of $+37.39\%$ over DP-FedAvg. This performance gain is largely attributed to the use of optimized client sampling ratios, which yield more favorable noise multipliers for each privacy group. These results further underscore the effectiveness and practical resilience of GDPFed$^+$ under varying clipping thresholds.

B.5 Broader Impacts

This work addresses the challenge of client-level HDPFL, where individual clients have their privacy preferences. While our proposed method, GDPFed$^+$, effectively reduces the noise added to the global model and improves utility under strict privacy constraints, it assumes that clients' privacy budgets are externally specified and fixed. In real-world deployments, however, determining an adequate privacy

Table 10: Test accuracy (%) under different clipping thresholds C. The last column reports the average accuracy.

Dataset	Method	$1.0\times$	$1.25\times$	$1.5\times$	Average
FMNIST	DP-FedAvg	$71.88_{\pm 0.15}$	$68.35_{\pm 0.60}$	$63.62_{\pm 1.23}$	67.95
	GDPFed$^+$	$\mathbf{75.83_{\pm 0.47}}$	$\mathbf{75.58_{\pm 0.60}}$	$\mathbf{74.49_{\pm 0.81}}$	**75.30**
SVHN	DP-FedAvg	$40.80_{\pm 1.73}$	$31.69_{\pm 2.74}$	$26.11_{\pm 2.34}$	32.87
	GDPFed$^+$	$\mathbf{71.10_{\pm 0.58}}$	$\mathbf{70.63_{\pm 0.46}}$	$\mathbf{69.05_{\pm 0.73}}$	**70.26**
CIFAR-10	DP-FedAvg	$32.10_{\pm 0.27}$	$32.11_{\pm 0.31}$	$31.10_{\pm 0.82}$	31.77
	GDPFed$^+$	$\mathbf{38.78_{\pm 0.36}}$	$\mathbf{38.29_{\pm 0.69}}$	$\mathbf{37.50_{\pm 0.39}}$	**38.19**
Shakespeare	DP-FedAvg	$31.48_{\pm 2.85}$	$29.36_{\pm 2.11}$	$25.76_{\pm 0.64}$	28.87
	GDPFed$^+$	$\mathbf{42.00_{\pm 0.45}}$	$\mathbf{42.61_{\pm 0.22}}$	$\mathbf{42.33_{\pm 0.16}}$	**42.31**

level ϵ_i for each individual $i \in [n]$ is non-trivial and often subject to misunderstanding or misuse. To ensure the ethical application of our method, the assignment of individual privacy guarantees must be transparent, informed, and not subject to coercion.

While our work improves the privacy-utility trade-off in HDPFL, it also highlights the importance of coupling technical advances with social mechanisms to promote responsible, equitable, and informed use of DP in practice.

C Proof of Theorem 1

CREDITS: our proof follows the proof of Theorem 1 in [25].

Proof. Suppose the client is sampled without replacement with probability q_m at each round for each group $m \in [M]$. By Lemma 6 and Lemma 7, the t-th round of GDPFed satisfies $(\alpha_m, \rho_t(\alpha_m))$-RDP for each group m, where

$$\rho_t(\alpha) = \frac{3.5 q_m^2 \alpha_m}{\sigma_m^2}, \tag{1}$$

if $\sigma_m^2 \geq 0.7$ and $\alpha_m \leq 1 + (2/3)C^2\sigma_m^2 \log(1/q_m\alpha_m(1+\sigma_m^2))$. Then by Lemma 5, each group in GDPFed satisfies $(\alpha_m, T\rho_t(\alpha_m))$-RDP after T rounds of training. Next, in order to guarantee (ϵ_m, δ)-DP according to Lemma 4, we need

$$\frac{3.5 q_m^2 T \alpha_m}{\sigma_m^2} + \frac{\log(1/\delta)}{\alpha_m - 1} \leq \epsilon_m. \tag{2}$$

Suppose α_m and σ_m are chosen such that the conditions for Equation (1) are satisfied. Choose $\alpha_m = 1 + 2\log(1/\delta_m)/\epsilon_m$ and rearrange the inequality in Inequality (2), we need

$$\sigma_m^2 \geq \frac{7 q_m^2 T(\epsilon_m + 2\log(1/\delta))}{\epsilon_m^2}. \tag{3}$$

Then, using the constraint on ϵ concludes the proof. □

D Proof of Convergence Bound of Sparsification-Amplified GDPFed

D.1 Useful Lemmas

Lemma 8. *Given any two vectors $a, b \in \mathbb{R}^d$,*

$$2\langle a, b\rangle \leq \alpha \|a\|^2 + \frac{1}{\alpha}\|b\|^2, \forall\, \alpha > 0.$$

Lemma 9. *Given any two vectors $a, b \in \mathbb{R}^d$,*

$$\|a+b\|^2 \leq (1+\delta)\|a\|^2 + (1+\delta^{-1})\|b\|^2, \forall\, \delta > 0.$$

Lemma 10. *Given arbitrary set of n vectors $\{a_i\}_{i=1}^n$, $a_i \in \mathbb{R}^d$,*

$$\left\| \sum_{i=1}^n a_i \right\|^2 \leq n \sum_{i=1}^n \|a_i\|^2.$$

D.2 Proof of Theorem 3

Notations. We let $\nabla f_{m,i}(\theta_{m,i}^{t,s})$ represent the local gradient for client i in group m so that $\mathbb{E}[\mathbf{g}_{m,i}^{t,s}] = \nabla f_{m,i}(\theta_{m,i}^{t,s})$. For ease of expression, we let $\mathbf{d}_{m,i}^t = (1/\tau)\sum_{s=0}^{\tau-1} \mathbf{g}_{m,i}^{t,s}$ and $\mathbf{h}_{m,i}^t = (1/\tau)\sum_{s=0}^{\tau-1} \nabla f_{m,i}(\theta_{m,i}^{t,s})$, and $\mathbb{E}_t[\mathbf{d}_i^t] = \mathbf{h}_i^t$. We have the update rule $\theta^{t+1} = \theta^t - (1/\sum_{j=1}^M r_j)\sum_{m=1}^M \omega_m(\mathbf{y}_m^t \odot \mathbf{mk}_m^t)$, where ω_m is a reweighting parameter and $\mathbf{y}_m^t =$

