Federated Learning with Bayesian Differential Privacy

Aleksei Triastcyn
Artificial Intelligence Lab
EPFL
Lausanne, Switzerland
aleksei.triastcyn@epfl.ch

Boi Faltings
Artificial Intelligence Lab
EPFL
Lausanne, Switzerland
boi.faltings@epfl.ch

Abstract—We consider the problem of reinforcing federated learning with formal privacy guarantees. We propose to employ Bayesian differential privacy, a relaxation of differential privacy for similarly distributed data, to provide sharper privacy loss bounds. We adapt the Bayesian privacy accounting method to the federated setting and suggest multiple improvements for more efficient privacy budgeting at different levels. Our experiments show significant advantage over the state-of-the-art differential privacy bounds for federated learning on image classification tasks, including a medical application, bringing the privacy budget below $\varepsilon = 1$ at the client level, and below $\varepsilon = 0.1$ at the instance level. Lower amounts of noise also benefit the model accuracy and reduce the number of communication rounds.

Index Terms—federated learning, differential privacy, privacy accounting, deep learning

I. INTRODUCTION

The rise of data analytics and machine learning (ML) presents countless opportunities for companies, governments and individuals to benefit from the accumulated data. At the same time, their ability to capture fine levels of detail potentially compromises privacy of data providers. Recent research [1], [2] suggests that even in a black-box setting it is possible to argue about the presence of individual records in the training set or recover certain features of these records.

To tackle this problem a number of solutions has been proposed. They vary in how privacy is achieved and to what extent data is protected. One approach that assumes privacy at its core is *federated learning* (FL) [3]. In the FL setting, a central entity (*server*) trains a model on user data without actually copying data from user devices. Instead, users (*clients*) update models locally, and the *server* aggregates these updates.

In spite of all the advantages, federated learning does not provide theoretical privacy guarantees, like it is done by *differential privacy* (DP) [4], which is viewed by many researchers as the privacy gold standard. Initially, DP algorithms focused on sanitising simple statistics, such as mean, median, etc., using a technique known as output perturbation. In recent years, the field made a lot of progress towards the goal of privacy-preserving machine learning, through works on objective perturbation [5], stochastic gradient descend with DP updates [6], to more complex and practical methods [7]–[10].

As shown in recent work [10], [11], the two approaches can be combined to provide joint benefits. However, unless the number of users is exceedingly high (e.g. in the scenario of a large population of mobile users considered in [10]), differentially private federated learning provides only weak guarantees. Contrary to a wide-spread opinion in machine learning community, values of ε close to 10 can hardly be seen as reassurance to a user: for certain types of attacks, an adversary can theoretically reach accuracy of 99.99%.

We propose to augment federated learning with a natural relaxation of differential privacy, called *Bayesian differential privacy* (BDP) [12], that provides tighter, and thus, more meaningful guarantees. The main idea of this relaxation is based on the observation that machine learning tasks are often restricted to a particular type of data (for example, finding a film review in the MRI dataset is very unlikely). Moreover, this information, and potentially even some prior distribution of data, is often available to the attacker. While the traditional DP treats all data as equally likely and hides differences by large amounts of noise, BDP calibrates noise to the data distribution. Hence, for any two datasets drawn from the same (arbitrary) distribution, and given the same privacy mechanism with the same amount of noise, BDP provides tighter guarantees than DP. Note that the full knowledge of this distribution is not required, as the necessary statistics can be estimated from data.

We introduce the notion of Bayesian differential privacy in Section IV and extend it to the federated learning setting in Section V. Our experiments (see Section VI) show significant advantage, both in privacy guarantees and the model quality.

The main contributions of this paper are the following:

- we adapt the notion of Bayesian differential privacy to federated learning, including more natural non-i.i.d. settings (Section V-A), to provide strong theoretical privacy guarantees under minor and practical assumptions;
- we propose a novel joint accounting method for estimating client-level and instance-level privacy simultaneously and securely (Section V-C);
- we experimentally demonstrate advantages of our method, such as shrinking the privacy budget to a fraction of the previous state-of-the-art, and improving the accuracy of the trained models by up to 10% (Section VI).

II. Related Work

As machine learning applications become more and more common, various vulnerabilities and attacks on ML models get discovered, based on both passive (for example, model inversion [1] and membership inference [2]) and active adversaries (e.g. [13]), raising the need for developing matching defences.

Differential privacy [4], [14] is one of the strongest privacy standards that can be employed to protect ML models from these and other attacks. Since pure ε-DP is hard to achieve in many realistic learning settings, a notion of approximate (ε, δ)-DP is used across-the-board in machine learning. It is often achieved as a result of applying the Gaussian noise mechanism [15].

For a long time, however, even approximate DP remained unachievable in more popular deep learning scenarios. Some earlier attempts [16] led to prohibitively high bounds on ε [7], [8] that were later shown to be ineffective against attacks [13]. A major step in the direction of bringing privacy loss values down to more practical magnitudes was done by [7] with the introduction of the *moments accountant*, currently a state-of-the-art method for keeping track of the privacy loss during training. Followed by improvements in differentially private training techniques [8], [9], it allowed to achieve single-digit DP guarantees ($\varepsilon < 10$) for classic supervised learning benchmarks, such as MNIST, SVHN, and CIFAR.

On the other end of spectrum, McMahan et al. [3] proposed federated learning as one possible solution to privacy issues (among other problems, such as scalability and communication costs). In this setting, privacy is enforced by keeping data on user devices and only submitting model updates to the server. Two of the popular approaches are the federated stochastic gradient descent (`FedSGD`) and federated averaging (`FedAvg`) [3], where *clients* do local on-device gradient descent using their data, then send these updates to the *server*, which applies an average update to the model. Privacy can further be enhanced by using secure multi-party computation (MPC) [17] to allow the server access only average updates of a big group of users and not individual ones. However, MPC or homomorphic encryption do not guarantee robustness against model inversion or membership inference [1], [2], because these attacks operate on the resulting model which remains the same. Alternatively, by using differential privacy and the moments accountant, [10] and [11] attained theoretical client-level privacy guarantees for federated learning settings.

Since federated learning on its own lacks theoretical privacy guarantees, combining it with a more formal notion of privacy is an attractive direction of research. On the other hand, for more complicated deep learning models, differential privacy leads to a poor privacy-utility trade-off, suggesting that pairing federated learning with an alternative notion might prove beneficial.

Apart from differential privacy, a number of alternative definitions have been proposed over the recent years, such as computational DP [18], mutual-information privacy [19], [20], different versions of concentrated DP (CDP [21], zCDP [22],

tCDP [23]), and Rényi DP (RDP) [24]. Some other relaxations [25]–[27] tip the balance even further in favour of applicability at the cost of weaker guarantees, for example considering the average-case instead of the worst-case [28].

