Subject Granular Differential Privacy in Federated Learning

VIRENDRA J. MARATHE, Oracle Labs, USA
PALLIKA KANANI, Oracle Labs, USA

Differential Privacy (DP) enforcement in Federated Learning (FL) appears at two granularities in the literature: (i) item level, and (ii) user level. In this paper, we consider a third granularity of privacy – data subject level privacy, 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. Neither item level nor user level privacy are sufficient to enforce subject level privacy. We formally define the notion if subject level DP for FL, and analyze its differences with item and user level DP guarantees. Furthermore, we present two algorithms that enforce subject level DP that build on the notion of group differential privacy. In the process we make some interesting observations: Enforcement of subject level privacy at individual users entails the same privacy even when subjects’ data items are distributed over multiple users. Additionally, while both item and user level DP are insufficient to enforce subject level DP, Local Differential Privacy guarantees subject level privacy, even when a subject’s data items span across multiple users.

1 INTRODUCTION

Federated Learning (FL) is a distributed training paradigm that lets different parties (users) collaborate with each other to jointly train a Machine Learning (ML) model [12]. In the process the users do not share their private training data with any other users. FL provides the benefit of the aggregate training data across all its users, which typically leads to much better performing models. Critically, FL automatically provides training data privacy since the data never leaves the user’s device or silo. However, ML models are known to learn the training data itself, which can leak out at inference time [3, 15, 17, 22].

Differential Privacy (DP) provides a compelling solution to the data leakage problem [5, 6]. Intuitively, a differentially private version of an algorithm \( \mathcal{A} \) introduces enough randomization in \( \mathcal{A} \) that makes it harder for an adversary to determine if any specific data item was used as an input to \( \mathcal{A} \). In the case of ML models, DP ensures that an adversary cannot determine if a specific data item was a part of the training dataset. For ML model training, DP is introduced in the model by adding carefully calibrated noise during training. In the FL setting, this noise is calibrated to hide either the use of any data item, called item level privacy, or the participation of any user, called user level privacy, in the training process [1, 14]. 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 [13, 14].

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, 12] – an individual’s data typically resides in just one cell phone that participates in a federation. However, the cross-silo FL setting [10], 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.

As an example, consider an online retail store customer \( C \). \( C \)'s online purchase history is highly sensitive, and must be kept private. At the same time, \( C \)'s purchase history contains a multitude of orders placed by \( C \) in the past. 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, 14]. User level privacy does not protect the privacy of \( C \)'s data either. User level privacy obfuscates each user’s participation in the training process [14]. However, a subject’s data can be distributed among several users. 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.

Vertical FL [9, 20] is perhaps most closely related to our requirements for data subjects and their private data. In vertical FL, a subject’s features are partitioned between multiple users, and collaborative learning between the users is necessary to train the model. While this distributed nature of a subject’s features shares some traits with our requirements for subject privacy, the work focuses on splitting features between users, and does not consider subjects with multiple data items. A combination of vertical FL and data subject privacy is an interesting research topic that we leave for future work.

In this paper, we consider a third granularity of privacy – subject level privacy [21], where a subject is an individual whose private data is spread across multiple data items, which can in turn be distributed across multiple federation users. To the best of our knowledge, this work is the first to formally characterize subject level privacy in terms of the notion of subject level differential privacy.

We present two novel algorithms that achieve subject level DP. Our algorithms leverage the intuition behind group differential privacy [5, 6], and use group composition results to dynamically enforce group level privacy for data items belonging to the same subject. Our first algorithm, called CentralSubDP, delegates subject level DP enforcement to a trusted central federation server. Our second algorithm, called LocalSubDP, lets each user locally enforce subject level DP guarantee, which automatically extends to subject level DP across multiple users. We show that both our algorithms enforce subject level DP. We also show that Local Differential Privacy [4, 11, 19, 23]

\[1\] Wang et al. [21] identify what we call subjects in this paper as users in their paper.
is a stricter privacy guarantee that subsumes subject level privacy. Empirical evaluation of our algorithms is the subject of future work.

The rest of the paper is structured as follows: We present formal definitions of DP and subject level DP in section 2. Our algorithms are described in section 3, along with informal arguments of their subject level DP guarantees. We also show that local DP subsumes subject level DP. We conclude in section 4.

