Bottlenecked Backpropagation to Train Differentially Private Deep Neural Networks

Arghyadeep Ghosh [a,*] **and Mrinal Das** [b,**]

[a,b] Indian Institute of Technology Palakkad

Abstract. Deep neural networks often tend to memorize data and can risk information leakage if trained on private data. There has been some attempts to train deep neural networks by contaminating the gradients during backpropagation. Over the years, this has become one of the most prominent techniques to make neural networks differentially private. One downside of this method is that contaminated gradients lead to suboptimal solutions during backpropagation. In this paper, we make an attempt to diminish the contamination effect by proposing a bottlenecked backpropagation technique. The proposed bottlenecked backpropagation technique follows Reny differential privacy, a recently developed more optimized version in the realm of differential privacy. On the other hand, the bottlenecked backpropagation considers the direction and neglects the magnitude of the gradient vectors. The idea is built on top of signed stochastic gradient descent, another recent advancement in the optimization methods for deep learning. By prioritizing gradient direction over magnitude, it minimizes noise impact on model convergence. Experimental results on benchmarks including MNIST, FMNIST, and IMDB datasets demonstrate substantial improvements over the state-of-the-art methods, achieving faster convergence and higher model accuracy, striking a promising balance between privacy and performance. Furthermore, we observe the proposed methods to be resilient against membership inference attacks.

1 Introduction

Deep neural networks excel in real-world tasks by learning from vast datasets through backpropagation, often outperforming traditional methods. While sharing these large models benefits the research community, they are sometimes trained on private data, raising privacy concerns. Ensuring these models maintain efficiency while protecting individual privacy is crucial.

Differential privacy is a powerful theory that has evolved over last few years which can be blend into training models in such a way that it provides plausible deniability to individuals participating in the data source. This provides sufficient gurantee to any individual that their secret is safe. Our interest is in training deep neural networks in such a way that after the training is done they become differentially private. There are several attempts made in recent past towards this goal [1]. However, often such methods are either less efficient in solving the task or not sufficiently private.

In privacy aware learning often there is a natural trade-off between privacy and accuracy. Larger privacy means less contribution from data and hence less accuracy. In state of the art models, gradient is clipped by some constant upper threshold which in turn clips the contribution from data in updation of the weights. It is hard to break this trade-off that is we want larger privacy as well as larger accuracy. We aim to achieve larger accuracy for same privacy budget. By playing around with the clipping threshold we may not be able to improve much.

To train differentially private neural networks to improve the balance between privacy and accuracy we have targeted the learning process itself which is the backpropagation algorithm. Our idea stems from the simple intuition that use less from the data to risk less on the data. That is we use a bottleneck on the information used to update weights in the neural networks during backpropagation. That makes the learnt model less sensitive to the private data leading to less noisy updates, which in turn often lead to faster learning costing less cumulative privacy budget.

Our contribution is to propose the novel concept of using the direction from the gradients discarding the magnitude. This drastically reduces the imapct of data on training the deep neural nets. We refer to this mechanism as bottlenecked backpropagation. We implement bottlenecked backpropagation through stochastic gradient descend (SGD) and Adam two of the most popular optimizers to train neural networks. We have used sampled Gaussian mechanism along with Reny differential privacy to build a differentially private bottlenecked backpropagation. We have theoretically analyzed the privacy budget of the proposed algorithm. We have also experimented with three real-life datasets to show the efficacy of the proposed algorithm on the state of the art.

The paper is organized as follows: in Section 2 we cover the relevant preliminaries on differential privacy and privacy-aware learning for deep neural networks. In Section 3 we discuss the related works. In Section 4 we describe the proposed approach, and in Section 5 present our experimental results.

2 Preliminaries

In this section we briefly state the required definitions and concepts related to differential privacy (DP), Rényi differential privacy (RDP), and their use in training privacy aware neural networks.

2.1 *Differential Privacy*

Differential privacy is a precise mathematical framework that formally defines the concept of data privacy. It stipulates that the inclusion or exclusion of any single entry in the input dataset should not

[*] Corresponding Author. Email: arghyadeep.ghosh@cvv.ac.in
[**] Corresponding Author. Email: mrinal@iitpkd.ac.in

result in statistically significant alterations in the output [17, 12, 13], provided that the principles of differential privacy are upheld.

Definition 1. ***(Differential Privacy [13])*** *A randomized mechanism $\mathcal{M} : \mathcal{D} \rightarrow \mathcal{R}$ with domain $\mathcal{D}$ and range $\mathcal{R}$ satisfies (ε, δ)-Differential Privacy (DP) if, for any two neighboring datasets D and D' that differ in only a single entry, and for any subset $S \subseteq$ Range$(\mathcal{M})$:*

$$Pr(\mathcal{M}(D) \in S) < e^{\varepsilon} \times Pr(\mathcal{M}(D') \in S) + \delta. \tag{1}$$

Here, $\varepsilon > 0$ controls the level of privacy guarantee in the worst case, with a smaller ε indicating a stronger privacy level. The parameter $\delta > 0$ represents the failure probability that the property does not hold, which ideally should be negligible [35, 28]. In practice, δ is often set to be less than $\frac{1}{|D|}$.

