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][1], 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]

Authors' addresses: Virendra J. Marathe, Oracle Labs, Nashua, NH, USA, virendra.marathe@oracle.com; Pallika Kanani, Oracle Labs, Burlington, MA, USA, pallika.kanani@oracle.com.

2021. XXXX-XXXX/2021/10-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

[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 $\mathcal{A} : \mathcal{D} \rightarrow \mathcal{R}$ 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 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 $\mathcal{D}_i$ 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 domain of models resulting from the FL training process.

Definition 2.2. Given a FL training algorithm $\mathcal{F} : \mathcal{D}_{\mathcal{U}} \rightarrow \mathcal{M}$, 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 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. $\mathcal{D}_{\mathcal{U}} = \bigcup_{s \in \mathcal{S}} \mathcal{D}_s$).

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) [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 (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,*

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

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 ε, and an even bigger penalty in the failure probability term ($ge^{(g-1)\varepsilon}\delta$). 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 ($\mathcal{E}$, Δ)-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 3.3. *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, 16]. 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 consisting of T mini-batches and a mini-batch sampling fraction of q, is $(O(q\varepsilon\sqrt{T}), \delta)$-differentially private. Theorem 3.1 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 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 $\mathcal{U} = u_i, u_2, ..., u_n$; $\mathcal{D}_i$, the dataset of user u_i; M, the model to be trained; θ, the parameters of model M ; gradient norm bound C; mini-batch size B; largest group size in a mini-batch Z; noise scale σ_Z for group size Z; R training rounds; the learning rate η.

User_CentralSubDP(u_i):
- S = random sample of B data items from $\mathcal{D}_i$
- **for** $s_i \in S$ **do**
  - **Compute gradients:**
  - $g(s_i) = \nabla \mathcal{L}(\theta, s_i)$
  - **Clip gradients:**
  - $\bar{g}(s_i) = Clip(g(s_i), C)$
- **end**
- $Z = LrgGrpCnt(S)$
- **return** $\frac{1}{B}\sum_i \bar{g}(s_i), Z$

Server_CentralSubDP():
- **for** $r = 1$ **to** R **do**
  - U_s = sample s users from $\mathcal{U}$
  - $\tilde{G} = 0$
  - **for** $u_i \in U_s$ **do**
    - $\bar{g}_s, Z =$ *User_CentralSubDP*(u_i)
    - $\tilde{g}_s = \bar{g}_s + \frac{1}{B}\mathcal{N}(0, \sigma_Z^2 C^2 \mathbf{I})$
    - $\tilde{G} = \tilde{G} + \tilde{g}_s$
  - **end**
  - $\theta = \theta - \eta \frac{\tilde{G}}{s}$
  - Send M to all users in $\mathcal{U}$
- **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 user returns the largest item count for that mini-batch to the federation server. The count lets the federation server know the *group size* needed to enforce group DP for that mini-batch. This group size, Z in Algorithm 1, 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 Theorem 3.2 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. The rest of the parameters to calculate σ_Z – $\mathcal{E}$, Δ, total number of mini-batches (sR), and sampling fraction ($B/total\ dataset\ size$) – remain the same throughout the training process.

The server adds appropriate amount of noise to gradients received from each user, and then averages the gradients before applying them to the model's parameters using the prescribed learning rate. Like *FedSGD*, the server broadcasts the updated model to users and repeats with the next training round, if any.

It is straightforward to see that the calculation of σ_Z enforces $(\mathcal{E}/Z, \Delta/(Ze^{(Z-1)\frac{\mathcal{E}}{Z}})$-differential privacy, which by Theorem 3.2 implies $(\mathcal{E},\Delta)$-group differential privacy, hence subject level DP by Theorem 3.3.

3.2 *LocalSubDP*

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

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.

User_LocalSubDP(u_i):
- **for** $t = 1$ **to** T **do**
  - S = random sample of B data items from $\mathcal{D}_i$
  - **for** $s_i \in S$ **do**
    - **Compute gradients:**
    - $g(s_i) = \nabla \mathcal{L}(\theta, s_i)$
    - **Clip gradients:**
    - $\bar{g}(s_i) = Clip(g(s_i), C)$
  - **end**
  - $Z = LrgGrpCnt(S)$
  - $\tilde{g}_s = \frac{1}{B}(\sum_i \bar{g}(s_i) + \mathcal{N}(0, \sigma_Z^2 C^2 \mathbf{I}))$
  - $\theta = \theta - \eta \frac{\tilde{g}_s}{s}$
- **end**
- **return** M

Server_LocalSubDP():
- **for** $r = 1$ **to** R **do**
  - U_s = sample s users from $\mathcal{U}$
  - **for** $u_i \in U_s$ **do**
    - $\theta_i =$ *User_LocalSubDP*(u_i)
  - **end**
  - $\theta = \frac{1}{s}\sum_i \theta_i$
  - Send M to all users in $\mathcal{U}$
- **end**