2 SUBJECT LEVEL DIFFERENTIAL PRIVACY

We begin with the definition of DP [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 $\mathcal{A} : D \rightarrow R$ is said to be $(\epsilon, \delta)$-differentially private if for any two adjacent datasets $D$, $D' \in D$, and set $R \subseteq R$,

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

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

The above definition directly provides item level privacy. McMahan et al. [14] 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 $D_i$ be the dataset of user $u_i \in \mathcal{U}$. Let $D_{\mathcal{U}} = \bigcup_{i=1}^n D_i$. Let $\mathcal{M}$ be the domain of models resulting from the FL training process.

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

$$P(\mathcal{F}(D_U) \in R) \leq e^\epsilon P(\mathcal{F}(D_{U'}) \in R) + \delta \quad (2)$$

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

Let $\mathcal{S}$ be the 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. $D_{\mathcal{U}} = \bigcup_{s \in \mathcal{S}} D_s$).

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

$$P(\mathcal{F}(D_S) \in R) \leq e^\epsilon P(\mathcal{F}(D_{S'}) \in R) + \delta \quad (3)$$

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) [21].

3 ENFORCING SUBJECT LEVEL DIFFERENTIAL PRIVACY

As noted earlier, guaranteeing item level DP in a randomized algorithm $\mathcal{A}$ is insufficient for subject level DP since item level DP obfuscates just a single data item’s contribution to $\mathcal{A}$’s output, whereas hiding a subject may entail obfuscation of multiple data items belonging to that subject. Similarly, guaranteeing user level DP in $\mathcal{A}$ is insufficient for subject level DP since user level DP obfuscates a single user’s contribution to $\mathcal{A}$’s output, whereas hiding a subject may entail obfuscation of multiple users’ data items belonging to that subject. Intuitively, to enforce subject level DP, we need to obfuscate the effects of a 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 3.1.** Any $(\epsilon, \delta)$-differentially private randomized algorithm $\mathcal{A}$ is $(ge^{(g-1)\delta} \epsilon)$-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,

$$P(A(D) \in R) \leq e^{\epsilon g} P(A(D') \in R) + ge^{(g-1)\delta}(g \epsilon \delta) \quad (4)$$

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

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

Theorem 3.1 is a bi-directional implication. So it can be restated as follows:

**Theorem 3.2.** Any $(\epsilon, \delta)$-group differentially private algorithm $\mathcal{A}$, for a group size of $g$, is $(\epsilon/g, \delta/(ge^{(g-1)\delta}))$-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 $M$ be the domain space of the ML model being trained in the FL setting.

**Theorem 3.3.** Let training algorithm $\mathcal{A}_g : S \rightarrow M$ be group differentially private for groups of size $g$, and I be the largest number of data items belonging to any single subject in $S$. If $I \leq g$, 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, 16]. For instance, the moments accountant method by Abadi et al. [1] shows that given an $(\epsilon, \delta)$-DP gradient computation for a single mini-batch, the full training algorithm, which consisting of $T$ mini-batches and a mini-batch sampling fraction of $q$, is $(O(qg^{\sqrt{T}}), \delta)$-differentially private. Theorem 3.1 implies that the same algorithm is $(O(qg^{\sqrt{T}}), ge^{(g-1)\delta})$-group differentially private for a group of size $g$.

We now present two new FL training algorithms that guarantee group DP, which as per Theorem 3.3 implies subject level DP. In both the algorithms, the federation server first initializes the common DNN model and distributes it to the users. We make a critical assumption.
in these algorithms: 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.

3.1 CentralSubDP

CentralSubDP, which is based on FedSGD [12], enforces subject level privacy at the federation server. In FedSGD [12], the federation server randomly samples a set of users and sends them a request to train the previously shipped model. Each user in turn computes gradients of a randomly selected mini-batch of local data, and then returns the gradients to the federation server. The federation server accumulates the received gradients, averages them over all responses from users, applies the gradients to its common model, and then redistributes the updated model to the users. This is a single training round. The federation server repeats this process until convergence or a threshold number of training rounds has elapsed.

Algorithm 1: Pseudo code for CentralSubDP that guarantees subject level DP.