In general, important aspects of a privacy notion are composability, accountability, and interpretability. Apart from sharp bounds, the moments accountant is attractive because it operates within the classic notion of (ε, δ)-DP. Some of the alternative notions of DP also provide tight composition theorems, along with some other advantages, but to the best of our knowledge, they are not broadly used in practice compared to traditional DP (although there are some examples [29]). One of the possible reasons for that is interpretability: parameters of (α, ε)-RDP or (μ, τ)-CDP are hard to interpret. While it may be difficult to quantify the actual guarantee provided by specific values of ε, δ of the traditional DP, it is still advantageous that they have a clearer probabilistic interpretation.

In this work, we rely on another relaxation, called Bayesian differential privacy [12]. This notion boosts privacy accounting efficiency by utilising the fact that data come from a particular distribution, and not all data are equally likely. At the same time, it maintains the probabilistic interpretation of its parameters ε and δ. It is worth noting, that unlike some of the relaxations mentioned above, the notion of Bayesian DP provides the worst-case guarantee (under specified conditions) and is not limited to a particular dataset, but rather a particular type of data (e.g. emails, MRI images, etc.), or a mixture of such types, which is a much more permitting assumption.

III. Preliminaries

In this section, we provide necessary definitions, background and notation used in the paper. We also describe the general setting of the problem.

A. Definitions and notation

We use D, D' to represent neighbouring (adjacent) datasets. If not specified, it is assumed that these datasets differ in a single example. Individual examples in a dataset are denoted by x or x_i, while the example by which two datasets differ— by x'. We assume $D' = D \cup \{x'\}$, whenever possible to do so without loss of generality.

Since this paper mainly deals with adding noise to gradients (w.r.t. model parameters, neural network weights, etc.), we often refer to non-noised gradients as *non-private outcome*, and denote it by g, g'. The private learning outcomes are denoted by w. Whenever it is ambiguous, we denote expectation over data (or equivalently, gradients) as $\mathbb{E}_x$, and over the learning outcomes as $\mathbb{E}_w$. Finally, for federated learning scenarios, u_i indicates an update of a user i, while $\mathbb{U}$—a set of all user updates.

Definition 1. *A randomised function (algorithm) $\mathcal{A} : \mathcal{D} \to \mathcal{R}$ with domain $\mathcal{D}$ and range $\mathcal{R}$ satisfies (ε, δ)-differential privacy if for any two adjacent inputs $D, D' \in \mathcal{D}$ and for any set of outcomes $\mathcal{S} \subset \mathcal{R}$ the following holds:*

$$\Pr\left[\mathcal{A}(D) \in \mathcal{S}\right] \le e^{\varepsilon} \Pr\left[\mathcal{A}(D') \in \mathcal{S}\right] + \delta.$$

Definition 2. *The privacy loss $L_{\mathcal{A}}$ of a randomised algorithm $\mathcal{A}: \mathcal{D} \to \mathcal{R}$ for outcome $w \in \mathcal{R}$ and datasets $D, D' \in \mathcal{D}$ is given by:*

$$L_{\mathcal{A}}(w, D, D') = \log \frac{\Pr\left[\mathcal{A}(D) = w\right]}{\Pr\left[\mathcal{A}(D') = w\right]}.$$

For notational simplicity, we omit the designation $\mathcal{A}$, i.e. we use $L(w, D, D')$ (or simply L) for the privacy loss random variable, and $p(w|D)$, and $p(w|D')$ for the outcome probability distributions for given datasets. Also, note that the privacy loss random variable L is distributed by drawing $w \sim p(w|D)$ (see [21, Section 2.1 and Definition 3.1]), which helps linking it to well-known divergences.

Definition 3. *The Gaussian noise mechanism achieving (ε, δ)-DP, for a function $f : \mathcal{D} \to \mathbb{R}^m$, is defined as*

$$\mathcal{M}(D) = f(D) + \mathcal{N}(0, I\sigma^2),$$

where $\sigma > C\sqrt{2\log\frac{1.25}{\delta}}/\varepsilon$ and $C = \max_{D, D'} \|f(D) - f(D')\|$ is the L2-sensitivity of f.

For more details on differential privacy and the Gaussian mechanism, we refer the reader to [15].

We will also need the definition of Rényi divergence:

Definition 4. *Rényi divergence of order λ between distributions P and Q, denoted as $D_{\lambda}(P\|Q)$ is defined as*

$$D_{\lambda}(P\|Q) = \frac{1}{\lambda - 1}\log\int p(x)\left[\frac{p(x)}{q(x)}\right]^{\lambda - 1}dx$$
$$= \frac{1}{\lambda - 1}\log\int q(x)\left[\frac{p(x)}{q(x)}\right]^{\lambda}dx,$$

where $p(x)$ and $q(x)$ are corresponding density functions of P and Q.

Analytic expressions for Rényi divergence exist for many common distributions and can be found in [30]. Van Erven and Harremos [31] provide a good survey of Rényi divergence properties in general.

B. Setting

In the first part of the paper, while describing the concept of Bayesian differential privacy, we consider a general iterative learning algorithm, such that each iteration t produces a non-private learning outcome $x^{(t)}$ (e.g. a gradient over a batch of data). In the second half of the paper, we consider the equivalent federated learning setting, where each communication round t produces a set of non-private learning outcomes $u_i^{(t)}$, one for each client i.

The non-private outcome, in both scenarios, gets transformed into a private learning outcome $w^{(t)}$ that is used as a starting point for the next iteration or communication round. The learning outcome can be made private by different means, but in this work we consider the most common approach of applying an additive noise mechanism (e.g. a Gaussian noise mechanism). We denote the distribution of private outcomes by $p(w^{(t)}|w^{(t-1)}, D)$ (we assume the Markov property of the

learning process for brevity of notation, but it is not necessary in general) or $p(w^{(t)}|w^{(t-1)}, \mathbb{U})$, depending on the scenario.

The process can run on subsamples of data or subsets of clients, in which case $w^{(t)}$ comes from the distribution $p(w^{(t)}|w^{(t-1)}, B^{(t)})$, where $B^{(t)}$ is a batch of data used for parameters update in iteration t, or $p(w^{(t)}|w^{(t-1)}, \mathbb{U}^{(t)})$, where $\mathbb{U}^{(t)}$ is a set of updates from users participating in the communication round t. In these cases, privacy is amplified through sampling [32].

For each iteration, we would like to compute a quantity c_t (we call it a *privacy cost*) that accumulates over the learning process and allows to compute privacy loss bounds ε, δ using concentration inequalities. The overall privacy accounting workflow does not significantly differ from prior work, but is in fact a generalisation of the well-known moments accountant [7]. Importantly, it is not tied to a specific learning algorithm or a class of algorithms, as long as one can map it to the above setting.

C. Motivation

Before we proceed, we find it important to motivate the research and usage of alternative definitions of privacy. The primary reason for this is that the complexity of the concept of differential privacy often leads to misunderstanding or overestimation of the guarantees it provides. And while we do not fully tackle the problem of interpretability, we provide a simple example below that allows to better judge the quality of provided guarantees.