This equation enforces an upper bound on the probability ratio of all possible outcomes from two neighboring datasets. Its purpose is to limit an adversary's confidence in distinguishing between the original dataset D and its neighbor D'. The parameters ε and δ play critical roles in quantifying the level of privacy and the associated failure probability, respectively.

Additionally, by adding random noise, differential privacy for a function $f : X^n \rightarrow \mathbb{R}^d$ can be achieved, as stated in Definition 2.1. The l-sensitivity of the function determines the amount of noise required to achieve differential privacy.

Definition 2. ***(l_k-Sensitivity [12])*** *For a function $f : X^n \rightarrow \mathbb{R}^d$, we define its l_k norm sensitivity (denoted as $\Delta_k f$) over all neighboring datasets $x, x' \in X^n$ differing in a single sample as*

$$\sup_{x,x' \in X^n} ||f(x) - f(x')||_k \leq \Delta_k f. \tag{2}$$

In this paper, we focus on l_2 sensitivity, i.e., $|| \cdot ||_2$ or the l_2 Norm. Additionally, the following Lemma 2.3 ensures the privacy guarantee of post-processing operations.

Lemma 1. ***(Post-processing [13])*** *Let $\mathcal{A}$ be a mechanism satisfying (ε, δ)-DP. Let f be a function whose input is the output of $\mathcal{A}$. Then $f(\mathcal{A})$ also satisfies (ε, δ)-DP.*

2.2 Rényi Differential Privacy

Rényi differential privacy (RDP) offers a relaxation of ε-differential privacy, utilizing Rényi divergence as its cornerstone.

Definition 3. ***(Rényi divergence [14])*** *The Rényi divergence of order $\alpha > 1$ between two probability distributions P and Q is expressed as:*

$$D_\alpha(P \parallel Q) = \frac{1}{\alpha - 1} \ln \mathbb{E}_{x \sim Q} \left[\left(\frac{P(x)}{Q(x)} \right)^{\alpha} \right], \tag{3}$$

where $\mathbb{E}_{x \sim Q}$ denotes the expected value of x under the distribution Q, and $P(x)$ and $Q(x)$ represent the densities of P and Q at x, respectively.

Definition 4. ***(Rényi Differential Privacy [24])*** *RDP is defined as follows: For any neighboring datasets $x, x' \in X^n$, a randomized mechanism $M : X^n \rightarrow R^d$ satisfies (α, τ)-RDP if:*

$$D_\alpha(M(x) \parallel M(x')) \leq \tau. \tag{4}$$

Next we state a formal definition of the Gaussian mechanism along with its associated RDP guarantee.

Definition 5. ***(RDP of Gaussian mechanism [24])*** *For a real valued function f with sensitivity μ, the Gaussian mechanism as follows*

$$G_\sigma f(D) = f(D) + \mathcal{N}\left(0, \mu^2\sigma^2\right), \tag{5}$$

satisfies $(\alpha, \alpha/2\sigma^2)$-RDP, where $\mathcal{N}(0, \mu^2\sigma^2)$ is a normally distributed random variable with standard deviation $\mu\sigma$ and mean 0.

Lemma 2. ***(Conversion from (α, τ)-RDP to (ε, δ)-DP [6]).*** *Let $\mathcal{M} : \mathcal{D} \rightarrow \mathcal{R}$ be a randomized mechanism with domain $\mathcal{D}$ and range $\mathcal{R}$. Suppose $\alpha > 1$ and $\varepsilon \geq 0$. If $\mathcal{M}$ satisfies (α, τ)-RDP, given a specific δ, $\mathcal{M}$ satisfies (ε, δ)-DP for*

$$\varepsilon = \tau + \frac{\log\left(\frac{1}{\delta}\right) + (\alpha - 1)\log\left(1 - \frac{1}{\alpha}\right) - \log(\alpha)}{\alpha - 1}. \tag{6}$$

2.3 Deep Learning With Differential Privacy

Differentially Private Stochastic Gradient Descent (DPSGD) is a popular training algorithm utilized in deep neural network training to ensure differential privacy. In each iteration, DPSGD samples a batch of data tuples from the dataset $\mathcal{D}$ with a fixed probability proportional to the batch size. After sampling, the gradient of each tuple $x_i \in B_t$ is computed with respect to the model parameters θ_i, denoted as $g_t(x_i) = \nabla_{\theta_i} \mathcal{L}(\theta_i, x_i)$, where $\mathcal{L}$ represents the loss function. DPSGD employs gradient clipping to limit the magnitude of the per-sample gradient, ensuring that it does not exceed a predetermined ℓ_2 norm bound (Equation(7). The clipped gradient $g_t(x_i)$ is computed as the original gradient $g_t(x_i)$ scaled by the ratio of the maximum allowed norm and the ℓ_2 norm of the original gradient.

$$\begin{aligned} \tilde{g}_t(x_i) &= \text{Clip}(g_t(x_i); C) \\ &= g_t(x_i) / \max\left(1, \frac{\|g_t(x_i)\|_2}{C}\right) \end{aligned} \tag{7}$$

