Subject Granular Differential Privacy in Federated Learning

Virendra J. Marathe
Oracle Labs
virendra.marathe@oracle.com

Pallika Kanani
Oracle Labs
pallika.kanani@oracle.com

Abstract

This paper introduces *subject* granular privacy in the Federated Learning (FL) setting, where a subject is an individual whose private information is embodied by several data items either confined within a single federation user or distributed across multiple federation users. We formally define the notion of *subject level differential privacy* for FL. We propose three new algorithms that enforce subject level DP. Two of these algorithms are based on notions of user level *local differential privacy (LDP)* and *group differential privacy* respectively. The third algorithm is based on a novel idea of *hierarchical gradient averaging (*HiGradAvgDP*)* for subjects participating in a training mini-batch. We also introduce *horizontal composition* of privacy loss for a subject across multiple federation users. We show that horizontal composition is equivalent to sequential composition in the worst case. We prove the subject level DP guarantee for all our algorithms and empirically analyze them using the FEMNIST and Shakespeare datasets. Our evaluation shows that, of our three algorithms, *HiGradAvgDP* delivers the best model performance, approaching that of a model trained using a DP-SGD based algorithm that provides a weaker *item level* privacy guarantee.

1 Introduction

Data privacy enforcement, using Differential Privacy (DP) (5; 6), in the Federated Learning (FL) setting (10) has been explored at two granularities: (i) *item level privacy*, where use of each data item in model training is obfuscated (1); and (ii) *user level privacy*, where participation of each federation user is hidden (13). User level privacy is generally understood to be a stronger privacy guarantee than item level privacy since the former hides use of *all* data of each user, whereas the latter may leak the user's data distribution even if it individually protects each data item (12; 13).

Item or user level privacy are perhaps the right privacy granularities in the original *cross-device* FL setting consisting of millions of hand held cell phones (2; 10) – an individual's data typically resides in just one cell phone that participates in a federation and one device typically only contains one individual's data. However, the *cross-silo* FL setting (8), where federation users are organizations that are themselves gatekeepers of data items of numerous individuals (which we call "subjects" henceforth), offer much richer mappings between subjects and their personal data.

Consider an online retail store customer C. C's online purchase history, which contains a multitude of orders placed in the past, is highly sensitive, and must be kept private. Furthermore, C may be a customer at other online retail stores. Thus C's aggregate private data may be distributed across several online retail stores. These retail stores could end up collaborating with each other in a federation to train a model using their customers', including C's, private purchase histories.

Item level privacy does not suffice to protect privacy of C's data. That is because item level privacy simply obfuscates participation of individual data items in the training process (1; 5; 6). Since a subject may have multiple data items in the dataset, item level private training may still leak a subject's data distribution (13; 12). User level privacy does not protect the privacy of C's data either.

Preprint. Under review.

User level privacy obfuscates each user's participation in training (13). However, a subject's data can be distributed among several users, and it can be leaked when aggregated through FL. In the worst case, multiple federation users may host only the data of a single subject. Thus C's data distribution can be leaked even if individual users' participation is obfuscated.

In this paper, we consider a third granularity of privacy – *subject level privacy* (18)[1], where a subject is an individual whose private data is spread across multiple data items, which can themselves be distributed across multiple federation users. The notion of subject level privacy is not new, and in fact appears in some of the original work on DP (5; 6). However, most existing work has either assumed a 1-to-1 mapping between subjects and data items (1), or has treated subjects as individual silos of data (a.k.a. users) in a collaborative learning setting such as FL (13). Recent work has addressed subject level privacy in a centralized setting (11), but no prior work has addressed the problem in a distributed collaborative learning setting such as FL. To the best of our knowledge, our work is the first study of subject level privacy in FL.

We formally characterize the notion of subject level privacy in terms of *subject level differential privacy*. We present three novel algorithms, called *UserLDP*, *LocalGroupDP*, and *HiGradAvgDP* respectively, that achieve subject level DP in the FL setting. Our algorithms assume a conservative trust model between the federation server and its users: The users do not trust the federation server (or other users) and enforce the subject level DP guarantee *locally*. All our algorithms are subtle enhancements of a FL training algorithm based on the *DP-SGD* algorithm by Abadi et al. (1) that provides the item level DP guarantee. We formally prove our algorithms' subject level DP guarantee.

We introduce the notion of *horizontal composition* for subject level privacy loss across multiple federation users. We show that, in the worst case, horizontal composition is equivalent to sequential composition, and as a result, adds additional constraints on the amount of training (number of training rounds in our algorithms) permitted under a given privacy loss budget compared to item level DP. These constraints can adversely affect model performance.

We empirically evaluate our algorithms on the FEMNIST and Shakespeare datasets. Our evaluation shows that while *UserLDP* and *LocalGroupDP* incur significant model utility overheads resulting in poor model performance, *HiGradAvgDP* delivers models whose performance approaches that of models trained using a baseline algorithm, called *LocalItemDP*, that provides a weaker *item level DP* guarantee in FL. Our evaluation further explores effects of (i) changing subject data distributions among federation users, and (ii) limiting federation data to a small number of subjects.

2 Subject Level Differential Privacy

We begin with the definition of Differential Privacy (5). Informally, DP bounds the maximum impact a single data item can have on the output of a randomized algorithm $\mathcal{A}$. Formally,

Definition 2.1. *A randomized algorithm $\mathcal{A} : \mathcal{D} \to \mathcal{R}$ is said to be (ε,δ)-differentially private if for any two* adjacent *datasets D, $D' \in \mathcal{D}$, and set $R \subseteq \mathcal{R}$,*

$$\mathcal{P}(\mathcal{A}(D) \in R) \leq e^{\varepsilon}\mathcal{P}(\mathcal{A}(D') \in R) + \delta \tag{1}$$

where D, D' are adjacent to each other if they differ from each other by a single data item. δ is the probability of failure to enforce the ε privacy loss bound.

The above definition provides item level privacy. McMahan et al. (13) present an alternate definition for user level DP in the FL setting. Let $\mathcal{U}$ be the set of n users participating in a federation, and $\mathcal{D}_i$ be the dataset of user $u_i \in \mathcal{U}$. Let $\mathcal{D}_{\mathcal{U}} = \bigcup_{i=1}^{n} \mathcal{D}_i$. Let $\mathcal{M}$ be the range of models resulting from the FL training process.