Consider the state-of-the-art differentially private machine learning models [7], [9]. In order to come close to the non-private accuracy (say within 10% of it), all of the reported models stretch their privacy budget to $\varepsilon > 2$ (for a reasonably low δ), while in many cases it goes up to $\varepsilon > 5$. In real-world applications, it can even be larger than 10[1]. These numbers seem small, and thus, may often be overlooked. But let us present an alternative interpretation.

What we are interested in is the change in the posterior distribution of the attacker after they see the private model compared to prior [24], [33]. Let us consider the stronger, pure DP for simplicity. According to the definition of ε-DP:

$$\frac{p(D|w)}{p(D'|w)} \leq e^{\varepsilon}\frac{p(D)}{p(D')}.$$

Assume the following specific example. The datasets D, D' consist of income values for residents of a small town. There is one individual x' whose income is orders of magnitude higher than the rest, and whose residency in the town is what the attacker wishes to infer. The attacker observes the mean income w sanitised by a differentially private mechanism with $\varepsilon = \varepsilon_0$. If the individual is not present in the dataset, the probability of w being above a certain threshold is extremely small. On the contrary, if x' is present, this probability is higher (say it is equal to r). The attacker takes a Bayesian approach, computes the likelihood of the observed value

---

[1]https://www.wired.com/story/apple-differential-privacy-shortcomings/

under each of the two assumptions and the corresponding posteriors given a flat prior. The attacker then concludes that the individual is present in the dataset and is a resident.

By the above expression, r can only be e^{ε_0} times larger than the corresponding probability without x'. But if the $re^{-\varepsilon_0}$ is small enough, then the probability $P(A)$ of the attacker's guess being correct is as high as $\frac{r}{r+re^{-\varepsilon_0}}$ or, equivalently,

$$P(A) = \frac{1}{1 + e^{-\varepsilon}}. \tag{1}$$

To put it in perspective, for a DP algorithm with $\varepsilon = 5$, the upper bound on the accuracy of this attack is as high as 99.33%. For $\varepsilon = 8$, it is 99.97%. For $\varepsilon = 10$, 99.995%. Remember that we used an uninformative flat prior, and for a more informed attacker these numbers could be even larger.

In a more realistic scenario, even without any privacy protection, this high accuracy is not likely to be achieved by the attacker. So such guarantee is hardly better than no guarantee, and cannot be seen as reassuring. Thus, we want to encourage the discussion and search for more meaningful privacy definitions or DP relaxations for machine learning and federated learning. One such relaxation we present and explore in this paper.

IV. BAYESIAN DIFFERENTIAL PRIVACY

In this section, we describe *Bayesian differential privacy (BDP)*, accompanied by a practical privacy loss accounting method. We restate just the main results necessary for the following section, while all the details, proofs and experimental evaluation of BDP can be found in [12].

Definition 5. *A randomised function (algorithm) $\mathcal{A} : \mathcal{D} \to \mathcal{R}$ with domain $\mathcal{D}$ and range $\mathcal{R}$ satisfies (ε, δ)-Bayesian differential privacy if for any two adjacent datasets $D, D' \in \mathcal{D}$, differing in a single data point $x' \sim p(x)$, and for any set of outcomes $\mathcal{S} \subset \mathcal{R}$ the following holds:*

$$\Pr\left[\mathcal{A}(D) \in \mathcal{S}\right] \le e^{\varepsilon} \Pr\left[\mathcal{A}(D') \in \mathcal{S}\right] + \delta, \tag{2}$$

To derive tighter sequential composition, we use an alternative definition that implies the above:

$$\Pr[L(w, D, D') \ge \varepsilon] \le \delta, \tag{3}$$

where probability is taken over the randomness of the outcome w and the additional example x'.

This definition is very close to the original definition of DP, except that it also takes into account the randomness of x'. Hence, the basic properties are similar to those of DP, although BDP does not provide guarantees in all the scenarios where DP does (e.g. when the distribution $p(x)$ is non-stationary, it is possible that BDP underestimates the actual privacy loss).

While Definition 5 does not specify the distribution of any point in the dataset other than the additional example x', it is natural and convenient to assume that all examples in the dataset are drawn from the same distribution $p(x)$. This holds in many real-world applications, including all applications evaluated in this paper, and it allows using sampling techniques instead of requiring knowing the true distribution.

We also assume that all data points are exchangeable [34], i.e. any permutation of data points has the same joint probability. It enables tighter accounting for iterative mechanisms, and is naturally satisfied in the considered scenarios.

Since basic composition is not enough to provide tight privacy bounds for iterative federated learning algorithms in which we are interested, let us present two theorems, generalising upon the moments accountant routine. We use it as a foundation for the privacy accounting framework for federated learning presented in the next section.

Theorem 1 (Advanced Composition). *Let a learning algorithm run for T iterations. Denote by $w^{(1)} \ldots w^{(T)}$ a sequence of private learning outcomes obtained at iterations $1, \ldots, T$, and $L^{(1:T)}$ the corresponding total privacy loss. Then,*

$$\mathbb{E}\left[e^{\lambda L^{(1:T)}}\right] = \prod_{t=1}^{T} \mathbb{E}_x\left[e^{\lambda D_{\lambda+1}(p_t \| q_t)}\right],$$

where $p_t = p(w^{(t)}|w^{(t-1)}, D)$, $q_t = p(w^{(t)}|w^{(t-1)}, D')$, and $D_{\lambda+1}(p_t\|q_t)$ is Rényi divergence between p_t and q_t.

We denote the logarithm of the quantity inside the product in Theorem 1 as $c_t(\lambda)$ and call it the *privacy cost* of the iteration, or communication round, t:

$$c_t(\lambda) = \log \mathbb{E}\left[e^{\lambda D_{\lambda+1}(p_t\|q_t)}\right] \tag{4}$$

The privacy cost of the whole learning process is then a sum of the costs of each iteration.

Theorem 2. *Let the algorithm produce a sequence of private learning outcomes $w^{(1)} \ldots w^{(T)}$ using a known probability distribution $p(w^{(t)}|w^{(t-1)}, D)$. Then, ε and δ are related as*

$$\log \delta \le \sum_{t=1}^{T} c_t(\lambda) - \lambda\varepsilon.$$

Corollary 1. *Under the conditions above, for a fixed δ:*

$$\varepsilon \le \frac{1}{\lambda} \sum_{t=1}^{T} c_t(\lambda) - \frac{1}{\lambda} \log \delta.$$

Theorem 2 provides an efficient privacy accounting algorithm. During training, we compute the privacy cost $c_t(\lambda)$ for each iteration t, accumulate it, and then use to compute ε, δ pair. This process is ideologically close to that of the moment accountant, but accumulates a different quantity (note the change from the privacy loss random variable to Rényi divergence and expectation over data).

Computing $c_t(\lambda)$ precisely requires access to the prior distribution of data $p(x)$, which is unrealistic. Therefore, we need an estimator for $\mathbb{E}[e^{\lambda D_{\lambda+1}(p_t\|q_t)}]$. Moreover, since Chernoff bound, which our Theorem 2 is based on, only holds for the true expectation value, we have to take into account the estimator error. To solve this, we employ a Bayesian view of the estimation problem [35] and use the upper confidence bound of the expectation estimator.