$\sum_{i\in\mathcal{S}_m^t}\Delta_{m,i}^t$. More specifically:

$$\begin{aligned}
\frac{1}{\sum_{j=1}^M r_j}\sum_{m=1}^M \omega_m(\mathbf{y}_m^t \odot \mathbf{mk}_m^t) &= \sum_{m=1}^M \frac{1}{\sum_{j=1}^M r_j}\omega_m(\mathbf{y}_m^t \odot \mathbf{mk}_m^t) \\
&\leq \sum_{m=1}^M \frac{1}{r_m}\omega_m(\mathbf{y}_m^t \odot \mathbf{mk}_m^t) \\
&= \sum_{m=1}^M \omega_m \frac{1}{|\mathcal{S}_m|}\left(\sum_{i\in\mathcal{S}_m}\left(\eta\tau \mathrm{Top}_{k_m}(\mathbf{d}_{m,i}^t) + \mathrm{Top}_{k_m}(\mathbf{b}_{m,i}^t)\right)\right) \\
&= \sum_{m=1}^M \omega_m \frac{1}{|\mathcal{S}_m|}\left(\sum_{i\in\mathcal{S}_m}\eta\tau \mathrm{Top}_{k_m}(\mathbf{d}_{m,i}^t) + \sum_{i\in\mathcal{S}_m}\mathrm{Top}_{k_m}(\mathbf{b}_{m,i}^t)\right) \\
&= \sum_{m=1}^M \omega_m \frac{1}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\eta\tau \mathrm{Top}_{k_m}(\mathbf{d}_{m,i}^t) + \sum_{m=1}^M \omega_m \frac{1}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\mathrm{Top}_{k_m}(\mathbf{b}_{m,i}^t).
\end{aligned}$$

Proof. With Assumption 1, we have

$$\begin{aligned}
f(\theta^{t+1}) - f(\theta^t) &\leq \mathbb{E}_t\langle \nabla f(\theta^t), \theta^{t+1} - \theta^t\rangle + \frac{L}{2}\mathbb{E}_t\|\theta^{t+1} - \theta^t\|^2 \\
&\leq \mathbb{E}_t\left\langle \nabla f(\theta^t), -\sum_{m=1}^M \omega_m \tilde{\Delta}_m^t\right\rangle + \frac{L}{2}\mathbb{E}_t\|\theta^{t+1} - \theta^t\|^2 \\
&= \underbrace{-\mathbb{E}_t\left\langle \nabla f(\theta^t), \sum_{m=1}^M \omega_m\left[\frac{1}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\eta\tau \mathrm{Top}_{k_m}(\mathbf{d}_{m,i}^t) + \frac{1}{|\mathcal{S}_m|}\sum_{j\in\mathcal{S}_m}\mathrm{Top}_{k_m}(\mathbf{b}_{m,j}^t)\right]\right\rangle}_{T_1} \\
&\quad + \underbrace{\frac{L}{2}\mathbb{E}_t\|\theta^{t+1} - \theta^t\|^2}_{T_2}.
\end{aligned}$$

To solve T_1, we have

$$\begin{aligned}
T_1 &= -\mathbb{E}_t\left\langle \nabla f(\theta^t), \sum_{m=1}^M \frac{\omega_m}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\left(\eta\tau \mathrm{Top}_{k_m}(\mathbf{d}_{m,i}^t)\right)\right\rangle - \mathbb{E}_t\left\langle \nabla f(\theta^t), \sum_{m=1}^M \frac{\omega_m}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\mathrm{Top}_{k_m}(\mathbf{b}_{m,i}^t)\right\rangle \\
&= -\mathbb{E}_t\left\langle \nabla f(\theta^t), \sum_{m=1}^M \frac{\omega_m}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\left(\eta\tau \mathrm{Top}_{k_m}(\mathbf{d}_{m,i}^t) + \eta\tau\mathbf{d}_{m,i}^t - \eta\tau\mathbf{d}_{m,i}^t\right)\right\rangle \\
&= -\mathbb{E}_t\left\langle \nabla f(\theta^t), \sum_{m=1}^M \frac{\omega_m}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\left(\eta\tau\mathbf{d}_{m,i}^t\right)\right\rangle + \mathbb{E}_t\left\langle \nabla f(\theta^t), \sum_{m=1}^M \frac{\omega_m}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\left(\eta\tau\mathbf{d}_{m,i}^t - \eta\tau \mathrm{Top}_{k_m}(\mathbf{d}_{m,i}^t)\right)\right\rangle \\
&= -\eta\tau\left\langle \nabla f(\theta^t), \sum_{m=1}^M \omega_m \mathbb{E}_t\left[\frac{1}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\mathbf{d}_{m,i}^t\right]\right\rangle + \eta\tau\mathbb{E}_t\left\langle \nabla f(\theta^t), \sum_{m=1}^M \frac{\omega_m}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\left(\mathbf{d}_{m,i}^t - \mathrm{Top}_{k_m}(\mathbf{d}_{m,i}^t)\right)\right\rangle \\
&= \underbrace{-\eta\tau\mathbb{E}_t\left\langle \nabla f(\theta^t), \sum_{m=1}^M \frac{\omega_m}{|\mathcal{G}_m|}\sum_{i\in\mathcal{G}_m}\mathbf{h}_{m,i}^t\right\rangle}_{A_1} + \underbrace{\eta\tau\mathbb{E}_t\left\langle \nabla f(\theta^t), \sum_{m=1}^M \frac{\omega_m}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\left(\mathbf{d}_{m,i}^t - \mathrm{Top}_{k_m}(\mathbf{d}_{m,i}^t)\right)\right\rangle}_{A_2}.
\end{aligned}$$