Parameters: Set of \( n \) users \( \mathcal{U} = u_1, u_2, \ldots, u_n; \mathcal{D}_i \), the dataset of user \( u_i; \mathcal{M} \), the model to be trained; \( \theta \), the parameters of model \( \mathcal{M} \); gradient norm bound \( C \); mini-batch size \( B \); largest group size \( Z \) in a mini-batch; noise scale \( \sigma_Z \) for group size \( Z \); \( R \) training rounds; the learning rate \( \eta \).

1 User_CentralSubDP(\( u_i \)):  
2 \( S = \) random sample of \( B \) data items from \( \mathcal{D}_i \)
3 for \( s_i \in S \) do
4 Compute gradients:
5 \( g(s_i) = \nabla L(\theta, s_i) \)
6 Clip gradients:
7 \( g_s = \text{Clip}(g(s_i), C) \)
8 end
9 \( Z = \text{LrgGrpCnt}(S) \)
10 return \( \frac{1}{|S|} \sum g(s_i), Z \)

11 Server_CentralSubDP():
12 for \( r = 1 \) to \( R \) do
13 \( U_s = \) sample \( s \) users from \( \mathcal{U} \)
14 \( \bar{G} = 0 \)
15 for \( u_i \in U_s \) do
16 \( g_s, Z = \text{User_CentralSubDP}(u_i) \)
17 \( \bar{G} = \bar{G} + g_s \)
18 end
19 \( \theta = \theta - \eta \bar{G} \)
20 Send \( \bar{M} \) to all users in \( \mathcal{U} \)
21 end

Algorithm 1 shows the pseudo code of CentralSubDP. Like prior work [1, 14, 18], we enforce DP in CentralSubDP 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, the user also tracks the item count of the subject with the largest number of items in the sampled mini-batch.

Along with mini-batch gradients, the federation server requests the largest item count for that mini-batch towards the federation server.

3.2 LocalSubDP

Algorithm 2: Pseudo code for LocalSubDP that guarantees subject level DP.

Parameters: Set of \( n \) users \( \mathcal{U} = u_1, u_2, \ldots, u_n; \mathcal{D}_i \), the dataset of user \( u_i; \mathcal{M} \), the model to be trained; \( \theta \), the parameters of model \( \mathcal{M} \); gradient norm bound \( C \); sample of users \( U_s \); mini-batch size \( B \); \( Z \), largest group size in a mini-batch; \( \sigma_Z \), noise scale for group of size \( Z \); \( R \) training rounds; \( T \) batches per round; the learning rate \( \eta \).

1 User_LocalSubDP(\( u_i \)):  
2 \( S = \) random sample of \( B \) data items from \( \mathcal{D}_i \)
3 for \( s_i \in S \) do
4 Compute gradients:
5 \( g(s_i) = \nabla L(\theta, s_i) \)
6 Clip gradients:
7 \( g(s_i) = \text{Clip}(g(s_i), C) \)
8 end
9 \( Z = \text{LrgGrpCnt}(S) \)
10 return \( \frac{1}{|S|} \sum g(s_i), Z \)

11 Server_LocalSubDP():
12 for \( r = 1 \) to \( R \) do
13 \( U_s = \) sample \( s \) users from \( \mathcal{U} \)
14 for \( u_i \in U_s \) do
15 \( \theta_i = \text{User_LocalSubDP}(u_i) \)
16 end
17 \( \theta = \frac{1}{T} \sum \theta_i \)
18 Send \( \bar{M} \) to all users in \( \mathcal{U} \)
19 end

LocalSubDP, which is based on DP-SGD [1], enforces subject level privacy locally at each user. DP-SGD was originally not designed for a FL setting, but can be easily tweaked to enforce item
level DP in the FL setting: The federation server samples a random set of users for each training round and sends them a request to perform local training. Each user in turn trains for a multitude of mini-batches, even multiple epochs, and introduces carefully calibrated Gaussian noise in parameter gradients computed for each mini-batch. For each mini-batch, gradients are computed for each data item separately, and clipped to the threshold $C$ to bound the gradients’ sensitivity. The gradients are then summed over the full mini-batch, and Gaussian noise scaled to $C$ is added to the sum. This sum is then averaged over the mini-batch size, and applied to the parameters. The users ship back updated model parameters to the federation server, which then simply averages the updates received from all the sampled users. The server redistributes the updated model and triggers another training round if needed.