In this way, for any two neighboring datasets, the sensitivity of the query $g(x_i)$ for $i \in B_t$ is bounded by C. Then, it adds Gaussian noise scaling with C to the sum of the gradients when computing the batch-averaged gradients:

$$\tilde{g}_t = \frac{1}{b}\left(\sum_{i \in B_t} g_t(x_i) + \mathcal{N}\left(0, \sigma^2 C^2 I\right)\right) \tag{8}$$

where σ is the noise multiplier depending on the privacy budget. Last, the gradient descent is performed based on the batch-averaged gradients. Since initial models are randomly generated and independent of the sample data, and the batch-averaged gradients satisfy differential privacy, the resulting models also satisfy differential privacy due to the post-processing property.

Three factors determine DPSGD's privacy guarantee — the noise multiplier σ, the sampling ratio $\frac{b}{|D|}$, and the number of training iterations T. In reality, given the privacy parameters (ε, δ), we can set appropriate values for these three hyperparameters to optimize the performance. The privacy calibration process is performed using a privacy accountant: a numerical algorithm providing tight upper bounds for the given (ε, δ) as a function of the hyperparameters , which in turn can be combined with numerical optimization routines to optimize one hyperparameter given the other two. In this work, we use the RDP for privacy accounting. In practice, given σ, δ, and b at each iteration, we select α from $\{2, 3, ..., 64\}$ to determine the smallest ε similar to recent practice.

3 Related Work

Privacy-preserving model training was initially proposed in the literature [34, 4]. Subsequently, a generalized algorithm known as DPSGD was introduced for deep learning with differential privacy [1]. Since then, various efforts have been made to enhance DPSGD from different perspectives.

At each iteration of training, an approximate bound on the gradient norm was derived using public data, and the gradients were subsequently clipped at this approximate bound [46]. The concept of adaptive clipping within each layer of the neural network was also suggested [37]. Additionally, a method for dynamically adjusting the clipping threshold to monitor a specified quantile of the update norm distribution during training, particularly in federated learning scenarios, was presented [2]. Furthermore, AdaCliP, which involves coordinate-wise adaptive clipping of the gradient, was proposed [31]. However, recent research has indicated that redefining the clipping equation can render clipping equivalent to normalization by setting the clipping bound sufficiently small [41]. Notably, our paper does not utilize any adaptive clipping techniques during the training phase.

The utilization of a family of bounded activation functions (tempered sigmoids), as opposed to the unbounded activation function ReLU in DPSGD, was discovered to yield favorable performance outcomes [29]. Scattering Networks were applied to pre-process images and extract features prior to DPSGD training [36]. Careful hyper-parameter tuning, coupled with group normalization and weight standardization, was amalgamated to deliver notable performance enhancements [9].

Adaptive noise addition was implemented in [30] using a hierarchical correlation propagation protocol approach, where a small amount of noise was added to features with high correlation to the model's output. An optimized Gaussian mechanism was introduced by [3], which directly calibrates variance using the Gaussian cumulative density function instead of relying on a tail-bound approximation.

The best learning rate was selected by [22] based on model evaluation, and adaptive privacy budget allocation was implemented in each round of DPSGD training. [12]adopts the NoisyMax method on loss values from a variety of models obtained through different learning rates in a single iteration.

Additionally, [40] employed the Root Mean Square Prop (RMSProp) gradient descent technique to adaptively add noise to coordinates of the gradient. Subsequently, many works have focused on reducing the dimensionality [16, 45, 44], of the model during training to mitigate the impact of noise on the overall model.

A method known as the Moments Accountant (MA) was proposed for providing an upper bound on the privacy curve of a composition of [1]. The Moments Accountant subsequently became integrated into the framework of Renyi Differential Privacy (RDP) [24]. Additionally, the notion of Gaussian Differential Privacy (GDP) based on hypothesis testing was introduced [7].

Various other variants of differential privacy exist, such as Concentrated DP (CDP) and zero Concentrated-DP [8] , each tailored for specific scenarios and convertible into one another under certain conditions. (ϵ, δ)-differential privacy remains our primary focus, as it is the most prevalent and widely adopted in both academic literature and practical applications. Moreover, several works based on local differential privacy concentrate on ϵ-differential privacy [10, 11, 42, 43, 47].

The problem of bias due to Poisson sampling in DPSGD was first explored in [18]. It was proposed, which weights the importance sampling by the gradient norm of the sample. The impact of Poisson sampling on convergence speed is mitigated through the process of resampling in a recent algorithm. This method [15] employs the validation test to select model updates, accelerating convergence and improving utility. To mitigate injected Gaussian noise, it integrates clipping and gradient thresholding. Moreover, the Gaussian mechanism with selective release reduces privacy budget consumption.

[19] addresses key challenges in FL, by introducing stochastic-sign-based gradient compressors and an error-feedback variant to enhance performance.