Definition 2.2. *Given a FL training algorithm $\mathcal{F} : \mathcal{D}_{\mathcal{U}} \to \mathcal{M}$, we say that $\mathcal{F}$ is* user level (ε, δ)-differentially private *if for any two adjacent user sets U, $U' \subseteq \mathcal{U}$, and $R \subseteq \mathcal{M}$,*

$$\mathcal{P}(\mathcal{F}(\mathcal{D}_U) \in R) \leq e^{\varepsilon}\mathcal{P}(\mathcal{F}(\mathcal{D}_{U'}) \in R) + \delta \tag{2}$$

where U, U' are adjacent user sets differing by a single user.

Let $\mathcal{S}$ be the set of subjects whose data is hosted by the federation's users $\mathcal{U}$. Our definition of subject level DP is based on the observation that, even though the data of individual subjects $s \in \mathcal{S}$ may be physically scattered across multiple users in $\mathcal{U}$, the aggregate data across $\mathcal{U}$ can be logically divided into its subjects in $\mathcal{S}$ (i.e. $\mathcal{D}_{\mathcal{U}} = \bigcup_{s \in \mathcal{S}} \mathcal{D}_s$).

[1]Wang et al. (18) identify what we call subjects in this paper as users in their paper.

Definition 2.3. *Given a FL training algorithm $\mathcal{F} : \mathcal{D}_\mathcal{U} \rightarrow \mathcal{M}$, we say that $\mathcal{F}$ is* subject level (ε, δ)-differentially private *if for any two adjacent subject sets S, $S' \subseteq \mathcal{S}$, and $R \subseteq \mathcal{M}$,*

$$\mathcal{P}(\mathcal{F}(\mathcal{D}_S) \in R) \leq e^\varepsilon \mathcal{P}(\mathcal{F}(\mathcal{D}_{S'}) \in R) + \delta \tag{3}$$

where S and S' are adjacent subject sets if they differ from each other by a single subject.

Note that our definition completely ignores the notion of users in a federation. This user obliviousness is crucial to make the definition work for both cases: (i) where a subject's data items are confined to a single user (e.g. for cross-device FL settings), and (ii) where a subject's data items are spread across multiple users (e.g. for cross-silo FL settings) (18).

3 Enforcing Subject Level Differential Privacy

We assume a federation that contains a federation server and its users. The server is responsible for (i) initialization and distribution of the model architecture to the federation users, (ii) coordination of training rounds, (iii) aggregation and application of model updates coming from different users in each training round, and (iv) redistribution of the updated model back to the users. Each user (i) receives updated models from the federation server, (ii) retrains the received models using its private training data, and (iii) returns updated model parameters to the federation server.

We assume that the federation users and the server behave as *honest-but-curious* participants in the federation: They do not interfere with or manipulate the distributed training process in any way, but may be interested in analyzing received model updates. Federation users do not trust each other or the federation server, and must locally enforce privacy guarantees for their private data.

Our algorithms enforce subject level DP *locally* at each user. But to prove the privacy guarantee for any subject, across the entire federation, we must ensure that the local subject level DP guarantee *composes* correctly through the *global* aggregation, at the federation server, of parameter updates received from these users. To that end we break down the federated training round into two functions: (i) $\mathcal{F}_l$, the user's training algorithm that enforces subject level DP locally, and (ii) $\mathcal{F}_g$, the server's operation that aggregates parameter updates received from all the users. We first present our three algorithms for $\mathcal{F}_l$ and show that they locally enforce subject level DP. Thereafter we show how an instance of $\mathcal{F}_g$ that simply averages parameter updates (at the federation server) composes the subject level DP guarantee across multiple users in the federation.

Our algorithms are based on a federated version of the *DP-SGD* algorithm by Abadi et al. (1). *DP-SGD* was originially not designed for FL, but can be easily extended to enforce item level DP in FL: The federation server samples a random set of users for each training round and sends them a request to perform local training. Each user trains the model locally using *DP-SGD*. Formally, the parameter update at step t in *DP-SGD* can be summarized in the following equation:

$$\Theta_t = \Theta_{t-1} + \eta(\nabla\mathcal{L}^C(\Theta_{t-1}) + \mathcal{N}(0, C^2\sigma^2)) \tag{4}$$

where, $\nabla\mathcal{L}^C$ is the loss function's gradient clipped by the threshold C, σ is the noise scale calculated using the moments accountant method, $\mathcal{N}$ is the Gaussian distribution used to calculate noise, and η is the learning rate. Note that, in a mini-batch, the gradient for each data item is computed and clipped separately to limit the influence (*sensitivity*) of each data item on the loss function's gradient.

The users ship back updated model parameters to the federation server, which averages the updates received from all the sampled users. The server redistributes the updated model and triggers another training round if needed. The original paper (1) also proposed the moments accountant method for tighter composition of privacy loss bounds compared to prior work on strong composition (7).

3.1 User Level Local Differential Privacy

In general, user level privacy (13) does not guarantee subject level privacy. However, we observe that a stronger privacy guarantee, called *Local Differential Privacy (LDP)* (4; 9; 19), enforced at user granularity, is sufficient to guarantee subject level privacy. There are strong parallels between the traditional LDP setting, where a data analyst can get access to the data only after it has been perturbed, and privacy in the FL setting, where the federation server gets access to parameter updates from users after they have been locally perturbed by the users. In fact, in the FL setting (17), LDP is a much stronger privacy guarantee than item level or user level DP in that it obfuscates the entire signal from a user to the extent that an adversary, even the federation server, cannot tell the difference between the signals coming from any two different users.

Algorithm 1: Pseudo code for *UserLDP*.

Parameters: Set of n users $\mathcal{U} = u_i, u_2, ..., u_n$; $\mathcal{D}_i$, the dataset of user u_i; M, the model to be trained; θ, the parameters of model M ; noise scale σ; gradient norm bound C; mini-batch size B; R training rounds; the learning rate η.

```
1  UserLDP($u_i$):
2  for $t = 1$ to $T$ do
3      $S$ = random sample of $B$ data items from $\mathcal{D}_i$
4      Compute gradients:
5      $g(S) = \nabla\mathcal{L}(\theta, S)$
6      Clip gradients:
7      $\bar{g}(S) = g(S)/max(1, \frac{\|g(S)\|_2}{C})$
8      Add Gaussian noise:
9      $\tilde{g}(S) = \bar{g}(S) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})$
10     $\theta = \theta - \eta\tilde{g}(S)$
11 return $\theta$

12 Server Loop:
13 for $r = 1$ to $R$ do
14     $U_s$ = sample $s$ users
15         from $\mathcal{U}$
16     for $u_i \in U_s$ do
17         $\theta_i$ = UserDPSGD($u_i$)
18     $\theta = \frac{1}{s}\sum_{i=1}^{s}\theta_i$
19     Send $M$ to all users
20         in $\mathcal{U}$
```

Definition 3.1. *We say that FL algorithm $\mathcal{F} : \mathcal{D}_\mathcal{U} \rightarrow \mathcal{M}$ is* user level (ε,δ)-local differentially private*, where $\mathcal{D}_\mathcal{U}$ is the aggregate dataset over users in set $\mathcal{U}$, and $\mathcal{M}$ is the range of parameters of the model getting trained, if for any two users u_1, u_2, and $S \subseteq \mathcal{M}$,*

$$\mathcal{P}(\mathcal{F}(D_{u_1}) \in S) \leq e^{\varepsilon}\mathcal{P}(\mathcal{F}(D_{u_2}) \in S) + \delta \tag{5}$$

where D_{u_1} and D_{u_2} are the datasets of users u_1 and u_2 respectively.