For A_1, we have

$$
\begin{aligned}
A_1 &= -\frac{\eta\tau}{2}\left\|\nabla f(\theta^t)\right\|^2 - \frac{\eta\tau}{2}\left\|\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{G}_m|}\sum_{i\in\mathcal{G}_m}\mathbf{h}_{m,i}^t\right\|^2 + \frac{\eta\tau}{2}\left\|\nabla f(\theta^t) - \sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{G}_m|}\sum_{i\in\mathcal{G}_m}\mathbf{h}_{m,i}^t\right\|^2 \\
&= -\frac{\eta\tau}{2}\left\|\nabla f(\theta^t)\right\|^2 + \frac{\eta\tau}{2}\left\|\nabla f(\theta^t) - \sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{G}_m|}\sum_{i\in\mathcal{G}_m}\frac{1}{\tau}\sum_{s=0}^{\tau-1}\nabla f_{m,i}(\theta_{m,i}^{t,s})\right\|^2 \\
&= -\frac{\eta\tau}{2}\left\|\nabla f(\theta^t)\right\|^2 + \frac{\eta\tau}{2}\left\|\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{G}_m|}\sum_{i\in\mathcal{G}_m}\nabla f_{m,i}(\theta^t) - \sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{G}_m|}\sum_{i\in\mathcal{G}_m}\frac{1}{\tau}\sum_{s=0}^{\tau-1}\nabla f_{m,i}(\theta_{m,i}^{t,s})\right\|^2 \\
&= -\frac{\eta\tau}{2}\left\|\nabla f(\theta^t)\right\|^2 + \frac{\eta\tau}{2}\left\|\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{G}_m|}\sum_{i\in\mathcal{G}_m}\frac{1}{\tau}\sum_{s=0}^{\tau-1}\left(\nabla f_{m,i}(\theta^t) - \nabla f_{m,i}(\theta_{m,i}^{t,s})\right)\right\|^2 \\
&\le -\frac{\eta\tau}{2}\left\|\nabla f(\theta^t)\right\|^2 + \frac{\eta\tau}{2}\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{G}_m|}\sum_{i\in\mathcal{G}_m}\frac{1}{\tau}\sum_{s=0}^{\tau-1}\left\|\nabla f_{m,i}(\theta^t) - \nabla f_{m,i}(\theta_{m,i}^{t,s})\right\|^2 \\
&\le -\frac{\eta\tau}{2}\left\|\nabla f(\theta^t)\right\|^2 + \frac{\eta\tau L^2}{2}\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{G}_m|}\sum_{i\in\mathcal{G}_m}\frac{1}{\tau}\sum_{s=0}^{\tau-1}\left\|\theta^t - \theta_{m,i}^{t,s}\right\|^2,
\end{aligned}
$$

where the first inequality holds according to Lemma 10 and the last inequality holds according to Assumption 1.

For A_2, we have

$$
\begin{aligned}
A_2 &= \eta\tau\mathbb{E}_t\left\langle \nabla f(\theta^t), \sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\left(\mathbf{d}_{m,i}^t - \mathrm{Top}_{k_m}(\mathbf{d}_{m,i}^t)\right)\right\rangle \\
&\le \frac{\eta\tau}{2}\mathbb{E}_t\left(\gamma\left\|\nabla f(\theta^t)\right\|^2 + \gamma^{-1}\mathbb{E}_t\left\|\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\left(\mathbf{d}_{m,i}^t - \mathrm{Top}_{k_m}(\mathbf{d}_{m,i}^t)\right)\right\|^2\right) \\
&\le \frac{\eta\tau}{2}\left(\gamma\left\|\nabla f(\theta^t)\right\|^2 + \gamma^{-1}\mathbb{E}_t\left[\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\left\|\mathbf{d}_{m,i}^t - \mathrm{Top}_{k_m}(\mathbf{d}_{m,i}^t)\right\|^2\right]\right) \\
&\le \frac{\eta\tau\gamma}{2}\left\|\nabla f(\theta^t)\right\|^2 + \frac{\eta\tau}{2\gamma}\mathbb{E}_t\left[\sum_{m=1}^{M}\frac{\omega_m\phi_m}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\left\|\mathbf{d}_{m,i}^t\right\|^2\right] \\
&\le \frac{\eta\tau\gamma}{2}\left\|\nabla f(\theta^t)\right\|^2 + \frac{\eta\tau}{2\gamma}\sum_{m=1}^{M}\frac{\omega_m\phi_m}{|\mathcal{G}_m|}\sum_{i\in\mathcal{G}_m}\mathbb{E}_t\left\|\mathbf{d}_{m,i}^t\right\|^2,
\end{aligned}
$$

where the first inequality follows Lemma 8, the second inequality follows Lemma 10, and the third inequality holds with bounded sparsification (Lemma 3) and $\phi_m = (1-k/d)^2$.

Plugging A_1, A_2 back to T_1, we have

$$
\begin{aligned}
T_1 &\le -\frac{\eta\tau}{2}\left\|\nabla f(\theta^t)\right\|^2 + \frac{\eta\tau L^2}{2}\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{G}_m|}\sum_{i\in\mathcal{G}_m}\frac{1}{\tau}\sum_{s=0}^{\tau-1}\left\|\theta^t - \theta_{m,i}^{t,s}\right\|^2 + \frac{\eta\tau\gamma}{2}\left\|\nabla f(\theta^t)\right\|^2 + \frac{\eta\tau}{2\gamma}\sum_{m=1}^{M}\frac{\omega_m\phi_m}{|\mathcal{G}_m|}\sum_{i\in\mathcal{G}_m}\mathbb{E}_t\left\|\mathbf{d}_{m,i}^t\right\|^2 \\
&= \frac{\eta\tau(\gamma-1)}{2}\left\|\nabla f(\theta^t)\right\|^2 + \frac{\eta\tau L^2}{2}\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{G}_m|}\sum_{i\in\mathcal{G}_m}\frac{1}{\tau}\sum_{s=0}^{\tau-1}\left\|\theta^t - \theta_{m,i}^{t,s}\right\|^2 + \frac{\eta\tau}{2\gamma}\sum_{m=1}^{M}\frac{\omega_m\phi_m}{|\mathcal{G}_m|}\sum_{i\in\mathcal{G}_m}\mathbb{E}_t\left\|\mathbf{d}_{m,i}^t\right\|^2.
\end{aligned}
$$