In the original paper [1], Abadi et al. also proposed the moments accountant method for tighter composition of privacy loss bounds compared to prior work on strong composition [7]. The same composition results trivially apply to the algorithm described above.

The algorithm described thus far enforces item level DP. We extend it to support subject level DP in a manner similar to CentralSubDP. Our algorithm appears in Algorithm 2. We compute each mini-batch’s group size $Z$ as the largest number of items of any subject appearing in the mini-batch. We use $Z$, and other parameters $(\mathcal{E}, \Delta)$, number of mini-batches $TR$, and the mini-batch sampling fraction $B/|D_t|$, to compute the appropriate value of $\sigma_Z$, and subsequently add the computed noise to the mini-batch gradients’ sum before averaging. As in the case of CentralSubDP, LocalSubDP enforces $(\mathcal{E}/Z, \Delta/(Ze^{(Z-1)/2} \sigma_Z^2))$-differential privacy, which by Theorem 3.2 implies $(\mathcal{E}, \Delta)$-group differential privacy, hence subject level DP by Theorem 3.3.

Although both our algorithms add noise at the granularity of individual mini-batches belonging to individual users, they enforce subject level DP even for subjects whose data items are distributed across multiple users. This is because, intuitively, each subject’s signal in a mini-batch is completely obfuscated by noise added to the mini-batch. Therefore, even though a subject’s data items appear in mini-batches of multiple users in a single training round, each of those mini-batches individually obfuscate that subject’s signal. Thus, in the aggregate, the entire signal of the subject (over all sampled users’ mini-batches in a training round) is completely obfuscated. This intuition is trivially provable for CentralSubDP due to the additivity property of Gaussian noise (due to averaging of gradients – line 20 in Algorithm 1).

### 3.3 Local Differential Privacy

In certain settings, the user does not trust the federation server, or is prohibited to share unperturbed model parameter updates or gradients due to specific interpretations of localization laws [8]. As a result, the user may need to perturb model parameters before shipping them back to the federation server. This setting has an uncanny resemblance to the motivation for classic Local Differential Privacy [4, 11, 23], where a data analyst can get access to the data only after it has been perturbed. Work on LDP in the FL setting has just begun [19]. In the FL setting, LDP is a much stronger privacy guarantee than item or user level DP in that it completely obfuscates the 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.

**Definition 3.4.** We say that FL algorithm $F : \mathcal{U} \rightarrow M$ is user level local $(r, \delta)$-differentially private, where $\mathcal{U}$ is the set of users in the federation and $M$ is the domain of parameters of the model getting trained, if for any two users $u_1, u_2 \in \mathcal{U}$, and $S \subseteq M$,

$$\mathcal{P}(F(u_1) \in S) = e^\epsilon \mathcal{P}(F(u_2) \in S) + \delta$$

(5)

In this paper we do not propose a new local DP algorithm. However, we note that local DP is a stronger privacy guarantee than user level DP. Intuitively, this is because by definition, a local DP algorithm must locally obfuscate the entire signal of every user [19]. Whereas subject level DP entails obfuscating the signal corresponding to any single subject. In the extreme case, where a user’s signal is completely dominated by a single subject, we find an equivalence between locally enforced subject level DP (e.g. LocalSubDP) and local DP. Furthermore, similar to CentralSubDP and LocalSubDP, enforcing user level local DP ensures subject level DP even for the subjects whose data items are distributed among multiple users. We leave out a more formal treatment of these observations from this paper due to space restrictions.

### 4 CONCLUSION

While various prior works on privacy in FL have explored DP guarantees at the user and item levels [13, 14], to the best of our knowledge, no prior work has studied subject level granularity for privacy, where a subject is an individual who has multiple data items in the FL training dataset, potentially distributed across multiple federation users. In this paper, we presented a formal definition of subject level DP. We also presented two novel FL training algorithms that guarantee subject level DP. Our algorithms leverage the insight that data items of a single subject form a group, and enforcing group DP on each mini-batch’s training output is sufficient to enforce subject level DP. We furthermore observed that local DP is a stronger privacy guarantee that can be used to enforce subject level DP guarantee. We leave more thorough formal treatment of subject level DP guarantees of our algorithms, as well as their empirical evaluation for future work.

### REFERENCES