Algorithm 1 SignSGD [5]

```
Input: learning rate $\alpha$, current point $x_k$
$\tilde{g}_k \leftarrow \text{stochasticGradient}(x_k)$
$x_{k+1} \leftarrow x_k - \alpha \cdot \text{sign}(\tilde{g}_k)$
```

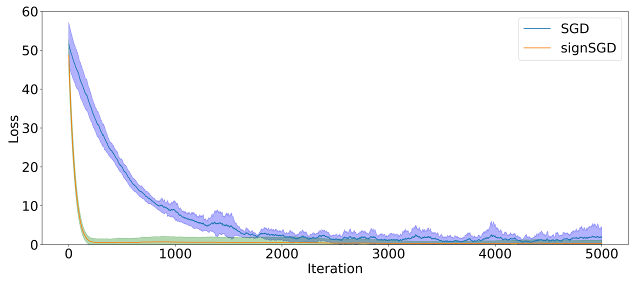

Figure 1. Comparison of SGD and SignSGD on $f(x) = \frac{1}{2}x^2$

4 Method

The main motivation of our method comes from two core ideas present in [1] which is one of the first papers in differentially private deep learning and [5] which discusses special type of compressed optimization techniques.

4.1 Bottlenecked Backpropagation

Backpropagation is de facto the standard in training deep neural networks. The basic idea in backpropagation is that we optimize the loss function using gradients and the gradients flow backward from the output layer to the input layer exactly in the reverse direction of flow of the information. Briefly, each weight in the neural networks gets updated as follows:

$$w^{t+1} = w^t - \eta \frac{\partial \mathcal{L}}{\partial w}, \tag{9}$$

where $\mathcal{L}$ is the loss function and η is the learning rate often fixed by some hyper-parameter. We can rewrite the above equation as

$$w^{t+1} = w^t - \eta\, s_w^t\, m_w^t, \tag{10}$$

where s_w^t is the sign of the gradient at iteration t for weight w, and m_w^t is the magnitude of the gradient at iteration t for weight w. For the bottlenecked backpropagation our idea is to use the following update rule:

$$w^{t+1} = w^t - \alpha^t\, s_w^t, \tag{11}$$

where α^t is not dependent on w and also can be a public information. The motivation behind the above idea is that we restrict the impact

Algorithm 2 SignAdam [38]

```
Input: learning rate $\alpha$, decay rates $\beta_1$, $\beta_2$, current point $x_k$, initial accumulators $m_0$, $v_0$
Initialize: $m_0 \leftarrow 0$, $v_0 \leftarrow 0$ $t \leftarrow 0$
$g_k \leftarrow$ stochasticGradient$(x_k)$
$m_t \leftarrow \beta_1 \cdot m_{t-1} + (1-\beta_1) \cdot g_k$
$v_t \leftarrow \beta_2 \cdot v_{t-1} + (1-\beta_2) \cdot g_k^2$
$\hat{m}_t \leftarrow m_t/(1-\beta_1^t)$
$\hat{v}_t \leftarrow v_t/(1-\beta_2^t)$
$\tilde{g}_k \leftarrow \hat{m}_t/(\sqrt{\hat{v}_t}+\epsilon)$
$x_{k+1} \leftarrow x_k - \alpha \cdot \text{sign}(\tilde{g}_k)$
```

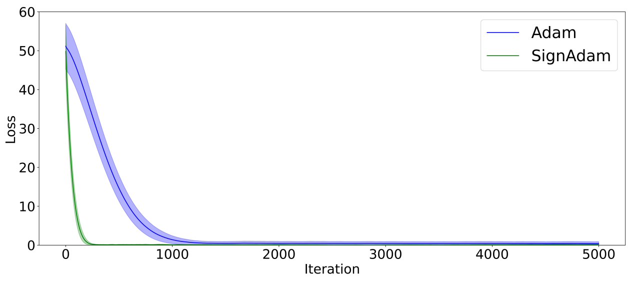

Figure 2. Comparison of Adam and SignAdam on $f(x) = \frac{1}{2}x^2$

of data on the change in parameters of the neural network to a great extent compared to the clipping trick used in DPSGD [1]. Furthermore, by restricting the impact of the data on the learning of the neural networks, we in turn reduce sensitivity of the backpropagation algorithm. Therefore we refer to this approach as bottlenecked backpropagation. We implement bottlenecked backpropagation with stochastic gradient descent (SGD) and Adam [20] to be reffered as signSGD and signAdam. We describe them below.

4.1.1 *Bottlenecked Backpropagation with SignSGD*

The SignSGD algorithm presents a subtle variation from traditional stochastic gradient descent [32, 5]. Instead of utilizing the magnitude of the gradient for descent, it solely relies on the sign of the gradient. In essence, if the gradient is positive, a value of $+1$ is utilized for the next update; conversely, if the gradient is negative, a value of -1 is utilized. Algorithm 1 provides an overview of the SignSGD algorithm. The exclusive use of signs may raise doubts regarding its convergence. To address this concern, we conducted a comparative analysis between traditional SGD and its signed counterpart on a toy function $f(x) = \frac{1}{2}x^2$ for 10 iterations. The comparison plot is illustrated in Figure 1, and for more comprehensive details on this optimizer, refer to [5].