2590

Let us define the following m-sample estimator of $c_t(\lambda)$:

$$\hat{c}_t(\lambda) = \log\left[M(t) + \frac{F^{-1}(1-\delta', m-1)}{\sqrt{m-1}} S(t)\right], \quad (5)$$

where $M(t)$ and $S(t)$ are the sample mean and the sample standard deviation of $e^{\lambda \hat{D}_{\lambda+1}^{(t)}}$, $F^{-1}(1-\delta', m-1)$ is the inverse of the Student's t-distribution CDF at $1-\delta'$ with $m-1$ degrees of freedom, and

$$\hat{D}_{\lambda+1}^{(t)} = \max\left\{D_{\lambda+1}(\hat{p}_t \| \hat{q}_t), \ D_{\lambda+1}(\hat{q}_t \| \hat{p}_t)\right\},$$
$$\hat{p}_t = p(w^{(t)} \mid w^{(t-1)}, B^{(t)}),$$
$$\hat{q}_t = p(w^{(t)} \mid w^{(t-1)}, B^{(t)} \setminus \{x_i\}).$$

One can show that, for continuous distributions $\hat{p}_t$ and $\hat{q}_t$, $\hat{c}_t(\lambda)$ overestimates the true privacy cost $c_t(\lambda)$ with probability $1-\delta'$. Therefore, the probability of underestimation δ' can be fixed upfront and incorporated in δ.

Remark 1. This step changes the interpretation of δ in Bayesian differential privacy compared to the traditional DP. Apart from the probability of the privacy loss exceeding ε, e.g. in the tails of its distribution, it also incorporates our uncertainty about the true data distribution (in other words, the probability of underestimating the true expectation because of not observing enough data samples). It can be intuitively understood as accounting for unlikely or unobserved data in δ, rather than in ε by adding more noise.

Remark 2. For discrete outcomes, a different estimator needs to be derived to meet the same bound. The process of obtaining it is identical to the one above, with the only change in the maximum entropy distribution.

Remark 3. There are other differences compared to the classic DP, such as allowance for unbounded sensitivity or estimator privacy, that are not discussed in this work. More details can be found in [12].

Gaussian Noise Mechanism. Consider the subsampled Gaussian noise mechanism [15]. The outcome distribution $p(w^{(t)} \mid w^{(t-1)}, D)$ in this case is the mixture of two Gaussians $(1-q)\mathcal{N}(g_t, \sigma^2) + q\mathcal{N}(g_t', \sigma^2)$, where g_t and g_t' are non-private outcomes at the iteration t (e.g. gradients), σ is the noise parameter, and q is the data sampling probability. Plugging the outcome distribution into the formula for Rényi divergence, we get the following result for the privacy cost.

Theorem 3. *Given the Gaussian noise mechanism with the noise parameter σ and subsampling probability q, the privacy cost for $\lambda \in \mathbb{N}$ at iteration t can be expressed as*

$$c_t(\lambda) = \max\{c_t^L(\lambda), c_t^R(\lambda)\},$$

where

$$c_t^L(\lambda) = \log \mathbb{E}_x\left[\mathbb{E}_{k \sim B(\lambda+1, q)}\left[e^{\frac{k^2-k}{2\sigma^2}\|g_t - g_t'\|^2}\right]\right],$$
$$c_t^R(\lambda) = \log \mathbb{E}_x\left[\mathbb{E}_{k \sim B(\lambda, q)}\left[e^{\frac{k^2+k}{2\sigma^2}\|g_t - g_t'\|^2}\right]\right],$$

and $B(\lambda, q)$ is the binomial distribution with λ experiments and the probability of success q.

V. FEDERATED LEARNING WITH BAYESIAN DIFFERENTIAL PRIVACY

In this section, we adapt the Bayesian differential privacy framework and its accounting method to guarantee the client-level privacy, the level most frequently addressed in the literature. We then justify and explore the instance-level privacy and two different techniques for accounting it. Finally, we propose a method to jointly account instance-level and client-level privacy for the `FedSGD` algorithm in order to provide the strongest trade-off between utility and privacy guarantees.

A. Client privacy

When it comes to reinforcing federated learning with differential privacy, the foremost attention is given to the client-level privacy [10], [11]. The goal is to hide the presence of a single user, or to be more specific, to bound the influence of any single user on the learning outcome distribution (i.e. the distribution of the model parameters).

Under the classic DP [10], [11], the privacy is enforced by clipping all user updates u_i to a fixed $L2$-norm threshold C and then adding Gaussian noise with the variance $C^2\sigma^2$. The noise parameter σ is calibrated to bound the privacy loss in each communication round, and then the privacy loss is accumulated across the rounds using the moments accountant [7].

We use the same privacy mechanism, but employ the Bayesian accounting method instead of the moments accountant. Intuitively, our accounting method should have a significant advantage over the moments accountant in the settings where data is distributed similarly across the users because in this case their updates would be in a strong agreement. In order to map the Bayesian differential privacy framework to this setting, let us introduce some notation.

Let N denote the number of clients in the federated learning system. Every client i computes and sends to the server a model update $u_i \sim p_i(u)$ drawn from the client's update distribution $p_i(u)$. Considering individual client distributions ensures that our approach is applicable to non-i.i.d. settings that are natural in the federated learning context. Generally, not all users participate in a given communication round. We denote the probability of a user i participating in the round by α_i. Thus, the overall update distribution is given by a mixture:

$$p(u) = \sum_{i=1}^{N} \alpha_i p_i(u). \quad (6)$$

In our experiments, we fix $\alpha_1 = \alpha_2 = \ldots = \alpha_N = \alpha$.

To match the notation above, let w_t indicate the privacy-preserving model update:

$$w_t = \mathcal{A}(\{u_i | u_i \in \mathbb{U}^{(t)}\}), \quad (7)$$

where $\mathcal{A}(\{u_i | u_i \in \mathbb{U}^{(t)}\}) = \frac{1}{|\mathbb{U}^{(t)}|}\sum_i u_i + \mathcal{N}(0, C^2\sigma^2)$ in the case of Gaussian mechanism, and $\mathbb{U}^{(t)}$ is the set of updates from users participating in the round t.

To bound ε and δ of Bayesian differential privacy, one needs to compute $c_t(\lambda) = \max\{c_t^L(\lambda), c_t^R(\lambda)\}$, where

$$c_t^L(\lambda) = \log \mathbb{E}_u \left[e^{\lambda D_{\lambda+1}(p_t \| q_t)} \right],$$

$$c_t^R(\lambda) = \log \mathbb{E}_u \left[e^{\lambda D_{\lambda+1}(q_t \| p_t)} \right],$$

and

$$p_t = p(w^{(t)} | w^{(t-1)}, \mathbb{U}^{(t)})$$
$$q_t = p(w^{(t)} | w^{(t-1)}, \mathbb{U}^{(t)} \setminus \{u\})$$

Since the randomness of w comes from the subsampled Gaussian noise mechanism, we use Theorem 3, in combination with user sampling and the estimator (5) for both expressions, to obtain $\hat{c}_t(\lambda)$ that upper-bounds $c_t(\lambda)$ with high probability.