LocalSubDP, which is based on *DP-SGD* [1], enforces subject level privacy locally at each user. *DP-SGD* was originially 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 ($\mathcal{E}$, Δ, number of mini-batches TR, and the mini-batch sampling fraction $B/|\mathcal{D}_i|$), to compute the appropriate value of σ_Z, and subsequently add the computed noise to the mini-batch gradients' sum before averaging. As in the case of *CentralSubDP*, *LocalSubDP* enforces $(\mathcal{E}/Z, \Delta/(Ze^{(Z-1)\frac{\mathcal{E}}{Z}})$-differential privacy, which by Theorem 3.2 implies ($\mathcal{E}$,Δ)-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 $\mathcal{F} : \mathcal{U} \rightarrow \mathcal{M}$ is *user level local (ε,δ)-differentially private*, where $\mathcal{U}$ is the set of users in the federation and $\mathcal{M}$ is the domain of parameters of the model getting trained, if for any two users $u_1, u_2 \in \mathcal{U}$, and $S \subseteq \mathcal{M}$,

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

In this paper we do not propose a new local DP algorithm. However, we note that local DP is a stronger privacy guarantee than subject 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 multipled 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

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep Learning with Differential Privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. 308–318.

[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. 2019. Towards Federated Learning at Scale: System Design. *CoRR* abs/1902.01046 (2019).

[3] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. 2019. The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks. In *28th USENIX Security Symposium*. 267–284.

[4] John C. Duchi, Michael I. Jordan, and Martin J. Wainwright. 2013. Local Privacy and Statistical Minimax Rates. *CoRR* abs/1302.3203 (2013). http://arxiv.org/abs/1302.3203

[5] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating Noise to Sensitivity in Private Data Analysis. In *Proceedings of the Third Conference on Theory of Cryptography (TCC'06)*. 265–284.

[6] Cynthia Dwork and Aaron Roth. 2014. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends in Theoretical Computer Science* 9, 3–4 (Aug. 2014), 211–407.

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

[8] gdpr [n. d.]. General Data Protection Regulation (GDPR). *https://gdpr-info.eu/*.

[9] Stephen Hardy, Wilko Henecka, Hamish Ivey-Law, Richard Nock, Giorgio Patrini, Guillaume Smith, and Brian Thorne. 2017. Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption. *CoRR* abs/1711.10677 (2017). http://arxiv.org/abs/1711.10677

[10] 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. 2019. Advances and Open Problems in Federated Learning. *CoRR* abs/1912.04977 (2019). http://arxiv.org/abs/1912.04977

[11] Shiva Prasad Kasiviswanathan, Homin K. Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam D. Smith. 2008. What Can We Learn Privately? *CoRR* abs/0803.0924 (2008). http://arxiv.org/abs/0803.0924

[12] Jakub Konecný, Brendan McMahan, and Daniel Ramage. 2015. Federated Optimization: Distributed Optimization Beyond the Datacenter. *CoRR* abs/1511.03575 (2015). http://arxiv.org/abs/1511.03575

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

[14] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. 2018. Learning Differentially Private Recurrent Language Models. In *6th International Conference on Learning Representations, ICLR 2018*.

[15] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. 2018. Inference Attacks Against Collaborative Learning. *CoRR* abs/1805.04049 (2018). http://arxiv.org/abs/1805.04049

[16] Ilya Mironov. 2017. Renyi Differential Privacy. *CoRR* abs/1702.07476 (2017). http://arxiv.org/abs/1702.07476

[17] Milad Nasr, Reza Shokri, and Amir Houmansadr. 2019. 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 2019, San Francisco, CA, USA, May 19-23, 2019*. IEEE, 739–753. https://doi.org/10.1109/SP.2019.00065

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

[19] Stacey Truex, Ling Liu, Ka Ho Chow, Mehmet Emre Gursoy, and Wenqi Wei. 2020. 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*. ACM, 61–66.

[20] Chang Wang, Jian Liang, Mingkai Huang, Bing Bai, Kun Bai, and Hao Li. 2020. Hybrid Differentially Private Federated Learning on Vertically Partitioned Data. *CoRR* abs/2009.02763 (2020). https://arxiv.org/abs/2009.02763

[21] 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. 2021. A Field Guide to Federated Optimization. arXiv:cs.LG/2107.06917

[22] Kangkang Wang, Rajiv Mathews, Chloé Kiddon, Hubert Eichner, Françoise Beaufays, and Daniel Ramage. 2019. Federated Evaluation of On-device Personalization. *CoRR* abs/1910.10252 (2019). http://arxiv.org/abs/1910.10252

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