For T_2, we have

$$
\begin{aligned}
T_2 &= \frac{L}{2}\mathbb{E}_t\left\|\sum_{m=1}^{M}\omega_m\left[\frac{1}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\eta\tau\mathrm{Top}_{k_m}(\mathbf{d}_{m,i}^t)+\frac{1}{|\mathcal{S}_m|}\sum_{j\in\mathcal{G}_m}\mathrm{Top}_{k_m}(\mathbf{b}_{m,j}^t)\right]\right\|^2 \\
&= \frac{L\eta^2\tau^2}{2}\mathbb{E}_t\left\|\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\left(\mathrm{Top}_{m,k}^{+}\left(\mathbf{d}_{m,i}^t\right)-\mathbf{d}_{m,i}^t+\mathbf{d}_{m,i}^t\right)\right\|^2+\frac{L}{2}\left\|\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{S}_m|}\sum_{j\in\mathcal{G}_m}\mathrm{Top}_{k_m}(\mathbf{b}_{m,j}^t)\right\|^2 \\
&\le \frac{L\eta^2\tau^2}{2}\mathbb{E}_t\left[\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\left\|\left(\mathrm{Top}_{k_m}\left(\mathbf{d}_{m,i}^t\right)-\mathbf{d}_{m,i}^t+\mathbf{d}_{m,i}^t\right)\right\|^2\right]+\frac{L}{2}\left\|\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{S}_m|}\sum_{j\in\mathcal{G}_m}\mathrm{Top}_{k_m}(\mathbf{b}_{m,j}^t)\right\|^2 \\
&\le L\eta^2\tau^2\mathbb{E}_t\left[\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\left\|\left(\mathrm{Top}_{k_m}\left(\mathbf{d}_{m,i}^t\right)-\mathbf{d}_{m,i}^t\right)\right\|^2\right]+L\eta^2\tau^2\mathbb{E}_t\left[\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\left\|\mathbf{d}_{m,i}^t\right\|^2\right] \\
&\quad+\frac{L}{2}\left\|\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{S}_m|}\sum_{j\in\mathcal{G}_m}\mathrm{Top}_{k_m}(\mathbf{b}_{m,j}^t)\right\|^2 \\
&\le L\eta^2\tau^2\mathbb{E}_t\left[\sum_{m=1}^{M}\frac{\omega_m\phi_m}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\left\|\mathbf{d}_{m,i}^t\right\|^2\right]+L\eta^2\tau^2\mathbb{E}_t\left[\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\left\|\mathbf{d}_{m,i}^t\right\|^2\right]+\frac{L}{2}\left\|\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{S}_m|}\sum_{j\in\mathcal{G}_m}\mathrm{Top}_{k_m}(\mathbf{b}_{m,j}^t)\right\|^2 \\
&= L\eta^2\tau^2\mathbb{E}_t\left[\sum_{m=1}^{M}\frac{\omega_m\left(\phi_m+1\right)}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\left\|\mathbf{d}_{m,i}^t\right\|^2\right]+\frac{L}{2}\left\|\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{S}_m|}\sum_{j\in\mathcal{G}_m}\mathrm{Top}_{k_m}(\mathbf{b}_{m,j}^t)\right\|^2,
\end{aligned}
$$

where the first inequality follows Lemma 10, the second inequality follows Lemma 9, the third inequality follows Lemma 3.

Combine T_1 and T_2, we have

$$
\begin{aligned}
T_1+T_2 &\le \frac{\eta\tau(\gamma-1)}{2}\left\|\nabla f(\theta^t)\right\|^2+\frac{\eta\tau L^2}{2}\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{G}_m|}\sum_{i\in\mathcal{G}_m}\frac{1}{\tau}\sum_{s=0}^{\tau-1}\left\|\theta^t-\theta_{m,i}^{t,s}\right\|^2+\frac{\eta\tau}{2\gamma}\sum_{m=1}^{M}\frac{\omega_m\phi_m}{|\mathcal{G}_m|}\sum_{i\in\mathcal{G}_m}\mathbb{E}_t\left\|\mathbf{d}_{m,i}^t\right\|^2 \\
&\quad+L\eta^2\tau^2\mathbb{E}_t\left[\sum_{m=1}^{M}\frac{\omega_m\left(\phi_m+1\right)}{|\mathcal{S}_m|}\sum_{i\in\mathcal{S}_m}\left\|\mathbf{d}_{m,i}^t\right\|^2\right]+\left\|\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{S}_m|}\sum_{j\in\mathcal{G}_m}\mathrm{Top}_{k_m}(\mathbf{b}_{m,j}^t)\right\|^2 \\
&\le \frac{\eta\tau(\gamma-1)}{2}\left\|\nabla f(\theta^t)\right\|^2+\frac{\eta\tau L^2}{2}\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{G}_m|}\sum_{i\in\mathcal{G}_m}\frac{1}{\tau}\sum_{s=0}^{\tau-1}\left\|\theta^t-\theta_{m,i}^{t,s}\right\|^2 \\
&\quad+\underbrace{2L\eta^2\tau^2\sum_{m=1}^{M}\frac{\omega_m\left(\phi_m+1\right)}{|\mathcal{G}_m|}\sum_{i\in\mathcal{G}_m}\mathbb{E}_t\left\|\mathbf{d}_{m,i}^t\right\|^2}_{B_1}+\frac{L}{2}\left\|\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{S}_m|}\sum_{j\in\mathcal{G}_m}\mathrm{Top}_{k_m}(\mathbf{b}_{m,j}^t)\right\|^2,
\end{aligned}
$$

if $\gamma \ge \phi_m/\left[2L\tau\eta(\phi_m+1)\right]$.