Finally, we use Theorems 1 and 2 to compute ε and δ. The required assumption of exchangeability is naturally satisfied because users are sampled independently and uniformly.

B. Instance privacy

As noted above, the same privacy mechanism can be used in conjunction with the moments accountant to get the classic DP guarantees [10], [11]. In this case, (ε, δ)-DP at the client level implies the same guarantee at the instance level (i.e. bounding the influence of a single data point). However, it does not hold for BDP. Moreover, the same privacy guarantee may not be meaningful at the instance level. For example, $\delta = 10^{-3}$ might be reasonable for 100 clients, but if a client has tens of thousands of data points, it is not a reasonable failure probability at the data point level.

At the same time, instance privacy is extremely important in some scenarios. Imagine federated training on medical data from different hospitals: while a hospital participation may be public knowledge, individual patients data must be protected at the highest degree. Another reason for considering instance-level privacy is that it provides an additional layer of protection for users in case of an untrusted curator.

In order to get tighter instance privacy guarantees, we apply the subsampled Gaussian noise mechanism to gradient computation on user devices. The accounting follows the same procedure as described above, except that the noise parameter σ and the sampling probability q may be different, depending on which of the settings described below is used.

There are two possible accounting schemes:

1) Sequential accounting: Part of the accounting is performed locally on user devices and part on the server. Overall privacy cost is equivalent to the centralised training with the data sampling probability $q = \frac{B_i}{N}$, where N is the total number of data points across all users, and B_i is the local batch size.

The process proceeds as follows. At each communication round, the server sends N to participating clients, every client performs private gradient updates, computes $\hat{c}_t(\lambda)$, and sends it to the server. The server then aggregates the sum of $\hat{c}_t(\lambda)$ from all users. Since the privacy costs are data-dependent, it is possible to use secure multi-party computation to allow the server know the sum without learning individual costs.

The disadvantage of this method is that every participating client learns the total number of data points, and especially in the settings with a small number of users it may not be desirable. Furthermore, the obtained bounds apply to the commonly learnt model but not to the individual updates of each user, requiring them to maintain a separate local bound. These issues are addressed by parallel accounting.

2) Parallel accounting: In this scheme, every client computes $\hat{c}_t(\lambda)$ using $q = \frac{B_i}{N_i}$, where N_i is the local dataset size of the client. Consequently, since $N_i \leq N$, the privacy costs will be higher. But this is compensated by using parallel composition instead of sequential: the server aggregates the maximum of $\hat{c}_t(\lambda)$ over all users. Again, using secure multi-party computation is possible to prevent the server from learning individual privacy costs.

Parallel composition is applicable in this scenario because user updates within a single round are independent. However, the server needs to sum up maximum privacy costs over the rounds because updates are dependent on previous rounds.

As we show in Section VI, parallel accounting may require more communication rounds to converge to the same quality solution with the same privacy guarantee. The gap is more notable on non-identically distributed data.

C. Joint privacy

Instance privacy provides tighter and more meaningful guarantees for every data point contribution to the trained model. Nevertheless, there is a downside: adding noise both during on-device gradient descent as well as during the averaging phase on the server results in slow convergence or complete divergence of the federated learning algorithm.

To tackle this problem, we propose *joint accounting*, where the noise added on the client side is re-counted towards the client-level privacy guarantee.

The main idea of joint accounting is that a client update received by the server is already noisy when instance privacy is enforced, and instead of adding more noise the server can re-count existing noise to compute the client-level bound[2]. The only problem: the server cannot sample non-private client updates to estimate $c_t(\lambda)$ because it no longer has access to their distribution.

Fortunately, the inner expectation in $c_t^L(\lambda)$ and $c_t^R(\lambda)$ can be computed locally, suggesting the following procedure. Every client computes $\hat{D}_{\lambda+1}^{(t)}$ (Eq. 5) with $\hat{p}_t$ and $\hat{q}_t$ being the private outcome distributions with and without their entire update. Then, the server computes $M(t)$, $S(t)$, and $\hat{c}_t(\lambda)$ by simple averaging. Additionally, one can implement this averaging step with secure multi-party computation to further privacy protection. For the moment, however, it can only be used with `FedSGD`, and not `FedAvg`, because every noisy step in `FedAvg` would change the point at which the gradient is computed, potentially leading to a different gradient distribution or underestimated total noise variance.

---

[2]A version of this technique for differential privacy has also been explored in a master's thesis project with Nikolaos Tatarakis.

Using joint accounting allows to achieve tight instance and client privacy guarantees, and at the same time, preserve the speed of convergence almost at the same level as the client-privacy-only solution (see Section VI-D).

VI. EVALUATION

In this section, we provide results of the experimental evaluation of our approach. We begin by describing the datasets we used, as well as the setting details shared by all experiments. The subsequent structure follows that of the previous section. We first evaluate the client-level privacy by comparing accuracy and privacy guarantees of the traditional DP method [11] to ours (Section VI-B). Then, in Section VI-C, we perform experiments on the two proposed methods of instance privacy accounting. Finally, Section VI-D describes the results of the joint accounting approach.

A. Experimental setup

We perform experiments on two datasets. The first dataset is MNIST [36]. It is a standard image classification task widely used in machine learning research. More specifically, it is a handwritten digit recognition dataset consisting of 60000 training examples and 10000 test examples, each example is a 28x28 greyscale image. The second is the APTOS 2019 Blindness Detection challenge dataset[3] (in figures, tables and text, we refer to this dataset as *Retina* or *APTOS*). It consists of 3662 retina images taken using fundus photography. The images are labelled by clinicians to reflect the severity of diabetic retinopathy on the scale from 0 to 4. Unlike other datasets commonly evaluated in the privacy literature [7], [10], [11], this one actually has more serious implications of a privacy leak.

All experiments have the following general setup. There is a number of clients (100, 1000, or 10000), each holding a subset of data, and the server that coordinates federated training of the shared model. Some setups with a higher number of users will entail repetition of data, like in [11], which is a natural scenario in some applications, e.g. shared or very similar images on different smartphones. In MNIST experiments, each user holds 600 examples. For the APTOS dataset, we use data augmentation techniques (e.g. random cropping, resizing, etc.) to obtain a larger training set, and then split it such that every client gets ∼350 images.

Testing is performed on the official test split for MNIST, and on the first 500 samples in case of APTOS.

While the parameters of the training vary based on experiments, the models remain the same. For MNIST, we use a simple CNN with two convolutional layers and two fully connected layers (similar to the one described in the TensorFlow tutorial[4]). In case of APTOS, due to the small dataset size and a harder learning task, we employ ResNet-50 [37] pre-trained on ImageNet [38] and re-train only the last fully-connected layer of the network. We do not do extensive

TABLE I: Accuracy and privacy guarantees (reported as a pair (ε, δ)) on MNIST, non-i.i.d. setting.