4.1.2 *Bottlenecked Backpropagation with SignAdam*

SignAdam introduces a nuanced adaptation to the traditional Adam optimization algorithm [20]. Unlike Adam, which considers both the magnitude and direction of the gradient, SignAdam solely focuses on the sign of the gradient similar to SignSGD. Algorithm 2 outlines the procedure of SignAdam. A comparative convergence analysis between Adam and SignAdam is illustrated in Figure 2 that is done on the same toy example and averaged over ten runs. Further insights on SignAdam can be explored in [38].

Algorithm 3 Differentially Private SignSGD (DPSignSGD)

```
Input: Examples $\{x_1, \dots, x_N\}$, loss function $L(\theta) = \frac{1}{N}\sum_i L(\theta, x_i)$.
Parameters: Learning rate $\alpha$, noise scale $\sigma$, group size $S$, gradient norm bound $C$.
Initialize $\theta_0$ randomly.
for $t \in [T]$ do
    Take a random sample $S_t$ with sampling probability $\frac{S}{N}$.
    Compute gradient:
    for each $i \in S_t$ do
        Compute $g_t(x_i) \leftarrow \nabla_{\theta_t} L(\theta_t, x_i)$.
    end for
    Clip gradient:
    $\bar{g}_t(x_i) \leftarrow g_t(x_i)/\max\left(1, \frac{\|g_t(x_i)\|_2}{C}\right)$.
    Add Noise and take sign:
    $g_t^{\sim} \leftarrow \text{sign}\left(\frac{1}{S}\sum_i \bar{g}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})\right)$.
    Descent:
    $\theta_{t+1} \leftarrow \theta_t - \alpha_t \cdot g_t^{\sim}$.
end for
Output $\theta_T$ and compute the overall privacy cost $(\epsilon, \delta)$ using a privacy accounting method.
```

Table 1. Model architectures used for MNIST and FMNIST

Layer	Parameters
Convolution	16 filters of 8×8, stride 2, padding 2
Max-Pooling	32 filters of 4×4, stride 2, padding 0
Convolution	2×2, stride 1
Max-Pooling	2×2, stride 1
Fully connected	32 units
Fully connected	10 units

4.2 *Differential Privacy with Bottlenecked Backprop*

Differentially private bottlenecked backpropagation extends the framework of DPSGD [1] that adds noise to the gradients during backpropagation. We consider SignSGD and SignAdam algorithms for the exploration in this paper. For ensuring differential privacy we adopt sampled Gaussian mechanism and Reny differential privacy. Privacy analysis of the method is given in the following subsection.

Note that, the strategy to make the training differentially private, we first clip the gradients and add noise using the sampled Gaussian mechanism and Reny differential privacy, then we take the sign of the contaminated gradient to update the weights. We do not add noise only to the sign because that may induce a large amount of randomness which may jeopardize the training process.

Deferentially Private Sign based Stochastic Gradient Descent (DP-SignSGD) algorithm takes a set of examples $x_1, \dots, x_N$ and a loss function $L(\theta)$, where $L(\theta) = \frac{1}{N}\sum_i L(\theta, x_i)$. It requires parameters such as the learning rate η_t, noise scale σ, group size S, and gradient norm bound C. The model parameters θ_0 are initialized randomly. At each iteration t, a random sample S_t is drawn with a sampling probability of S/N. For each $i \in S_t$, the gradient $g_t(x_i)$ is computed as $\nabla_{\theta_t} L(\theta_t, x_i)$. The gradients are clipped to ensure that their L_2 norm does not exceed the specified bound C. The clipped gradient $\bar{g}_t(x_i)$ is calculated as $g_t(x_i)/\max(1, |g_t(x_i)|_2/C)$. Gaussian noise with zero mean and variance $\sigma^2 C^2 I$ is added to the average of clipped gradients, and then the sign function is applied to obtain the noisy gradient $g_t^{\sim}$. Here I is the identity matrix. The new gradient is calculated as sign $\left(\frac{1}{S}\sum_i \bar{g}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 I)\right)$ i.e. first adding the Gaussian noise to the gradient and the taking the sign of noise