Consequently, for B_1, we have

$$
\begin{aligned}
B_1 &= \sum_{m=1}^{M} \frac{\omega_m\left(\phi_m+1\right)}{|\mathcal{G}_m|} \sum_{i\in\mathcal{G}_m} \mathbb{E}_t \left\|\mathbf{d}_{m,i}^t - \mathbf{h}_{m,i}^t + \mathbf{h}_{m,i}^t\right\|^2 \\
&\leq 2\sum_{m=1}^{M} \frac{\omega_m\left(\phi_m+1\right)}{|\mathcal{G}_m|} \sum_{i\in\mathcal{G}_m} \mathbb{E}_t \left(\left\|\mathbf{d}_{m,i}^t - \mathbf{h}_{m,i}^t\right\|^2 + \left\|\mathbf{h}_{m,i}^t\right\|^2\right) \\
&= 2\sum_{m=1}^{M} \frac{\omega_m\left(\phi_m+1\right)}{|\mathcal{G}_m|} \sum_{i\in\mathcal{G}_m} \mathbb{E}_t \left(\left\|\mathbf{d}_{m,i}^t - \mathbf{h}_{m,i}^t\right\|^2 + \left\|\mathbf{h}_{m,i}^t - \nabla f_{m,i}(\theta^t) + \nabla f_{m,i}(\theta^t)\right\|^2\right) \\
&\leq 2\sum_{m=1}^{M} \frac{\omega_m\left(\phi_m+1\right)}{|\mathcal{G}_m|} \sum_{i\in\mathcal{G}_m} \mathbb{E}_t \left(\left\|\mathbf{d}_{m,i}^t - \mathbf{h}_{m,i}^t\right\|^2 + 2\left\|\mathbf{h}_{m,i}^t - \nabla f_{m,i}(\theta^t)\right\|^2 + 2\left\|\nabla f_{m,i}(\theta^t)\right\|^2\right) \\
&\leq 2\sum_{m=1}^{M} \frac{\omega_m\left(\phi_m+1\right)}{|\mathcal{G}_m|} \sum_{i\in\mathcal{G}_m} \mathbb{E}_t \left[\left\|\mathbf{d}_{m,i}^t - \mathbf{h}_{m,i}^t\right\|^2 + 2\left\|\mathbf{h}_{m,i}^t - \nabla f_{m,i}(\theta^t)\right\|^2\right] \\
&\quad + 4\sum_{m=1}^{M} \frac{\omega_m\left(\phi_m+1\right)}{|\mathcal{G}_m|} \sum_{i\in\mathcal{G}_m} \left\|\nabla f_{m,i}(\theta^t)\right\|^2,
\end{aligned}
$$

where these inequalities follows Lemma 9.

Given that

$$
\mathbb{E}_t \left\|\mathbf{d}_{m,i}^t - \mathbf{h}_{m,i}^t\right\|^2 = \mathbb{E}_t \left\|\frac{1}{\tau}\sum_{s=0}^{\tau-1}\left(\mathbf{g}_{m,i}^{t,s} - \nabla f_i(\theta_{m,i}^{t,s})\right)\right\|^2 \leq \frac{1}{\tau}\sum_{s=0}^{\tau-1}\mathbb{E}_t \left\|\mathbf{g}_{m,i}^{t,s} - \nabla f_i(\theta_{m,i}^{t,s})\right\|^2 \leq \frac{1}{\tau}\sum_{s=0}^{\tau-1} d\zeta_{m,i}^2 = d\zeta_{m,i}^2,
$$

where the first inequality follows Lemma 10 and the second inequality follows Assumption 2, we have

$$
\begin{aligned}
B_1 &\leq 2\sum_{m=1}^{M} \frac{\omega_m\left(\phi_m+1\right)}{|\mathcal{G}_m|} \sum_{i\in\mathcal{G}_m} \mathbb{E}_t \left[\zeta_{m,i}^2 + 2\left\|\mathbf{h}_{m,i}^t - \nabla f_{m,i}(\theta^t)\right\|^2\right] + 4\sum_{m=1}^{M} \frac{\omega_m\left(\phi_m+1\right)}{|\mathcal{G}_m|} \sum_{i\in\mathcal{G}_m} \left\|\nabla f_{m,i}(\theta^t)\right\|^2 \\
&\leq 2\sum_{m=1}^{M} \frac{\omega_m\left(\phi_m+1\right)}{|\mathcal{G}_m|} \sum_{i\in\mathcal{G}_m} \left(\zeta_{m,i}^2 + \frac{2L^2}{\tau}\sum_{s=0}^{\tau-1}\mathbb{E}_t \left\|\theta^t - \theta_{m,i}^{t,s}\right\|^2\right) + 4\sum_{m=1}^{M} \frac{\omega_m\left(\phi_m+1\right)}{|\mathcal{G}_m|} \sum_{i\in\mathcal{G}_m} \left\|\nabla f_{m,i}(\theta^t)\right\|^2 \\
&\leq 2\sum_{m=1}^{M} \frac{\omega_m\left(\phi_m+1\right)}{|\mathcal{G}_m|} \sum_{i\in\mathcal{G}_m} \zeta_{m,i}^2 + \frac{4L^2}{\tau}\sum_{m=1}^{M} \frac{\omega_m\left(\phi_m+1\right)}{|\mathcal{G}_m|} \sum_{i\in\mathcal{G}_m}\sum_{s=0}^{\tau-1}\mathbb{E}_t \left\|\theta^t - \theta_{m,i}^{t,s}\right\|^2 \\
&\quad + 4\sum_{m=1}^{M} \frac{\omega_m\left(\phi_m+1\right)}{|\mathcal{G}_m|} \sum_{i\in\mathcal{G}_m} \left\|\nabla f_{m,i}(\theta^t)\right\|^2.
\end{aligned}
$$