User level LDP is a stronger privacy guarantee than subject level DP. More formally,

Theorem 3.1. *User level (ε,δ)-local differential privacy entails (ε, δ)-subject level differential privacy.*

The formal proof for Theorem 3.1 appears in the appendix.

We now present a new user level (ε,δ)-LDP algorithm called *UserLDP*. The underlying intuition behind this algorithm is to let the user locally inject enough noise to make its entire signal *indistinguishable* from any other user's signal. In every training round, each federation user enforces user level LDP independently of the federation and any other users in the federation. The federation server simply averages parameter updates received from users and braodcasts the new averaged parameters back to the users.

UserLDP's pseudo code appears in Algorithm 1. *UserLDP* appears significantly similar to *DP-SGD*. The key difference is that while *DP-SGD* computes noise proportional to the gradient contribution of any single data item in a mini-batch, *UserLDP* computes noise proportional to the gradient contribution of the entire mini-batch, thus obfuscating the entire signal from the mini-batch. To guarantee DP, we need to first cap the sensitivity of each user u_i's contribution to parameter updates. To that end, we focus on change affected by any mini-batch b trained at u_i.

Lemma 3.2. *For every mini-batch b of a sampled user u_i's training round in* UserLDP*, the sensitivity $\mathbb{S}_b$ of the computed parameter gradient is bounded by C; i.e. $\mathbb{S}_b \leq |C|$.*

Theorem 3.3. UserLDP *enforces user-level local (ε,δ)-differential privacy for each mini-batch b.*

The proofs for above lemma and theorem appear in Appendix A.

Applying standard DP composition results (1; 7) to Theorem 3.3, and combining it with Theorem 3.1 locally proves subject level DP guarantee for *UserLDP* over individual training rounds.

3.2 Locally Enforced Group Level Differential Privacy

While user level LDP is a stronger guarantee than subject level privacy, it is known to induce excessive noise in the training process, leading to significant utility degradation in the trained model (4; 9). Intuitively, user level LDP is guaranteeing privacy at a coarser granularity (user level) as opposed to granularity of individual data subjects. As a result, we need to find better alternatives that more precisely calibrate noise proportional to a data subject's influence on training. A direct method to attain that is by obfuscating the effects of the *group* of data items belonging to the same subject. We can apply formalism of *group differential privacy* (6) to achieve this group-level obfuscation. Formally (from (6)),

Theorem 3.4. *Any (ε, δ)-differentially private randomized algorithm $\mathcal{A}$ is ($g\varepsilon, ge^{(g-1)\varepsilon}\delta$)-differentially private for groups of size g. That is, given two g-adjacent datasets D and D', and $R \in \mathcal{M}$, where $\mathcal{M}$ is the output space domain,*

$$\mathcal{P}(\mathcal{A}(D) \in R) \leq e^{g\varepsilon}\mathcal{P}(\mathcal{A}(D') \in R) + ge^{(g-1)\varepsilon}\delta \tag{6}$$

Algorithm 2: Pseudo code for *LocalGroupDP* that guarantees *subject level* DP via group DP enforcement.

Parameters: Set of n users $\mathcal{U} = u_i, u_2, ..., u_n$; $\mathcal{D}_i$, the dataset of user u_i; M, the model to be trained; θ, the parameters of model M ; gradient norm bound C; sample of users U_s; mini-batch size B; Z, largest group size in a mini-batch, σ_Z, noise scale for group of size Z; R training rounds; T batches per round; the learning rate η.

```
1  LocalGroupDP($u_i$):
2      for $t = 1$ to $T$ do
3          S = random sample of $B$ data items from $\mathcal{D}_i$
4          for $s_i \in S$ do
5              Compute gradients:
6              $g(s_i) = \nabla\mathcal{L}(\theta, s_i)$
7              Clip gradients:
8              $\bar{g}(s_i) = Clip(g(s_i), C)$
9          $Z = LrgGrpCnt(S)$
10         $\tilde{g}_s = \frac{1}{B}(\sum_i \bar{g}(s_i) + \mathcal{N}(0, \sigma_Z^2 C^2 \mathbf{I}))$
11         $\theta = \theta - \eta\tilde{g}_s$
12     return $M$

13 Server Loop:
14     for $r = 1$ to $R$ do
15         $U_s$ = sample $s$ users from $\mathcal{U}$
16         for $u_i \in U_s$ do
17             $\theta_i = LocalGroupDP(u_i)$
18         $\theta = \frac{1}{s}\sum_i \theta_i$
19         Send $M$ to all users in $\mathcal{U}$
```

where D and D' are g-adjacent if they differ from each other in g data items.

Clearly, group DP incurs a big linear penalty on the privacy loss ε, and an even bigger penalty in the failure probability ($ge^{(g-1)\varepsilon}\delta$). Nevertheless, if g is restricted to a small value (e.g. 2) the group DP penalty may be acceptable.

Theorem 3.4 is a bi-directional implication. So it can be restated as follows:

Corollary 3.5. *Any ($\mathcal{E}, \Delta$)-group differentially private algorithm $\mathcal{A}$, for a group size of g, is ($\mathcal{E}/g, \Delta/(ge^{(g-1)\frac{\mathcal{E}}{g}}$)-differentially private.*

In the FL setting, subject level DP immediately follows from group DP for every sampled mini-batch of data items at every federation user. Let S be a sampled mini-batch of data items at a user u_i, and $\mathcal{M}$ be the domain space of the ML model being trained in the FL setting.

Theorem 3.6. *Let training algorithm $\mathcal{A}_g : S \rightarrow \mathcal{M}$ be group differentially private for groups of size g, and l be the largest number of data items belonging to any single subject in S. If $l \leq g$, then $\mathcal{A}_g$ is subject level differentially private.*

Composition of group DP guarantees over multiple mini-batches and training rounds also follows established DP composition results (1; 7; 15). For instance, the moments accountant method by Abadi et al. (1) shows that given an (ε,δ)-DP gradient computation for a single mini-batch, the full training algorithm, which consists of T mini-batches and a mini-batch sampling fraction of q, is $(O(q\varepsilon\sqrt{T}), \delta)$-differentially private. Theorem 3.4 implies that the same algorithm is $(O(gq\varepsilon\sqrt{T}, ge^{(g-1)\varepsilon}\delta)$-group differentially private for a group of size g.