Algorithm 4 Differentially Private SignSGD with Adaptive Moment Estimation (DPSignAdam)

```
**Input:** Examples $\{x_1, \ldots, x_N\}$, loss function $L(\theta) = \frac{1}{N}\sum_i L(\theta, x_i)$.
**Parameters:** Base learning rate $\alpha$ noise scale $\sigma$, group size $S$, gradient norm bound $C$, decay rates $\beta_1$, $\beta_2$.
Initialize $\theta_0$ randomly.
Initialize first moment vector $m_0 = 0$, second moment vector $v_0 = 0$ and $\epsilon_a = 10^{-8}$
**for** $t \in [T]$ **do**
    Take a random sample $S_t$ with sampling probability $\frac{S}{N}$.
    **Compute gradient:**
    **for** each $i \in S_t$ **do**
        Compute $g_t(x_i) \leftarrow \nabla_{\theta_t} L(\theta_t, x_i)$.
    **end for**
    **Clip gradient:**
    $\bar{g}_t(x_i) \leftarrow g_t(x_i)/\max\left(1, \frac{\|g_t(x_i)\|_2}{C}\right)$.
    **Add Noise and take sign:**
    $g_t^{\sim} \leftarrow \text{sign}\left(\frac{1}{S}\sum_i \bar{g}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})\right)$.
    **Update biased first moment estimate:**
    $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1-\beta_1) \cdot g_t^{\sim}$.
    **Update biased second raw moment estimate:**
    $v_t \leftarrow \beta_2 v_{t-1} + (1-\beta_2)(g_t^{\sim})^2$.
    **Compute bias-corrected first moment estimate:**
    $\hat{m}_t \leftarrow m_t/(1-\beta_1^t)$.
    **Compute bias-corrected second raw moment estimate:**
    $\hat{v}_t \leftarrow v_t/(1-\beta_2^t)$.
    **Descent:**
    $\theta_{t+1} \leftarrow \theta_t - \alpha_t \cdot \hat{m}_t/(\sqrt{\hat{v}_t} + \epsilon_a)$.
**end for**
Output $\theta_T$ and compute the overall privacy cost $(\epsilon, \delta)$ using a privacy accounting method.
```

Table 2. Model architecture used for IMDB

Layer	Parameters
Embedding	100 units
Fully connected	32 units
Bidirectional LSTM	32 units
Fully connected	16 units
Fully connected	2 units

added gradient. The model parameters are updated using the descent step, where θ_{t+1} is calculated as $\theta_t - \alpha_t \cdot g_t^{\sim}$. Finally, the trained parameters θ_T are outputted, and the overall privacy cost (ϵ, δ) is computed using a privacy accounting method. Algorithm 3 describes the method.

The Differentially Private SignSGD with Adaptive Moment Estimation (DPSignAdam) algorithm operates as follows: it takes a set of examples $x_1, \ldots, x_N$ and a loss function $L(\theta)$, where $L(\theta) = \frac{1}{N}\sum_i L(\theta, x_i)$. Parameters include the base learning rate η_t, noise scale σ, group size S, gradient norm bound C, and decay rates β_1 and β_2. The model parameters θ_0 are initialized randomly, along with the first moment vector m_0 and the second moment vector v_0. At each iteration, the algorithm samples a subset S_t with a probability of S/N, computes the gradients $g_t(x_i)$ for each $i \in S_t$, clips the gradients to ensure their L_2 norm does not exceed C, adds Gaussian noise to the average of clipped gradients, and takes the sign to obtain the noisy gradient $g_t^{\sim}$. It then updates the biased first moment estimate m_t and second raw moment estimate v_t, computes bias-corrected estimates $\hat{m}_t$ and $\hat{v}_t$, and updates the model parameters using the descent step. Finally, it outputs the trained parameters θ_T and computes the overall privacy cost (ϵ, δ) using a privacy accounting method. Algorithm 4 describes the method.

In both algorithms, instead of adding noise directly to each gradient computation, noise is added after aggregating the gradients from a group of samples (in DPSignSGD) or after calculating the biased first and the second moment estimates (in DPSignAdam).

4.3 Privacy Analysis

Calculation of the privacy budget is a significant task in differentially private algorithms. Our method to compute the privacy budget related to bottlenecked backprop follows sampled Gaussian mechanism (SGM) and Rényi Differential Privacy (RDP).

Definition 6. ***(Sampled Gaussian Mechanism (SGM) [25])*** *The sampled Gaussian mechanism $M_{Gq,\sigma}(S)$ for a real valued function f defined samples from a set S associated with a sampling probability mass function q is defined to be*

$$M_{Gq,\sigma}(S) \triangleq f(\{x : x \in S \text{ sampled with } q\}) + \mathcal{N}\left(0, \sigma^2 I_d\right) \quad (12)$$

Now we state a result to quantify the privacy budget of sampled Gaussian mechanism following Reny differential privacy.

Lemma 3. ***(RDP privacy budget of SGM [25])*** *If $M_{q,\sigma}$ be the Sampled Gaussian Mechanism for some function f with sensitivity 1, then $M_{q,\sigma}$ satisfies (α, τ)-RDP for*

$$\tau \leq \frac{1}{\alpha - 1} \ln\Big(\max(A_\alpha(q,\sigma), B_\alpha(q,\sigma))\Big), \quad (13)$$

where

$$\begin{cases} A_\alpha(q,\sigma) \triangleq E_{z\sim\mu_0}\left[\left(\frac{\mu(z)}{\mu_0(z)}\right)^\alpha\right] \\ B_\alpha(q,\sigma) \triangleq E_{z\sim\mu}\left[\left(\frac{\mu_0(z)}{\mu(z)}\right)^\alpha\right] \end{cases} \quad (14)$$

with $\mu_0 \triangleq N(0,\sigma^2), \mu_1 \triangleq N(1,\sigma^2), \mu \triangleq (1-q)\mu_0 + q\mu_1$.