Plugging B_1 back, we have

$$
\begin{aligned}
T_1 + T_2 \leq{}& \frac{\eta\tau(\gamma-1)}{2}\left\|\nabla f(\theta^t)\right\|^2 + 4L\eta^2\tau^2 \sum_{m=1}^{M} \frac{\omega_m(\phi_m+1)}{|\mathcal{G}_m|} \sum_{i\in\mathcal{G}_m} d\zeta_{m,i}^2 \\
&+ \left(\frac{\eta\tau L^2}{2} + \frac{4L^2}{\tau}2L\eta^2\tau^2\right)\frac{1}{\tau}\sum_{s=0}^{\tau-1}\underbrace{\sum_{m=1}^{M} \frac{\omega_m(\phi_m+1)}{|\mathcal{G}_m|} \sum_{i\in\mathcal{G}_m} \mathbb{E}_t \left\|\theta^t - \theta_{m,i}^{t,s}\right\|^2}_{C_1} \\
&+ 8L\eta^2\tau^2 \sum_{m=1}^{M} \frac{\omega_m(\phi_m+1)}{|\mathcal{G}_m|} \sum_{i\in\mathcal{G}_m} \left\|\nabla f_{m,i}(\theta^t)\right\|^2 + \frac{L}{2}\left\|\sum_{m=1}^{M} \frac{\omega_m}{|\mathcal{S}_m|} \sum_{j\in\mathcal{G}_m} \mathrm{Top}_{k_m}(\mathbf{b}_{m,j}^t)\right\|^2.
\end{aligned}
$$

For C_1, we first handle

$$\begin{aligned}
\mathbb{E}_t \left\|\theta^t - \theta_{m,i}^{t,s}\right\|^2 &= \mathbb{E}_t \left\|\theta_{m,i}^{t,s-1} - \theta^t - \eta g_{m,i}^{t,s-1} + \eta\nabla f_{m,i}(\theta_{m,i}^{t,s-1}) - \eta\nabla f_{m,i}(\theta_{m,i}^{t,s-1}) + \eta\nabla f_{m,i}(\theta^t) - \eta\nabla f_{m,i}(\theta^t)\right\|^2 \\
&= \mathbb{E}_t \left\|\theta_{m,i}^{t,s-1} - \theta^t - \eta\nabla f_{m,i}(\theta_{m,i}^{t,s-1}) + \eta\nabla f_{m,i}(\theta^t) - \eta\nabla f_{m,i}(\theta^t)\right\|^2 + \eta^2 \left\|g_{m,i}^{t,s-1} - \nabla f_{m,i}(\theta_{m,i}^{t,s-1})\right\|^2 \\
&\leq \left(1 + \frac{1}{2\tau - 1}\right) \mathbb{E}_t \left\|\theta_{m,i}^{t,s-1} - \theta^t\right\|^2 + 2\tau\eta^2 \left\|\nabla f_{m,i}(\theta_{m,i}^{t,s-1}) + \nabla f_{m,i}(\theta^t) - \nabla f_{m,i}(\theta^t)\right\|^2 + d\eta^2\zeta_{m,i}^2 \\
&\leq \left(1 + \frac{1}{2\tau - 1}\right) \mathbb{E}_t \left\|\theta_{m,i}^{t,s-1} - \theta^t\right\|^2 + 4\tau\eta^2 L^2 \left\|\theta_{m,i}^{t,s-1} - \theta^t\right\|^2 + 4\tau\eta^2 \left\|\nabla f_{m,i}(\theta^t)\right\|^2 + d\eta^2\zeta_{m,i}^2 \\
&= \left(1 + \frac{1}{2\tau - 1} + 4\tau\eta^2 L^2\right) \mathbb{E}_t \left\|\theta_{m,i}^{t,s-1} - \theta^t\right\|^2 + 4\tau\eta^2 \left\|\nabla f_{m,i}(\theta^t)\right\|^2 + d\eta^2\zeta_{m,i}^2.
\end{aligned}$$

Therefore, we have

$$\begin{aligned}
C_1 &= \sum_{m=1}^{M} \frac{\omega_m(\phi_m + 1)}{|\mathcal{G}_m|} \sum_{i \in \mathcal{G}_m} \mathbb{E}_t \left\|\theta^t - \theta_{m,i}^{t,s}\right\|^2 \\
&\leq \sum_{m=1}^{M} \frac{\omega_m(\phi_m + 1)}{|\mathcal{G}_m|} \sum_{i \in \mathcal{G}_m} \left(\left(1 + \frac{1}{2\tau - 1} + 4\tau\eta^2 L^2\right) \mathbb{E}_t \left\|\theta_{m,i}^{t,s-1} - \theta^t\right\|^2 + 4\tau\eta^2 \left\|\nabla f_{m,i}(\theta^t)\right\|^2 + d\eta^2\zeta_{m,i}^2\right) \\
&= \left(1 + \frac{1}{2\tau - 1} + 4\tau\eta^2 L^2\right) \sum_{m=1}^{M} \frac{\omega_m(\phi_m + 1)}{|\mathcal{G}_m|} \sum_{i \in \mathcal{G}_m} \mathbb{E}_t \left\|\theta_{m,i}^{t,s-1} - \theta^t\right\|^2 + 4\tau\eta^2 \sum_{m=1}^{M} \frac{\omega_m(\phi_m + 1)}{|\mathcal{G}_m|} \sum_{i \in \mathcal{G}_m} \left\|\nabla f_{m,i}(\theta^t)\right\|^2 \\
&\quad + d\eta^2 \sum_{m=1}^{M} \frac{\omega_m(\phi_m + 1)}{|\mathcal{G}_m|} \sum_{i \in \mathcal{G}_m} \zeta_{m,i}^2 \\
&\leq \left(1 + \frac{1}{\tau - 1}\right) \sum_{m=1}^{M} \frac{\omega_m(\phi_m + 1)}{|\mathcal{G}_m|} \sum_{i \in \mathcal{G}_m} \mathbb{E}_t \left\|\theta_{m,i}^{t,s-1} - \theta^t\right\|^2 + 4\tau\eta^2 \sum_{m=1}^{M} \frac{\omega_m(\phi_m + 1)}{|\mathcal{G}_m|} \sum_{i \in \mathcal{G}_m} \left\|\nabla f_{m,i}(\theta^t)\right\|^2 \\
&\quad + d\eta^2 \sum_{m=1}^{M} \frac{\omega_m(\phi_m + 1)}{|\mathcal{G}_m|} \sum_{i \in \mathcal{G}_m} \zeta_{m,i}^2,
\end{aligned}$$

when $\eta \leq 1/3\tau L$.