We now present our new FL training algorithm, *LocalGroupDP*, that guarantees group DP. We make a critical assumption in *LocalGroupDP*: Each user can determine the subject for any of its data items. Absent this assumption, the user may need to make the worst case assumption that all data items used to train the model belong to the same subject. On the other hand, these algorithms are *strictly local*, and do not require that the identity of the subjects be resolved across users.

LocalGroupDP (Algorithm 2) enforces subject level privacy locally at each user. Like prior work (1; 13; 16), we enforce DP in *LocalGroupDP* by adding carefully calibrated Gaussian noise in each mini-batch's gradients. Each user clips gradients for each data item in a mini-batch to a clipping threshold C prescribed by the federation server. The clipped gradients are subsequently averaged over the mini-batch. The clipping step bounds the *sensitivity* of each mini-batch's gradients to C.

To enforce group DP, *LocalGroupDP* also locally tracks the item count of the subject with the largest number of items in the sampled mini-batch. This count determines the *group size* needed to enforce group DP for that mini-batch. This group size, Z in Algorithm 2, helps determine the noise scale σ_Z, given the target privacy parameters $(\mathcal{E}, \Delta)$ over the entire training computation. More specifically, we use the moments accountant method and Corollary 3.5 to calculate σ for $\varepsilon = \mathcal{E}/Z$, and $\delta = \Delta/(Ze^{(Z-1)\frac{\mathcal{E}}{Z}})$. Note that the value of Z can vary between mini-batches, due to which we represent the noise scale as σ_Z in the pseudo code. σ_Z is computed using the moments accountant method. The rest of the parameters to calculate σ_Z – $\mathcal{E}$, Δ, total number of mini-batches ($T.R$), and sampling fraction ($B/total\ dataset\ size$) – remain the same throughout the training process.

Algorithm 3: Pseudo code for *HiGradAvgDP* that guarantees *subject level* DP via hierarchical gradient averaging.

Parameters: Set of n users $\mathcal{U} = u_i, u_2, ..., u_n$; $\mathcal{D}_i$, the dataset of user u_i; M, the model to be trained; θ, the parameters of model M ; gradient norm bound C; noise scale σ; sample of users U_s; mini-batch size B; R training rounds; T batches per round; η the learning rate; $\mathcal{S}_a^S$ the subset of data items from set S that have a as their subject.

```
1  HiGradAvgDP($u_i$):
2      for $t = 1$ to $T$ do
3          $S$ = random sample of $B$ data items from $\mathcal{D}_i$
4          for $a \in subjects(S)$ do
5              for $s_i \in \mathcal{S}_a^S$ do
6                  Compute gradients:
7                  $g(s_i) = \nabla\mathcal{L}(\theta, s_i)$
8                  Clip gradients:
9                  $\bar{g}(s_i) = Clip(g(s_i), C)$
10             Average subject $a$'s gradients:
11             $g(\mathcal{S}_a^S) = \frac{1}{|\mathcal{S}_a^S|}(\sum_i \bar{g}(s_i))$
12         $\tilde{g}_S = \frac{1}{B}(\sum_{a\in subjects(S)} g(\mathcal{S}_a^S) + \mathcal{N}(0, \sigma^2C^2\mathbf{I}))$
13         $\theta = \theta - \eta\tilde{g}_S$
14     return $M$
15 Server Loop:
16     for $r = 1$ to $R$ do
17         $U_s$ = sample $s$ users from $\mathcal{U}$
18         for $u_i \in U_s$ do
19             $\theta_i = HiGradAvgDP(u_i)$
20         $\theta = \frac{1}{s}\sum_i \theta_i$
21         Send $M$ to all users in $\mathcal{U}$
```

LocalGroupDP enforces $(\mathcal{E}/Z, \Delta/(Ze^{(Z-1)\frac{\mathcal{E}}{Z}})$-differential privacy, which by Corollary 3.5 implies $(\mathcal{E}$,$\Delta)$-group differential privacy, hence subject level DP by Theorem 3.6.

3.3 Hierarchical Gradient Averaging

While *LocalGroupDP* may seem like an attractive alternative to *UserLDP*, the former's utility penalty due to group DP can be significant. For instance, even a group of size 2 effectively *halves* the available privacy budget $\mathcal{E}$ for training. The key challenge to enforce subject level DP is that the following constraint seems fundamental: *To guarantee subject level DP, any training algorithm must obfuscate the entire contribution made by any subject in the model's parameter updates. LocalGroupDP* complies with this constraint by enforcing group DP.

Our new algorithm, called *HiGradAvgDP* (Algorithm 3), takes a diametrically opposite view to comply with the same constraint: Instead of scaling the noise to a subject's group size (as is done in *LocalGroupDP*), *HiGradAvgDP scales down* each subject's mini-batch gradient contribution to the clipping threshold C. This is done in three steps: (i) collect data items belonging to a common subject in the sampled mini-batch, (ii) compute and clip gradients using the threshold C for each individual data item of the subject, and (iii) average those clipped gradients for the subject, denoted by $g(\mathcal{S}_a^S)$. Clipping and then averaging gradients ensures that the entire subject's gradient norm is bounded by C. Subsequently, *HiGradAvgDP* sums all the per-subject averaged gradients along with the Gaussian noise, which are then averaged over the mini-batch size B. *HiGradAvgDP* gets its name from this average-of-averages step.

The Gaussian noise scale σ is calculated independently at each user u_i using standard parameters – the privacy budget ε, the failure probability δ, total number of mini-batches $T.R$, and the sampling fraction per mini-batch $\frac{B}{|D_i|}$. The calculation uses the moments accountant method to compute σ.

To formally prove that *HiGradAvgDP* enforces subject level DP, we first provide a formal definition of *subject sensitivity* in a sampled mini-batch.

Definition 3.2 (Subject Sensitivity)**.** *Given a model $\mathcal{M}$, and a sampled mini-batch S of training data, we define* subject sensitivity $\mathbb{S}^S$ *for S as the maximum difference caused by any single subject $a \in subjects(S)$ in $\mathcal{M}$'s parameter gradients computed over S.*

Lemma 3.7. *For every sampled mini-batch S in a sampled user u_i's training round in* HiGradAvgDP*, the subject sensitivity $\mathbb{S}^S$ for S is bounded by C; i.e. $\mathbb{S}^S \leq |C|$.*

Theorem 3.8. HiGradAvgDP *locally enforces subject-level (ε,δ)-differential privacy.*

Proofs for Lemma 3.7 and Theorem 3.8 appear in the appendix.