Furthermore, it holds for $\forall(q,\sigma) \in (0,1] \times \mathbb{R}_+$, $A_\alpha(q,\sigma) \geq B_\alpha(q,\sigma)$. Thus, $M_{q,\sigma}$ satisfies $(\alpha, \frac{1}{\alpha-1}\ln(A_\alpha(q,\sigma))$-RDP.

Next we state the composition property of the Reny differential privacy which will be useful for our computation.

Lemma 4. ***(Composition of RDP [24])*** *For two randomized mechanisms f and g such that f is (α, R_1)-RDP and g is (α, R_2)-RDP, the composition of f and g, which is defined as a sequence of results (X, Y), where $X \sim f$ and $Y \sim g$, satisfies $(\alpha, R_1 + R_2)$-RDP.*

Now we state the main result of the paper to quantify privacy budget of the bottlenecked backprop.

Theorem 5. *After accepting t model updates, the bottlenecked backprop mechanism satisfies (α, τ)-RDP with τ as:*

$$\tau(\alpha) = \frac{t}{\alpha-1}\ln\left(\sum_{i=0}^{\alpha}\binom{\alpha}{i}(1-q)^{\alpha-i}q^i \exp\left(\frac{i(i-1)}{2\sigma^2}\right)\right),$$

where $q = \frac{S}{N}$, σ is the noise multiplier of the training phase, and $\alpha > 1$ is the order.

Table 3. Classification Accuracy Comparison of DPSGD and DPSIGNSGD

Dataset Name	Method	$\epsilon = 0.5$	$\epsilon = 1.0$	$\epsilon = 2.0$	Non-Private
MNIST (Image Dataset)	DPSGD [1]	93.1%	94.9%	96.1%	99.1%
	DPSIGNSGD	**94.8%**	**95.8%**	**96.7%**	
Fashion MNIST (Image Dataset)	DPSGD [1]	78.4%	80.8%	82.3%	90.9%
	DPSIGNSGD	**79.0%**	**82.1%**	**84.5%**	
IMDB (Text Dataset)	DPAdam	56.4%	60.3%	63.5%	79.5%
	DPSignAdam	**60.9%**	**65.0%**	**67.1%**	

Table 4. Classification Accuracy Comparison of DPSGD-TS and DPSIGNSGD-TS

Dataset Name	Method	$\epsilon = 0.5$	$\epsilon = 1.0$	$\epsilon = 2.0$	Non-Private
MNIST (Image Dataset)	DPSGD-TS [29]	96.0%	96.8%	97.7%	99.1%
	DPSIGNSGD-TS	**96.2%**	**97.2%**	**97.9%**	
Fashion MNIST (Image Dataset)	DPSGD-TS [29]	79.1.%	82.6%	84.5%	90.9%
	DPSIGNSGD-TS	**80.0%**	**83.2%**	**84.7%**	
IMDB (Text Dataset)	DPAdam-TS	57.5%	60.3%	62.8%	79.5%
	DPSignAdam-TS	**63.0%**	**68.1%**	**69.1%**	

Proof. To establish the theorem, we proceed through the following logical steps. First, we leverage the RDP characteristics of the sampling Gaussian mechanism to quantify the privacy impact of each accepted model update. This analysis draws upon Definitions 6 and Lemma 3. Next, we employ the concept of composition in RDP mechanisms, as outlined in Lemma 4 to assess the cumulative privacy cost incurred by multiple accepted model updates.

In the proposed approach, f denotes the computation of clipped gradients on sampled data points, expressed as $f(x_i,\ i \in B) = \sum_{i \in B} g_t(x_i)$. As g_t is derived from the clipping of gradients with a norm bound of C, the sensitivity of f is defined to be C. [26] describes a procedure to compute $A_\alpha(q, \sigma)$ depending on integer α as Eq. 15.

$$A_\alpha = \sum_{k=0}^{\alpha} \binom{\alpha}{k} (1-q)^{\alpha-k} q^k \exp\left(\frac{k(k-1)}{2\sigma^2}\right) \tag{15}$$

Lemma 4 shows the composition property of RDP mechanisms. According to Definition 6, Lemma 3 and Lemma 4 Theorem 5 is proved.

□

5 Experimental Evaluation

In this section, we perform experiments to demonstrate the performance of bottlenecked backprop to train a differentially private neural network. We have used three real datasets and two popular optimization methods. Furthermore, we conduct experiments involving two membership inference attacks. The source code to reproduce our experiments is available at https://nmrinl.github.io/dpbb.zip.

5.1 Baseline

As baseline, we have used DPSGD [1] and two contemporary variants: DPSGD with handcrafted features [36], DPSGD with tempered sigmoid activation [29] denoted as DPSGD-HF, DPSGD-TS respectively. We do not include comparisons with approaches that modify the structures of over-parameterized models [9] or semi-supervised models like PATE [28], as they fall outside the scope of our study. We will call our algorithms with *sign* keyword attached such as DPSIGNSGD, DPSIGNSGD-TS and DPSIGNSGD-HF.