Unrolling the recursion, we obtain the following

$$\begin{aligned}
C_1 &\leq \sum_{h=0}^{s-1} \left(1 + \frac{1}{\tau - 1}\right)^h \left(4\tau\eta^2 \sum_{m=1}^{M} \frac{\omega_m(\phi_m + 1)}{|\mathcal{G}_m|} \sum_{i \in \mathcal{G}_m} \left\|\nabla f_{m,i}(\theta^t)\right\|^2 + d\eta^2 \sum_{m=1}^{M} \frac{\omega_m(\phi_m + 1)}{|\mathcal{G}_m|} \sum_{i \in \mathcal{G}_m} \zeta_{m,i}^2\right) \\
&\leq (\tau - 1)\left(\left(1 + \frac{1}{\tau - 1}\right)^\tau - 1\right)\left(4\tau\eta^2 \sum_{m=1}^{M} \frac{\omega_m(\phi_m + 1)}{|\mathcal{G}_m|} \sum_{i \in \mathcal{G}_m} \left\|\nabla f_{m,i}(\theta^t)\right\|^2 + d\eta^2 \sum_{m=1}^{M} \frac{\omega_m(\phi_m + 1)}{|\mathcal{G}_m|} \sum_{i \in \mathcal{G}_m} \zeta_{m,i}^2\right) \\
&\leq (4\tau - 4)\left(4\tau\eta^2 \sum_{m=1}^{M} \frac{\omega_m(\phi_m + 1)}{|\mathcal{G}_m|} \sum_{i \in \mathcal{G}_m} \left\|\nabla f_{m,i}(\theta^t)\right\|^2 + d\eta^2 \sum_{m=1}^{M} \frac{\omega_m(\phi_m + 1)}{|\mathcal{G}_m|} \sum_{i \in \mathcal{G}_m} \zeta_{m,i}^2\right) \\
&\leq (4\tau - 4)4\tau\eta^2 \sum_{m=1}^{M} \frac{\omega_m(\phi_m + 1)}{|\mathcal{G}_m|} \sum_{i \in \mathcal{G}_m} \left\|\nabla f_{m,i}(\theta^t)\right\|^2 + (4\tau - 4)d\eta^2 \sum_{m=1}^{M} \frac{\omega_m(\phi_m + 1)}{|\mathcal{G}_m|} \sum_{i \in \mathcal{G}_m} \zeta_{m,i}^2,
\end{aligned}$$

where the fourth inequality holds since $\left(1 + \frac{1}{\tau - 1}\right)^\tau \leq 5$ when $\tau > 1$.

Plugging C_1 back and rearranging, we have

$$
\begin{aligned}
T_1 + T_2 \leq & \left[\frac{\eta\tau(\gamma-1)}{2} + \beta^2\left[\left(\frac{\eta\tau L^2}{2} + \frac{4L^2}{\tau}2L\eta^2\tau^2\right)(4\tau-4)4\tau\eta^2 + 8L\eta^2\tau^2\right]\right]\left\|\nabla f(\theta^t)\right\|^2 \\
& + \left[\left(\frac{\eta\tau L^2}{2} + \frac{4L^2}{\tau}2L\eta^2\tau^2\right)(4\tau-4)4\tau\eta^2 + 8L\eta^2\tau^2\right]\kappa^2 \\
& + \left[4L\eta^2\tau^2 + \left(\frac{\eta\tau L^2}{2} + \frac{4L^2}{\tau}2L\eta^2\tau^2\right)(4\tau-4)\eta^2\right] d\sum_{m=1}^{M}\frac{\omega_m\left(\phi_m+1\right)}{|\mathcal{G}_m|}\sum_{i\in\mathcal{G}_m}\zeta_{m,i}^2 \\
& + \frac{L}{2}\underbrace{\left\|\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{S}_m|}\sum_{j\in\mathcal{G}_m}\mathrm{Top}_{k_m}(\mathbf{b}_{m,j}^t)\right\|^2}_{D_1}.
\end{aligned}
$$

For D_1, we have

$$
\begin{aligned}
\mathbb{E}\left\|\sum_{m=1}^{M}\frac{\omega_m}{|\mathcal{S}_m|}\sum_{j\in\mathcal{S}_m}\mathrm{Top}_{k_m}(\mathbf{b}_{m,j}^t)\right\|^2 &= \mathbb{E}\left[\sum_{m=1}^{M}\frac{\omega_m^2}{|\mathcal{S}_m|^2}\left\|\sum_{j\in\mathcal{S}_m}\mathrm{Top}_{k_m}(\mathbf{b}_{m,j}^t)\right\|^2\right] = \mathbb{E}\left[\sum_{m=1}^{M}\frac{\omega_m^2}{|\mathcal{S}_m|^2}\sum_{j\in\mathcal{S}_m}\left\|\mathrm{Top}_{k_m}(\mathbf{b}_{m,j}^t)\right\|^2\right] \\
&= \mathbb{E}\left[\sum_{m=1}^{M}\frac{\omega_m^2}{|\mathcal{S}_m|^2}\sum_{j\in\mathcal{S}_m}\left\|\mathcal{N}\left(0, C^2\sigma_m^2/r_m\cdot\mathbf{I}^k\right)\right\|^2\right] \\
&= \sum_{m=1}^{M}k_m\omega_m^2C^2\sigma_m^2\mathbb{E}\left[\frac{1}{|\mathcal{S}_m|^2}\right] \\
&= \sum_{m=1}^{M}\frac{k_m\omega_m^2C^2\sigma_m^2}{r_m^2}.
\end{aligned}
$$