3.4 Composition Over Multiple Training Rounds

Composition of privacy loss across multiple training rounds can be done by straightforward application of DP composition results, such as the moments accountant method that we use in our work. Thus the privacy loss ε_r incurred in any single training round r amplifies by a factor of $\sqrt{R}$ when federated training runs for R rounds. We note that privacy losses are incurred by federation users independently of other federation users. Foreknowledge of the number of training rounds R lets us calculate the Gaussian noise distribution's standard deviation σ for a privacy loss budget of (ε,δ) for the aggregate training over R rounds. Given an aggregate privacy loss budget of ε, since all users train for an identical number of rounds R, they incur a privacy loss of $\varepsilon_r = \frac{\varepsilon}{\sqrt{R}}$ in each training round r. Notably, this privacy loss per training round is the the same for all users even if their dataset cardinalities are dramatically different.

3.5 Composing Subject Level DP Across Federation Users

At the beginning of a training round r, each sampled user receives a copy of the global model, with parameters Θ_{r-1}, which it then retrains using its private data. Since all sampled users start retraining from the same model $\mathcal{M}_{\Theta_{r-1}}$, and independently retrain the model using their respective private data, parallel composition of privacy loss across these sampled users may seem to apply naturally (14). In that case, the aggregate privacy loss incurred across multiple federation users, via an aggregation such as federated averaging, remains identical to the privacy loss ε_r incurred individually at each user. However, parallel composition was proposed for item level privacy, where an item belongs to at most one participant. With subject level privacy, a subject's data items can span across multiple users, which limits application of parallel privacy loss composition to only those federations where each subject's data is restricted to at most one federation user. In the more general case, we show that subject level privacy loss composes *sequentially* via the federated averaging aggregation algorithm used in our FL training algorithms.

Formally, consider a FL training algorithm $\mathcal{F} = (\mathcal{F}_l, \mathcal{F}_g)$, where $\mathcal{F}_l$ is the user local component, and $\mathcal{F}_g$ the global aggregation component of $\mathcal{F}$. Given a federation user u_i, let $\mathcal{F}_l : (\mathcal{M}, D_{u_i}) \rightarrow P_{u_i}$, where $\mathcal{M}$ is a model, D_{u_i} is the private dataset of user u_i, and P_{u_i} is the updated parameters produced by $\mathcal{F}_l$. Let $\mathcal{F}_g = \frac{1}{n}\sum_i P_{u_i}$, a parameter update averaging algorithm over a set of n federation users u_i.

Theorem 3.9. *Given a FL training algorithm $\mathcal{F} = (\mathcal{F}_l, \mathcal{F}_g)$, in the most general case where a subject's data resides in the private datasets of multiple federation users u_i, the aggregation algorithm $\mathcal{F}_g$ sequentially composes subject level privacy losses incurred by $\mathcal{F}_l$ at each federation user.*

We term this sequential composition of privacy loss across federation users as *horizontal composition*. Horizontal composition has a significant effect on the number of federated training rounds permitted under a given privacy loss budget.

Theorem 3.10. *Consider a FL training algorithm $\mathcal{F} = (\mathcal{F}_l, \mathcal{F}_g)$ that samples s users per training round, and trains the model $\mathcal{M}$ for R rounds. Let $\mathcal{F}_l$ at each participating user, over the aggregate of R training rounds, locally enforce subject-level (ε,δ)-DP. Then $\mathcal{F}$ globally enforces the same subject-level (ε,δ)-DP guarantee by training for $\frac{R}{\sqrt{s}}$ rounds.*

The main intuition behind Theorem 3.10 is that the s-way horizontal composition via $\mathcal{F}_g$ results in an increase in training mini-batches by a factor of s. As a result, the privacy loss calculated by the moments accountant method amplifies by a factor of $\sqrt{s}$, thereby forcing a reduction in number of training rounds by a factor of $\sqrt{s}$ to counteract the privacy loss amplification. This reduction in training rounds can have a significant impact on the resulting model's performance, as we demonstrate in section 4. Proofs for Theorem 3.9 and Theorem 3.10 appear in the appendix.

4 Empirical Evaluation

We implemented all our algorithms (*UserLDP*, *LocalGroupDP*, and *HiGradAvgDP*), and a version of the DP-SGD algorithm by Abadi et al. (1) that enforces item level DP in the FL setting (*LocalItemDP*). We also compare these algorithms with a FL training algorithm, *FedAvg* (10), that does not enforce any privacy guarantees. All our algorithms are implemented in our distributed FL framework built on distributed PyTorch.

We focus our evaluation on Cross-Silo FL (8), which we believe is the most appropriate setting for the subject level privacy problem. We use the FEMNIST and Shakespeare datasets (3) for our evaluation. In FEMNIST, the hand-written numbers and letters can be divided based on authors, which ordinarily serve as federation users in FL experiments by most researchers. In Shakespeare, each character in the Shakespeare plays serves as a federation user. In our experiments however, the FEMNIST authors and Shakespeare play characters are treated as data subjects. To emulate the cross-silo FL setting, we report evaluation on a 16-user federation.

We use the CNN model on FEMNIST appearing in the LEAF benchmark suite (3) as our target model to train. More specifically, the model consists of two convolution layers interleaved with

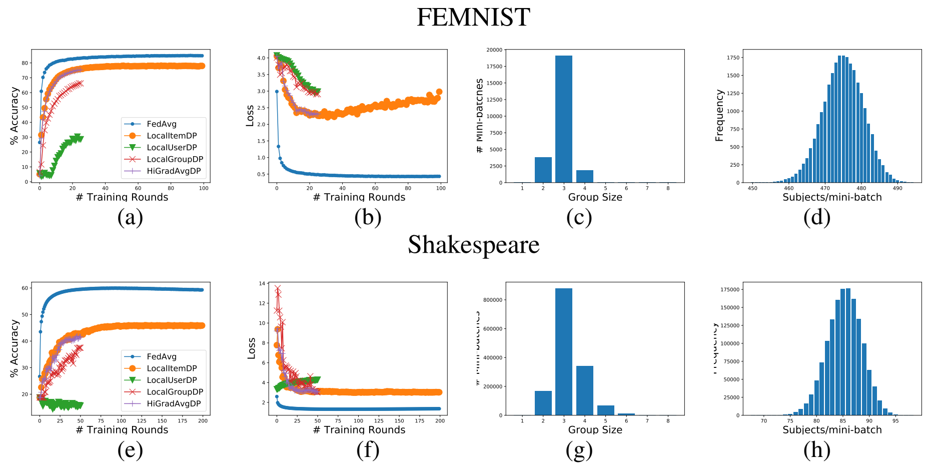


Figure 1: Average test accuracy and loss on the FEMNIST (a),(b) and Shakespeare (e),(f) datasets over training rounds for various algorithms. For DP guarantees: $\epsilon = 4.0$ and $\delta = 10^{-5}$ budgeted over all 100 and 200 training rounds for FEMNIST and Shakespeare respectively. Model performance for the subject level privacy algorithms is constrained by the limited number of training rounds (25 for FEMNIST, and 50 for Shakespeare) permitted under the prescribed privacy budget. Number of mini-batches with subject group sizes over the entire training run for FEMNIST (c) and Shakespeare (g). Number of mini-batches with distinct subjects per mini-batch for FEMNIST (d) and Shakespeare (h).