Clients	Accuracy			Privacy	
	Baseline	DP	BDP	DP	BDP
100	97%	78%	**88**%	$(8, 10^{-3})$	$(\mathbf{4.0, 10^{-3}})$
1K	98%	95%	**96**%	$(3, 10^{-5})$	$(\mathbf{1.5, 10^{-5}})$
10K	99%	96%	**97**%	$(1, 10^{-6})$	$(\mathbf{0.6, 10^{-6}})$

TABLE II: Accuracy and privacy guarantees (reported as a pair (ε, δ)) on MNIST, i.i.d. setting.

Clients	Accuracy			Privacy	
	Baseline	DP	BDP	DP	BDP
100	97%	86%	**92**%	$(8, 10^{-3})$	$(\mathbf{2.0, 10^{-3}})$
1K	98%	97%	97%	$(3, 10^{-5})$	$(\mathbf{1.0, 10^{-5}})$
10K	99%	97%	**98**%	$(1, 10^{-6})$	$(\mathbf{0.5, 10^{-6}})$

hyper-parameter tuning in general, since we are interested in relative performance of private models compared to non-private ones rather than the best classification accuracy, and thus, our non-private baseline results may not always match the ones reported in [3]. For the same reason, we restrict the number of communication rounds (≤ 300) and use `FedSGD` instead of `FedAvg`, although all the methods except for joint accounting are compatible with `FedAvg`.

One of the important aspects of federated learning is that data might not be distributed identically among users. In agreement with previous work [3], [11], we include experiments in both *i.i.d.* and *non-i.i.d.* settings for MNIST, because it allows for a natural non-identical split. More specifically, in the i.i.d. setting, every user is assigned a subset of uniformly sampled examples. In the non-i.i.d. setting, we follow the same scheme as [3] and [11]: splitting the dataset on shards of 300 points within the same class and then assigning 2 random shards to each client. The scenario of 100 clients with non-identically distributed data is particularly hard for privacy applications: there are $\binom{10}{2} = 45$ possible digit combinations that clients can hold and only 100 clients, meaning that some clients might be easily distinguishable by their data distribution. Therefore, it is important to note that it may not be possible to obtain a reasonable privacy bound in this scenario without seriously compromising accuracy.

The privacy accounting is performed by two methods. To obtain the bounds on ε and δ of differential privacy, we use the moments accountant [7], the state-of-the-art DP accounting method. In the case of Bayesian differential privacy, we follow the technique described in Sections IV and V: we sample a number of user updates (or gradients for instance privacy), estimate an upper bound on the privacy cost, and use Chernoff inequality to compute the corresponding pair of ε, δ.

B. Client privacy

In this experiment, we test adding client privacy the same way it is done in [10] and [11]. We use Bayesian accounting, as described in Section V-A, and compare it to the classic

(a) APTOS 2019   (b) MNIST, i.i.d.   (c) MNIST, non-i.i.d.

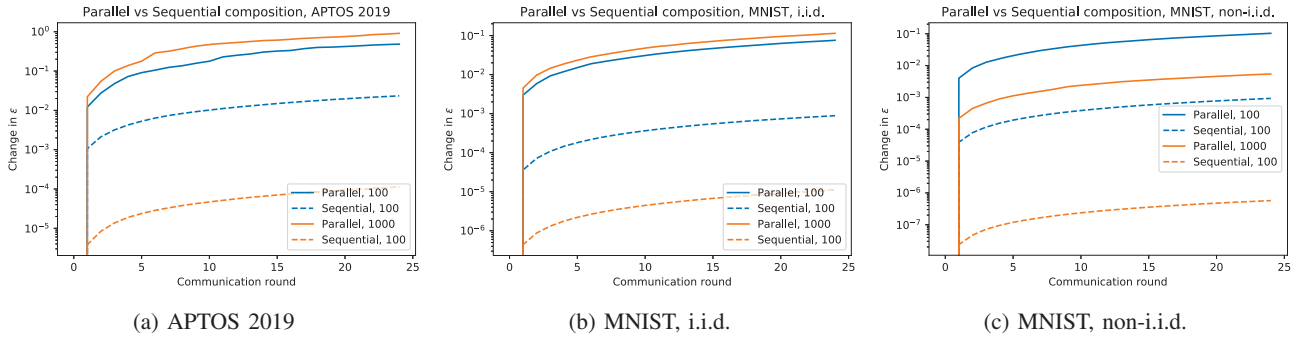Fig. 1: Change in ε relative to its initial value for parallel and sequential composition modes of instance privacy in the settings of 100 and 1000 clients.

TABLE III: Accuracy and privacy guarantees (reported as a pair (ε, δ)) on APTOS 2019, i.i.d. setting.

Clients	Accuracy			Privacy	
	Baseline	DP	BDP	DP	BDP
100	70%	60%	**65**%	$(8, 10^{-3})$	$\mathbf{(2.1, 10^{-3})}$
1K	71%	67%	**68**%	$(2, 10^{-5})$	$\mathbf{(0.5, 10^{-5})}$
10K	72%	68%	**69**%	$(1, 10^{-6})$	$\mathbf{(0.2, 10^{-6})}$

TABLE IV: Accuracy and privacy guarantees (a pair (ε, δ)), at instance and client levels, using joint privacy accounting in the setting of 100 clients.

Dataset	Accuracy			Privacy	
	Baseline	DP	BDP	Client	Instance
APTOS 2019	70%	42%	**64**%	$(1, 10^{-3})$	
MNIST (iid)	97%	15%	**74**%	$(2, 10^{-3})$	$(0.1, 10^{-5})$
MNIST (non-iid)	97%	12%	**62**%	$(4, 10^{-3})$	

differential privacy accounting by the moments accountant [7]. We fix the noise level σ and account DP and BDP in parallel.

Tables I, II, III summarise accuracy and privacy guarantees obtained in this setting for MNIST (non-i.i.d. and i.i.d.) and APTOS respectively. The first column indicates the number of clients, the second—the baseline accuracy of a non-private federated classifier (models described in the previous section). The following columns contain accuracy and privacy parameters obtained for private models using the classic DP and BDP. Despite being trained in parallel, the two techniques may differ in accuracy because in some cases we do early stopping for DP to prevent exceeding privacy budget.

In all cases and for all datasets, we observe substantial benefits of using Bayesian accounting. The accuracy gains are most notable in the non-i.i.d. setting of MNIST, where our method can achieve 10% higher accuracy in the 100 clients setting, because it presents a more difficult learning scenario as explained in the previous section. The privacy gains are consistently significant across all datasets and settings, and taking into account the fact that ε is exponentiated to get the bound on outcome probability ratios, BDP can reach $e^8/e^2 \approx 400$ times stronger guarantee. Nevertheless, in the settings with few clients, even Bayesian differential privacy does not reach a more comfortable guarantee of $\varepsilon = 1$, suggesting that a better privacy-accuracy trade-off may not be feasible due to higher clients identifiability, or that more work is needed in improving training with noise and developing novel privacy mechanisms for federated learning.