Plugging D_1 back, and if $\eta \leq \min\{1/\left[4L\beta^2\left(\tau+1\right)+8L\tau\beta^2\right], 1/(16\tau L)\}$, we have

$$
\begin{aligned}
f(\theta^{t+1}) - f(\theta^t) \leq & -\frac{1}{8}\eta\tau\left\|\nabla f(\theta^t)\right\|^2 + \left(\frac{L\eta^2\tau^2}{2} + \frac{L\tau\eta^2}{2} + 8L\eta\tau\right)\kappa^2 \\
& + \left(4L\eta^2\tau^2 + \frac{2}{16}L\tau\eta^2 + \frac{1}{8}L\eta^2\right)d\sum_{m=1}^{M}\frac{\omega_m(\phi_m+1)}{|\mathcal{G}_m|}\sum_{i\in\mathcal{G}_m}\zeta_{m,i}^2 \\
& + \frac{L}{2}\sum_{m=1}^{M}\frac{k_m\omega_m^2C^2\sigma_m^2}{r_m^2}.
\end{aligned}
$$

Let $\zeta_m^2 := \frac{1}{|\mathcal{G}_m|}\sum_{i\in\mathcal{G}_m}\zeta_{m,i}^2$, rearranging and summing it form $t=0$ to $t=T-1$ and dividing by T, one yields

$$
\begin{aligned}
\frac{1}{T}\sum_{t=0}^{T-1}\left\|\nabla f(\theta^t)\right\|^2 \leq & \frac{8(f(\theta^0)-f^*)}{\eta T\tau} + (4L\eta\tau + 4L\eta + 64L)\kappa^2 \\
& + \left(32L\eta\tau + L\eta + \frac{L\eta}{\tau}\right)\sum_{m=1}^{M}\omega_m(\phi_m+1)d\zeta_m^2 \\
& + \frac{4L}{\eta\tau}\sum_{m=1}^{M}\frac{k_m\omega_m^2C^2\sigma_m^2}{r_m^2}.
\end{aligned}
$$

which concludes the proof. □

E Sketch of Deriving Optimal Sparsification Levels

We start with simplifying the last two terms of the convergence bound of sparsification-amplified GDPFed.

$$
\begin{aligned}
&\left(32L\eta\tau + L\eta + \frac{L\eta}{\tau}\right)\sum_{m=1}^{M}\omega_m(\phi_m+1)d\zeta_m^2 + \frac{4L}{\eta\tau}\sum_{m=1}^{M}\frac{k_m\omega_m^2C^2\sigma_m^2}{r_m^2}\\
&\leq \left(32L\eta\tau + L\eta + \frac{L\eta}{\tau}\right)\sum_{m=1}^{M}\omega_m(\phi_m+1)dC^2 + \frac{4L}{\eta\tau}\sum_{m=1}^{M}\frac{k_m\omega_m^2C^2\sigma_m^2}{r_m^2}\\
&\Rightarrow \left(32\eta\tau + \eta + \frac{\eta}{\tau}\right)\sum_{m=1}^{M}\omega_m(\phi_m+1) + \frac{4}{\eta\tau}\sum_{m=1}^{M}\frac{\alpha_m\omega_m^2\sigma_m^2}{r_m^2}\\
&\Rightarrow \sum_{m=1}^{M}\left(\left(32\eta\tau + \eta + \frac{\eta}{\tau}\right)\omega_m(1+(1-\alpha_m)^2) + \frac{4}{\eta\tau}\frac{\alpha_m\omega_m^2\sigma_m^2}{r_m^2}\right) = f(k_1,k_2,\ldots,k_M),
\end{aligned}
$$

where the first inequality holds as $\zeta_m^2 \leq C^2$. Therefore, take the derivation of $f(k_1,k_2,\ldots,k_M)$ with respect to $k_1,k_2,\ldots,k_M$ and set the gradient to 0, we can easily get

$$
k_m^*/d = 1 - \frac{2\omega_m\sigma_m^2}{\eta\tau r_m^2\left(32\eta\tau + \eta + \frac{\eta}{\tau}\right)},
$$

which concludes the sketch.

F Formulation of Problem for Optimal Client Sampling Ratios

Note that in the following formulation process, we use the optimal sparsification level for each group as we derived in Remark 3. With the last two terms in the convergence bound shown in Theorem 3, we can simplify it as:

$$
\begin{aligned}
&\left(32L\eta\tau + L\eta + \frac{L\eta}{\tau}\right)\sum_{m=1}^{M}\omega_m(\phi_m^*+1)d\zeta_m^2 + \frac{4L}{\eta\tau}\sum_{m=1}^{M}\frac{k_m^*\omega_m^2C^2\sigma_m^2}{r_m^2}\\
&\leq \left(32L\eta\tau + L\eta + \frac{L\eta}{\tau}\right)\sum_{m=1}^{M}\omega_m(\phi_m^*+1)dC^2 + \frac{4L}{\eta\tau}\sum_{m=1}^{M}\frac{k_m^*\omega_m^2C^2\sigma_m^2}{r_m^2}\\
&\Rightarrow \left(32\eta\tau + \eta + \frac{\eta}{\tau}\right)\sum_{m=1}^{M}\omega_m(1+\phi_m^*) + \frac{4}{\eta\tau}\sum_{m=1}^{M}\frac{(1-\sqrt{\phi_m^*})\omega_m^2\sigma_m^2}{r_m^2}\\
&\Rightarrow \mu_4\sum_{m=1}^{M}\omega_m(1+\phi_m^*) + \mu_5\sum_{m=1}^{M}\frac{(1-\sqrt{\phi_m^*})\omega_m^2\sigma_m^2}{r_m^2}\\
&\Rightarrow \sum_{m=1}^{M}\omega_m\left(\mu_4(1+\phi_m^*) + \mu_5\frac{(1-\sqrt{\phi_m^*})\omega_m\sigma_m^2}{r_m^2}\right),
\end{aligned}
$$

where the first inequality holds as $\zeta_m^2 \leq C^2$. Consequently, we have

$$
\begin{aligned}
\min_{\{q_m\}_{m\in[M]}} \quad & \sum_{m=1}^{M}\omega_m\left(\mu_4(1+\phi_m^*) + \mu_5\frac{(1-\sqrt{\phi_m^*})\omega_m\sigma_m^2}{r_m^2}\right)\\
\text{s.t.} \quad & r_m = q_m|\mathcal{G}_m|,\ \sum_{m\in[M]} r_m = qn,
\end{aligned}
$$

which finished the formulation.