ReLU activations and maxpooling, followed by two fully connected layers before a final log softmax layer. For the Shakespeare dataset we use a stacked LSTM model with two linear layers at the end.

We use 80% of the training data for training, and 20% for validation. Test data comes separately in FEMNIST and Shakespeare. Training and testing was done on a local GPU cluster comprising 2 nodes, each containing 8 Nvidia Tesla V100 GPUs.

We extensively tuned the hyperparameters of mini-batch size B, number of training rounds T, gradient clipping threshold C, and learning rate η. The final hyperparameters for FEMNIST were: $B = 512$, $T = 100$, $C = 0.001$, and learning rates η of 0.001 and 0.01 for the non-private and private FL algorithms respectively. Shakespeare hyperparameters were: $B = 100$, $T = 200$, $C = 0.00001$, and learning rates η of 0.0002 and 0.01 for the non-private and private FL algorithms.

4.1 FEMNIST and Shakespeare Performance

We first conduct an experiment that reports average test accuracy and loss at the end of each training round, over a total of 100 and 200 training rounds for FEMNIST and Shakespeare respectively. The FEMNIST dataset contains 3500 subjects, and the Shakespeare dataset contains 660 subjects. In both datasets each subject comprises hundreds of data items. Each subject's data items are uniformly distributed among the 16 federation users.

Figure 1 shows performance of the models trained using our algorithms. *FedAvg* performs the best since it does not incur any DP enforcement penalties. Item level privacy enforcement in *LocalItemDP* results in performance degradation of 8% for FEMNIST and 22% for Shakespeare. The utility cost of user level LDP in *UserLDP* is quite clear from the figure. This cost is also reflected in the relatively high observed loss for the respective model. *LocalGroupDP* performs significantly better than *UserLDP*, but worse than *LocalItemDP*, by 15% on FEMNIST, and 18% on Shakespeare. The reason for *LocalGroupDP*'s worse performance is clear from Figure 1(c) and (g): the group size for a mini-batch tends to be dominated by 3 on both FEMNIST and Shakespeare, which cuts the privacy budget for these mini-batches by a factor of 3, leading to greater Gaussian noise, which in turn leads to model performance degradation.

HiGradAvgDP performs competitively with *LocalItemDP* for the 25 and 50 rounds it is trained for on FEMNIST and Shakespeare respectively. Figure 1 (d) and (h) show that instances of sampling multiple data items corresponding to the same subject in a single mini-batch are relatively low – the number of distinct subjects sampled per mini-batch of 512 for FEMNIST averages to 475, and per mini-batch of 100 for Shakespeare averages to 86. As a result, *HiGradAvgDP* incurs insignificant performance degradation for both datasets. However, the training round restriction does result in degradation of the final model produced by *HiGradAvgDP*compared to *LocalItemDP*: For FEMNIST, *HiGradAvgDP* gives 75.24% prediction accuracy after 25 rounds compared to 77.96%

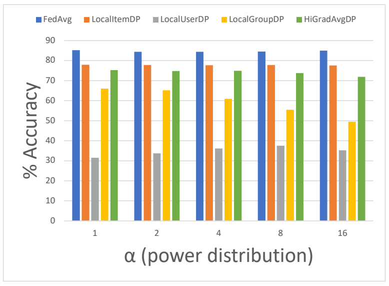
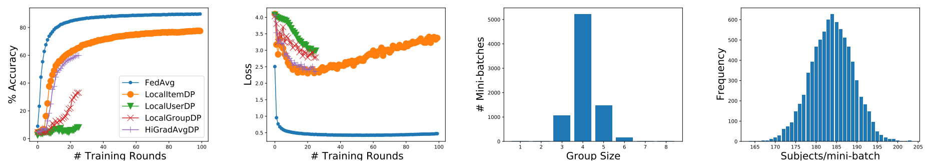


Figure 2: Model performance over FEMNIST dataset of our algorithms over different subject data distributions dictated by the parameter α of the power distribution.



(a) (b) (c) (d)

Figure 3: Average test accuracy (a) and loss (b) measured over training rounds for various algorithms on the Small-FEMNIST dataset (350 subjects) distributed among 16 federation users. For DP guarantees: $\epsilon = 4.0$ and $\delta = 10^{-5}$. Observed grouping sizes per mini-batch in *LocalGroupDP* aggregated across all mini-batches at all federation users in the training run (c), and number of subjects observed per mini-batch in *HiGradAvgDP* aggregated across all mini-batches at all federation users in the training run (d).

accuracy after 100 rounds with *LocalItemDP*. For Shakespeare, *HiGradAvgDP* gives 41.58% model accuracy after 50 rounds compared to 45.91% accuracy with *LocalItemDP* after 200 rounds.

4.2 Effect of Subject Data Distribution

While evaluation of our algorithms using a uniform distribution of subject data among federation users is a good starting point, often times the data distribution is non-uniform in real world settings. To emulate varying subject data distributions, we conduct experiments on the FEMNIST dataset where subject data is distributed among federation users according to the power distribution

$$P(x; \alpha) = \alpha x^{\alpha-1}, 0 \leq x \leq 1, \alpha > 0$$

Figure 2 shows performance of the models trained using our algorithms over varying subject data distributions of FEMNIST. As expected, different data distributions clearly do not significantly affect *FedAvg*, *LocalItemDP*, and *User-Local-SGD*. However, performance of the model trained using *LocalGroupDP* degrades substantially as the unevenness of data distribution increases, resulting in test accuracy under 50% for $\alpha = 16$. This degradation is singularly attributable to growth in subject group size per mini-batch – the average group size per mini-batch ranges from 3 when $\alpha = 2$ to 6 when $\alpha = 16$. This increase in group size significantly reduces the privacy budget leading to increase in Gaussian noise that restricts test accuracy. On the other hand, *HiGradAvgDP* appears to be much more resilient to non-uniform subject data distributions among federation users – test accuracy drops by just about 5% from $\alpha = 1$ (75.84% accuracy) to $\alpha = 16$ (71.89% accuracy). The corresponding subjects per minibatch observed in our experiments goes from an average of 475 to 310 respectively (not reported in detail due to space constraints).