Importantly, there is no computation or communication overhead from the users' point of view in these experiments since the privacy accounting code is executed on the server.

C. Instance privacy

As noted in Section V-B, instance privacy is very important in scenarios like collecting medical data from a number of hospitals where patient privacy is at least as crucial as hospital privacy. In this section, we compare two accounting methods proposed earlier: *sequential* and *parallel* accounting.

Depicted in Figure 1 are the curves showing the growth of ε estimate with communication rounds. We subtracted the initial value and applied logarithmic scale in order to better show the difference in the rate of growth. Across all settings, it can be seen that parallel accounting leads to faster growth rates, despite the fact that the parallel composition is more efficient (taking maximum over clients instead of a sum). This behaviour can be explained by the fact that each client is unaware of the total dataset size and, having a small number of data points, is convinced that every data example has significant influence on the outcome. The unawareness about other clients in the case of parallel accounting can also explain the fact that we don't observe any improvement in the ε growth rate with increasing the number of clients. The only exception is the non-i.i.d. MNIST experiment, where the difference is likely to come from increased stability of training and decreased gradient variability with more clients.

The main takeaway from this experiment is that it is beneficial to use sequential accounting for privacy of the federated model whenever communicating the total size of the dataset to users is acceptable. In other cases, and for personal privacy accounting in case of the untrusted curator, parallel accounting can be used, but more noise is necessary for reasonable privacy guarantees.

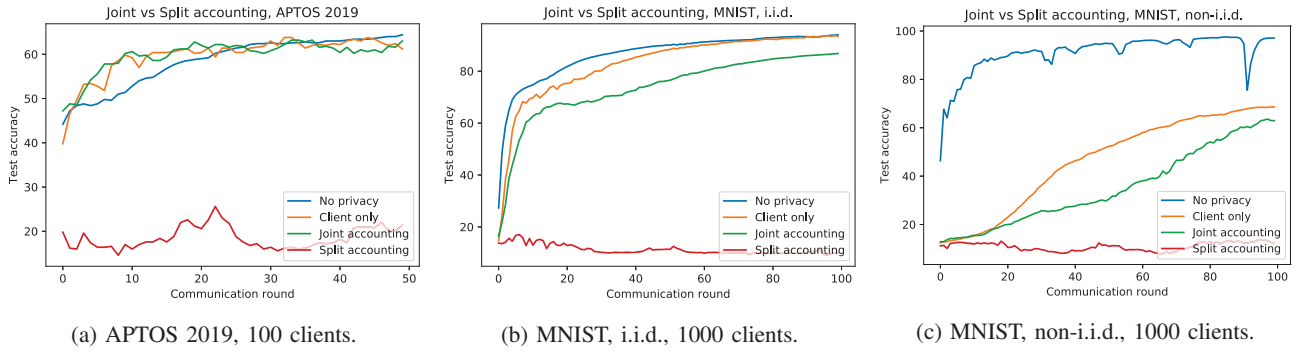| (a) APTOS 2019, 100 clients. | (b) MNIST, i.i.d., 1000 clients. | (c) MNIST, non-i.i.d., 1000 clients. |

Fig. 2: Test accuracy as a function of a communication round for non-private, client-level-only private, and jointly private (using either joint or separate accounting) scenarios.

D. Joint privacy

Lastly, we would like to test the proposed method of the joint accounting of instance-level and client-level privacy and contrast it to accounting at these two levels separately. We perform experiments in the same settings as above, fixing the client privacy at a certain level ($\varepsilon = 1$) and evaluating the speed and quality of training. We also compare to what can be achieved by introducing privacy only at the client level.

Figure 2 displays the test accuracy evolution over communication rounds in the setup of 100 clients for APTOS and 1000 clients for MNIST. The graphs contain curves for training without privacy, client-level-only privacy, and the two accounting paradigms: *joint* and *split*. As expected, the non-private training achieves the best accuracy. Nevertheless, in the i.i.d. setting, client-only private training quickly approaches non-private training in quality. Notably, training with both instance and client privacy using the joint accounting performs nearly as well, while training with the separate accounting completely fails due to excessive amounts of noise at both instance and client levels. For the non-i.i.d. setting, private training is slower, but there is little difference between introducing privacy only at the client level and using the joint accounting at both levels: after a slightly larger number of rounds, training with the joint accounting reaches similar performance. Based on these experiments, we conclude that by using joint accounting we can introduce instance privacy on clients and get client-level privacy at almost no cost.

Finally, we evaluate our method in the strong privacy setting. We set the instance-level privacy to $\varepsilon = 0.1$ and stop training when the client privacy reaches the level similar to previous experiments (except APTOS dataset, where we were able to achieve comparable results with lower privacy cost), and report the accuracy that can be achieved in this strict setting. We have also chosen the most difficult scenario of 100 clients. As seen in Table IV, the algorithm with differential privacy performs very poorly on APTOS dataset, and fails to learn on MNIST, in both i.i.d. and non-i.i.d. setting. Its performance is especially affected by the strict instance privacy requirement, since such low levels of ε necessitate large quantities of noise to be added. It is worth noting, that

it might be possible to achieve better results with DP by performing per-example gradient clipping, as in [7], but we do not use this technique due to its impracticality.

On the other hand, our approach manages to achieve reasonable accuracy even under such a strong privacy guarantee. On APTOS dataset, it is just 6% lower than the non-private baseline, while on MNIST, it correctly classifies more than 70% of the test data in the i.i.d. setting and over 60% in the non-i.i.d. setting. One could potentially add more noise on the server and combine the accounting with the instance level noise to slow down the growth of ε and reach even better performance, but we leave these experiments for future work.

Both instance and joint privacy accounting add some computation overhead on user devices due to multiple gradient calculations. However, performing FL routines when devices are idle and charging, as suggested in [39], alleviates this problem. Communication overhead is negligible because only a single floating point number is added to user messages.

VII. CONCLUSION

We employ the notion of (ε, δ)-Bayesian differential privacy, a relaxation of (ε, δ)-differential privacy, to obtain tighter privacy guarantees for clients in the federated learning settings. The main idea of this approach is to utilise the fact that users come from a certain population with similarly distributed data, and therefore, their updates will likely be in agreement with each other. This is a meaningful assumption in many machine learning scenarios because they target a specific type of data (e.g. medical images, emails, motion sensor data, etc.). For example, it may be unjustified to try hiding an absence of an audio record in a training set for the ECG analysis, since the probability of it appearing is in fact much smaller than δ.

We adapt an efficient and tight privacy accounting method for Bayesian differential privacy to the federated setting in order to estimate client privacy guarantees. Moreover, we emphasise the importance of instance-level privacy and propose two variants of privacy accounting at this level. Finally, we introduce a novel technique of joint accounting suitable for obtaining privacy guarantees at instance and client levels jointly from only instance-level noise.