4.3 Small-FEMNIST Performance

HiGradAvgDP appears to generally perform well when the number of subjects in the federation is sufficiently large. To study the effects of fewer subjects in a federation we experimented with a trimmed down version of FEMNIST, called Small-FEMNIST, that contains just the first 350 subjects in the aggregate dataset. For these experiments we reduced mini-batch size to 256. Figure 3 shows the performance of our algorithms on Small-FEMNIST. We note a substantial drop in the performance of *LocalGroupDP*, which can be explained by the increase in subject group size (to an average of 4) as can be seen in Figure 3 (c). There is also a noticable drop in performance of *HiGradAvgDP*, which is attributable to a decrease in distinct subjects occuring per mini-batch (from Figure 3 (d)). This drop in number of subjects adversely affects performance of models trained using the algorithms that enforce subject level privacy.

5 Conclusion

While various prior works on privacy in FL have explored DP guarantees at the user and item levels (12; 13), to the best of our knowledge, no prior work has studied subject level granularity for privacy in the FL setting. In this paper, we presented a formal definition of subject level DP. We also presented three novel FL training algorithms that guarantee subject level DP by either enforcing user level LDP (*UserLDP*), local group DP (*LocalGroupDP*), or by applying hierarchical gradient averaging to obfuscate a subject's contribution to mini-batch gradients (*HiGradAvgDP*). Our empirical evaluation on the FEMNIST dataset suggests that while both *UserLDP* and *LocalGroupDP* can significantly degrade model performance, *HiGradAvgDP* tends to incur little loss in performance compared to *LocalItemDP*, an algorithm that provides a weaker item level privacy guarantee. We observe an interesting new aspect of *horizontal composition* of privacy loss for subject level privacy in FL that results in model performance degradation, which we intend to research in 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.

[2] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloé Kiddon, Jakub Konecný, Stefano Mazzocchi, H. Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. Towards federated learning at scale: System design. *CoRR*, abs/1902.01046, 2019.

[3] Sebastian Caldas, Peter Wu, Tian Li, Jakub Konečný, H. Brendan McMahan, Virginia Smith, and Ameet Talwalkar. LEAF: A benchmark for federated settings. *CoRR*, abs/1812.01097, 2018.

[4] John C. Duchi, Michael I. Jordan, and Martin J. Wainwright. Local privacy and statistical minimax rates. *CoRR*, abs/1302.3203, 2013.

[5] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the Third Conference on Theory of Cryptography*, TCC'06, pages 265–284, 2006.

[6] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3&#8211;4):211–407, August 2014.

[7] Cynthia Dwork, Guy N. Rothblum, and Salil P. Vadhan. Boosting and differential privacy. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS*, pages 51–60, 2010.

[8] Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaïd Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konecný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Mariana Raykova, Hang Qi, Daniel Ramage, Ramesh Raskar, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. Advances and open problems in federated learning. *CoRR*, abs/1912.04977, 2019.

[9] Shiva Prasad Kasiviswanathan, Homin K. Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam D. Smith. What can we learn privately? *CoRR*, abs/0803.0924, 2008.

[10] Jakub Konecný, Brendan McMahan, and Daniel Ramage. Federated optimization: Distributed optimization beyond the datacenter. *CoRR*, abs/1511.03575, 2015.

[11] Daniel Levy, Ziteng Sun, Kareem Amin, Satyen Kale, Alex Kulesza, Mehryar Mohri, and Ananda Theertha Suresh. Learning with user-level privacy. *CoRR*, abs/2102.11845, 2021.

[12] Yuhan Liu, Ananda Theertha Suresh, Felix X. Yu, Sanjiv Kumar, and Michael Riley. Learning discrete distributions: user vs item-level privacy. *CoRR*, abs/2007.13660, 2020.

[13] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *6th International Conference on Learning Representations, ICLR 2018*, 2018.

[14] Frank McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 19–30, 2009.

[15] Ilya Mironov. Renyi differential privacy. *CoRR*, abs/1702.07476, 2017.

[16] Shuang Song, Kamalika Chaudhuri, and Anand D. Sarwate. Stochastic gradient descent with differentially private updates. In *IEEE Global Conference on Signal and Information Processing*, pages 245–248, 2013.

[17] Stacey Truex, Ling Liu, Ka Ho Chow, Mehmet Emre Gursoy, and Wenqi Wei. Ldp-fed: Federated learning with local differential privacy. In *Proceedings of the 3rd International Workshop on Edge Systems, Analytics and Networking, EdgeSys@EuroSys 2020, Heraklion, Greece, April 27, 2020*, pages 61–66. ACM, 2020.

[18] Jianyu Wang, Zachary Charles, Zheng Xu, Gauri Joshi, H. Brendan McMahan, Blaise Aguera y Arcas, Maruan Al-Shedivat, Galen Andrew, Salman Avestimehr, Katharine Daly, Deepesh Data, Suhas Diggavi, Hubert Eichner, Advait Gadhikar, Zachary Garrett, Antonious M. Girgis, Filip Hanzely, Andrew Hard, Chaoyang He, Samuel Horvath, Zhouyuan Huo, Alex Ingerman, Martin Jaggi, Tara Javidi, Peter Kairouz, Satyen Kale, Sai Praneeth Karimireddy, Jakub Konecny, Sanmi Koyejo, Tian Li, Luyang Liu, Mehryar Mohri, Hang Qi, Sashank J. Reddi, Peter Richtarik, Karan Singhal, Virginia Smith, Mahdi Soltanolkotabi, Weikang Song, Ananda Theertha Suresh, Sebastian U. Stich, Ameet Talwalkar, Hongyi Wang, Blake Woodworth, Shanshan Wu, Felix X. Yu, Honglin Yuan, Manzil Zaheer, Mi Zhang, Tong Zhang, Chunxiang Zheng, Chen Zhu, and Wennan Zhu. A field guide to federated optimization, 2021.

[19] Stanley L. Warner. Randomized response: A survey tech-nique for eliminating evasive answer bias. *Journal ofthe American Statistical Association*, 60(309):63–69, 1965.