2595

Our evaluation provides evidence that Bayesian differential privacy is more appropriate for federated learning. First, it requires significantly less noise to reach the same privacy guarantees, allowing models to train in fewer communication rounds. Second, the bounds on privacy budget ε are much tighter, and thus, more meaningful. When the number of clients reaches an order of thousands, which is realistic in many federated learning scenarios, ε can be kept below 1. Finally, we demonstrate that by using joint accounting we can get client privacy for free when adding instance privacy. This way, the privacy budget can be kept close to $\varepsilon = 1$ for client privacy and $\varepsilon = 0.1$ for instance privacy while maintaining reasonably high accuracy.

An important future direction of research is automatically detecting and mitigating scenarios in which tighter privacy guarantees are inapplicable, such as non-stationary data distributions or datasets with non-exchangeable samples.

REFERENCES

[1] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2015, pp. 1322–1333.

[2] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *Security and Privacy (SP), 2017 IEEE Symposium on*. IEEE, 2017, pp. 3–18.

[3] H. B. McMahan, E. Moore, D. Ramage, S. Hampson *et al.*, "Communication-efficient learning of deep networks from decentralized data," *arXiv preprint arXiv:1602.05629*, 2016.

[4] C. Dwork, "Differential privacy," in *33rd International Colloquium on Automata, Languages and Programming, part II (ICALP 2006)*, vol. 4052. Venice, Italy: Springer Verlag, July 2006, pp. 1–12. [Online]. Available: https://www.microsoft.com/en-us/research/publication/differential-privacy/

[5] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, "Differentially private empirical risk minimization," *Journal of Machine Learning Research*, vol. 12, no. Mar, pp. 1069–1109, 2011.

[6] S. Song, K. Chaudhuri, and A. D. Sarwate, "Stochastic gradient descent with differentially private updates," in *Global Conference on Signal and Information Processing (GlobalSIP), 2013 IEEE*. IEEE, 2013, pp. 245–248.

[7] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016, pp. 308–318.

[8] N. Papernot, M. Abadi, U. Erlingsson, I. Goodfellow, and K. Talwar, "Semi-supervised knowledge transfer for deep learning from private training data," *arXiv preprint arXiv:1610.05755*, 2016.

[9] N. Papernot, S. Song, I. Mironov, A. Raghunathan, K. Talwar, and Ú. Erlingsson, "Scalable private learning with pate," *arXiv preprint arXiv:1802.08908*, 2018.

[10] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning differentially private recurrent language models," *arXiv preprint arXiv:1710.06963*, 2017.

[11] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," *arXiv preprint arXiv:1712.07557*, 2017.

[12] A. Triastcyn and B. Faltings, "Bayesian differential privacy for machine learning," *arXiv preprint arXiv:1901.09697*, 2019.

[13] B. Hitaj, G. Ateniese, and F. Pérez-Cruz, "Deep models under the gan: information leakage from collaborative deep learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2017, pp. 603–618.

[14] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of cryptography conference*. Springer, 2006, pp. 265–284.

[15] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[16] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. ACM, 2015, pp. 1310–1321.

[17] A. C.-C. Yao, "Protocols for secure computations," in *FOCS*, vol. 82, 1982, pp. 160–164.

[18] I. Mironov, O. Pandey, O. Reingold, and S. Vadhan, "Computational differential privacy," in *Annual International Cryptology Conference*. Springer, 2009, pp. 126–142.

[19] D. J. Mir, "Information-theoretic foundations of differential privacy," in *International Symposium on Foundations and Practice of Security*. Springer, 2012, pp. 374–381.

[20] W. Wang, L. Ying, and J. Zhang, "On the relation between identifiability, differential privacy, and mutual-information privacy," *IEEE Transactions on Information Theory*, vol. 62, no. 9, pp. 5018–5029, 2016.

[21] C. Dwork and G. N. Rothblum, "Concentrated differential privacy," *arXiv preprint arXiv:1603.01887*, 2016.

[22] M. Bun and T. Steinke, "Concentrated differential privacy: Simplifications, extensions, and lower bounds," in *Theory of Cryptography Conference*. Springer, 2016, pp. 635–658.

[23] M. Bun, C. Dwork, G. N. Rothblum, and T. Steinke, "Composable and versatile privacy via truncated cdp," in *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*. ACM, 2018, pp. 74–86.

[24] I. Mironov, "Renyi differential privacy," in *Computer Security Foundations Symposium (CSF), 2017 IEEE 30th*. IEEE, 2017, pp. 263–275.

[25] J. M. Abowd, M. J. Schneider, and L. Vilhuber, "Differential privacy applications to bayesian and linear mixed model estimation," *Journal of Privacy and Confidentiality*, vol. 5, no. 1, p. 4, 2013.

[26] M. J. Schneider and J. M. Abowd, "A new method for protecting interrelated time series with bayesian prior distributions and synthetic data," *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, vol. 178, no. 4, pp. 963–975, 2015.

[27] A.-S. Charest and Y. Hou, "On the meaning and limits of empirical differential privacy," *Journal of Privacy and Confidentiality*, vol. 7, no. 3, p. 3, 2017.

[28] A. Triastcyn and B. Faltings, "Generating artificial data for private deep learning," in *Proceedings of the PAL: Privacy-Enhancing Artificial Intelligence and Language Technologies, AAAI Spring Symposium Series*, ser. CEUR Workshop Proceedings, no. 2335, 2019, pp. 33–40. [Online]. Available: http://ceur-ws.org/Vol-2335/1st_PAL_paper_7.pdf

[29] J. Geumlek, S. Song, and K. Chaudhuri, "Renyi differential privacy mechanisms for posterior sampling," in *Advances in Neural Information Processing Systems*, 2017, pp. 5289–5298.

[30] M. Gil, F. Alajaji, and T. Linder, "Rényi divergence measures for commonly used univariate continuous distributions," *Information Sciences*, vol. 249, pp. 124–131, 2013.

[31] T. Van Erven and P. Harremos, "Rényi divergence and kullback-leibler divergence," *IEEE Transactions on Information Theory*, vol. 60, no. 7, pp. 3797–3820, 2014.

[32] B. Balle, G. Barthe, and M. Gaboardi, "Privacy amplification by subsampling: Tight analyses via couplings and divergences," in *Advances in Neural Information Processing Systems*, 2018, pp. 6280–6290.

[33] M. Bun, "A teaser for differential privacy," 2017.

[34] D. J. Aldous, "Exchangeability and related topics," in *École d'Été de Probabilités de Saint-Flour XIII—1983*. Springer, 1985, pp. 1–198.

[35] T. E. Oliphant, "A bayesian perspective on estimating mean, variance, and standard-deviation from data," 2006.

[36] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[37] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[38] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[39] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konecny, S. Mazzocchi, H. B. McMahan *et al.*, "Towards federated learning at scale: System design," *arXiv preprint arXiv:1902.01046*, 2019.