A Proofs

Proof for Theorem 3.1: The definition of user level local DP can be easily reframed as a group differential privacy instance, where groups are as large as the entire dataset of each user. More specifically, Equation 5 is the definition of (ε, δ)-group differential privacy for groups of size $g = |D_{u_1}| + |D_{u_2}|$.

Beyond this observation, a simple application of Theorem 3.6 proves (ε, δ)-subject level differential privacy. □

Proof for Lemma 3.2: The gradient clipping step of *UserDPSGD()* forces the bound on the gradient: $\|g(S)\|_2 \leq C$. Thus the sensitivity of parameter gradient $\mathbb{S}_b \leq |C|$. □

Proof for Theorem 3.3: For any two users u_1 and u_2, the parameter gradient for each mini-batch is identically bounded by C. The Gaussian noise added per mini-batch parameter gradient is also drawn from the same distribution $\mathcal{N}(0, \sigma^2 C^2 \mathbf{I})$ scaled to the gradient bound C.

Over T mini-batches, the cumulative parameter gradient is bounded by TC. Thus the aggregate sensitivity of parameter gradients over a training round (T mini-batches) for both u_1 and u_2 is TC.

For every training round, we use the moments accountant technique (1) to compute the correct noise scale σ given a privacy budget of ε, and a DP failure probability of δ.

The resulting parameter updates from both u_1 and u_2 received by the federation server in a training round are effectively *locally randomized* (9).

Parameter updates from both u_1 and u_2, as observed by the federation server, can each be broken down into a signal bounded by $|TC|$ (we can ignore the scaling factor of the learning rate η since it is identical for all users), and the cumulative noise from the distribution $\mathcal{N}(0, T\sigma^2C^2\mathbf{I})$ (by linear composition of random variables with identical Gaussian distributions). More precisely, let $\mathcal{C}$ be the change in parameters affected by any user u_i. Then

$$|\mathcal{C}(u_i)| \leq \eta(|TC| + |\mathcal{N}(0, T\sigma^2C^2\mathbf{I}|) \tag{7}$$

Also note that a dataset containing just u_1 is adjacent to a dataset containing just u_2 since they differ from each other in just one data item. Thus we can apply the classic proof of (ε,δ)-DP for the Gaussian mechanism (6) Theorem A.1 to show that

$$\mathcal{P}(\mathcal{F}(u_1, M) \in M_o) \leq e^{\varepsilon}\mathcal{P}(\mathcal{F}(u_2, M) \in M_o) + \delta \tag{8}$$

We can further extend this guarantee over multiple training rounds with a σ computed correctly using the moments accountant DP-composition technique.

□

Proof for Lemma 3.7: Clipping gradients for each data item belonging to subject $a \in S$ before averaging the clipped gradients over the $\mathcal{S}_a^S$ ensures that the averaged gradients' L2-norm is bounded by C. Hence $\mathbb{S}^S \leq |C|$. □

Proof for Theorem 3.8: Bounding subject sensitivity $\mathbb{S}^S \leq |C|$ and scaling the Gaussian noise to that sensitivity bound clearly results in (ε, δ)-DP guarantee in gradient computation for each subject of each mini-batch S. The mini-batch wide averaging of gradients $\tilde{g}_S$ done using the mini-batch size B is justified since the per subject gradients' average can be restated as aggregation of scaled down gradients for data items corresponding to a subject a; i.e. $g(\mathcal{S}_a^S) = \sum_i \frac{\bar{g}(s_i)}{|\mathcal{S}_a^S|}$. This gives us B distinct gradient quantities for the data items in the sampled mini-batch S, and averaging these quantities requires the term B in the denominator of the expression that computes $\tilde{g}_S$. We can apply the moments accountant method for privacy budget composition to extend the subject level (ε,δ)-DP guarantee over T mini-batches in a training round, aggregated over R training rounds. Thus *HiGradAvgDP* locally enforces subject level (ε,δ)-differential privacy. □

Proof for Theorem 3.9: Assume two distinct users u_1 and u_2 in a federation that host private data items of subject s. Let ε_1 and ε_2 be the respective subject privacy losses incurred by the two users during a training round.

It is straightforward to see that, in the worst case, data items of s at users u_1 and u_2 can affect disjoint parameters in $\mathcal{M}$. Thus parameter averaging done by $\mathcal{F}_g$ simply results in summation and scaling of these disjoint parameter updates. As a result, the privacy losses, ε_1 and ε_2 incurred by u_1 and u_2 respectively are retained to their entirety by $\mathcal{F}_g$. In other words, privacy losses incurred for subject s at users u_1 and u_2 compose sequentially. □

Proof for Theorem 3.10: The proof of training round constraints on horizontal composition can be broken down into two cases: First, each user in the federation locally trains for exactly T mini-batches per training round, with exactly the same mini-batch sampling fraction q. Since horizontal composition is equivalent to sequential composition in the worst case, the moments accountant method shows us that the resulting algorithm will be $(O(q\varepsilon\sqrt{TRs}), \delta)$-differentially private. To compensate for the $\sqrt{s}$ factor scaling of the privacy loss, $\mathcal{F}$ can be executed for $\frac{R}{\sqrt{s}}$ training rounds, yielding a $(O(q\varepsilon\sqrt{TR}), \delta)$-differentially private algorithm.

In the second case, each user u_i may train for a unique number of mini-batches per training round, with a unique mini-batch sampling fraction dictated by u_i's private dataset. Let $T_1, T_2, ..., T_s$, and $q_1, q_2, ..., q_s$ be the number of mini-batches per training round and mini-batch sampling fraction for the sampled users $u_1, u_2, ..., u_s$ respectively.

All our algorithms 1, 2, and 3 locally enforce subject level ($O(q_i\varepsilon\sqrt{T_iR}$),$\delta$)-DP at each user u_i. Privacy enforcement is done independantly at each federation user u_i. Furthermore, note that the privacy loss is uniformly apportioned among training rounds. Let $\mathcal{E} = q_i\varepsilon\sqrt{T_iR}$. Note that $\mathcal{E}$ is identical for each user u_i in the federation. Thus if $\mathcal{E}$ is the total privacy loss budget over R training rounds, a sampled user incurs $\varepsilon_r = \mathcal{E}/R$ privacy loss in a single training round r. Similarly, each of the s sampled users in round r incurs identical privacy loss ε_r despite having different mini-batches per training round T_i and mini-batch sampling fractions q_i. As noted earlier, these privacy losses compose horizontally (sequentially) via $\mathcal{F}_g$ over s users, leading to privacy loss amplification by a factor of $\sqrt{s}$ as per the moments accountant method. To compensate for this privacy loss amplification, $\mathcal{F}$ can be executed for $\frac{R}{\sqrt{s}}$ training rounds. $\square$