5.2 Parameter Setting

In our experimental setup, we configured the privacy budget (ε) to three different values: 0.5, 1.0, and 2.0, for each dataset. We kept the value of δ fixed at 10^{-5}. For image datasets, we utilized the SIgnSGD optimizer with zero momentum (*momentum* $= 0$), while for the IMDB dataset, we employed the SignAdam optimizer with parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$ and without any weight decay.

5.3 Datasets and Results

Table 3, Table 4, and Table 5 shows the comparison study between bottlenecked privacy-aware training and standard privacy-aware training. We give the details below.

MNIST [21] dataset comprises 60,000 training samples and 10,000 testing samples, each representing handwritten digits categorized into ten classes, with approximately 7,000 grayscale images per class. Each sample consists of a 28 × 28 grayscale image paired with a label indicating its category. In the absence of privacy measures, a model trained on handcrafted features achieves an accuracy of 99.1% after 20 epochs [36]. Table 3 shows the architecture we followed for training as indicated in [18].

FMNIST [39] encompasses 60,000 training samples and 10,000 testing samples of fashion products, distributed across 10 categories. Each category consists of approximately 7,000 grayscale images sized 28 × 28. The dataset is annotated with labels indicating the category of each image. In the absence of privacy measures, a model trained on handcrafted features achieves an accuracy of 90.9% after 20 epochs [36]. Table 3 shows the architecture we followed for training as indicated in [18].

IMDB [23] comprises 50,000 movie reviews, with each review represented as a list of word indexes and labeled with a discernible bias towards either positive or negative sentiment. The dataset is partitioned into a training set consisting of 25,000 reviews and a test set containing another 25,000 reviews. In the non-private scenario, employing a cross-entropy loss function, Adam optimizer, and an expected batch size of 32, a model attains an accuracy of 79.9% after

Table 5. Classification Accuracy Comparison of DPSGD-HF and DPSIGNSGD-HF

Dataset Name	Method	$\epsilon = 0.5$	$\epsilon = 1.0$	$\epsilon = 2.0$	Non-Private
MNIST (Image Dataset)	DPSGD-HF [36]	96.1%	97.2%	98.3%	99.1%
	DPSIGNSGD-HF	**96.8%**	**97.7%**	**98.4%**	
Fashion MNIST (Image Dataset)	DPSGD-HF [36]	83.9%	86.0%	87.8%	90.9%
	DPSIGNSGD-HF	**85.2%**	**86.9%**	**88.2%**	

Table 6. Accuracy of models for membership inference attack on MNIST

Attack	Model	$\epsilon = 0.5$	$\epsilon = 1.0$	$\epsilon = 2.0$
BlackBox-Shadow	DPSGD	0.502	0.501	0.499
	DPSIGNSGD	**0.495**	**0.500**	**0.498**
	DPSGD-TS	0.500	0.502	0.506
	DPSIGNSGD-TS	**0.495**	**0.495**	**0.498**
WhiteBox-Partial	DPSGD	0.500	0.500	0.500
	DPSIGNSGD	**0.500**	**0.500**	**0.500**
	DPSGD-TS	0.500	0.500	0.500
	DPSIGNSGD-TS	**0.500**	**0.500**	**0.500**

Table 7. Accuracy of models for membership inference attack on FMNIST

Attack	Model	$\epsilon = 0.5$	$\epsilon = 1.0$	$\epsilon = 2.0$
BlackBox-Shadow	DPSGD	0.500	0.500	0.497
	DPSIGNSGD	**0.500**	**0.498**	**0.494**
	DPSGD-TS	0.502	0.502	0.499
	DPSIGNSGD-TS	**0.500**	**0.498**	**0.494**
WhiteBox-Partial	DPSGD	0.500	0.500	0.500
	DPSIGNSGD	**0.500**	**0.500**	**0.500**
	DPSGD-TS	0.500	0.500	0.500
	DPSIGNSGD-TS	**0.500**	**0.500**	**0.500**

20 epochs. Table 2 shows the architecture we followed for training as indicated in [18]. We will not show the results for DPSIGNSGD-HF for IMDB dataset because that is a text dataset and scattering of the network which is main concept in the algorithm is not possible to perform on text.

5.4 *Resilience Against Membership Inference Attacks*

Differential privacy protection is inherently meant to be resilient against membership inference attacks. The objective is to infer the membership of some sample in the training data by utilizing the trained models. In case of a classification task, if a sample provides high accuracy then it is understood that that sample belongs to the training dataset. We used two state-of-the-art frameworks for membership inference attacks: BlackBox-Shadow [33] and WhiteBox-Partial [27].

5.4.1 *Attack Details*

We randomly partition each dataset into four subsets: the target training dataset, target testing dataset, shadow training dataset, and shadow testing dataset, maintaining a 2:1:2:1 sample size ratio. The training phase involves optimizing the attack model's parameters using the Adam optimizer and a cross-entropy loss function. During testing, the trained attack model's performance is evaluated to assess its effectiveness. Additionally, both methods integrate a Random Forest classifier to process data from shadow and partial models, enhancing attack performance. Functionalities for saving trained models and managing temporary files ensure efficient resource utilization throughout the attack process.

Table 8. Accuracy of models for membership inference attack on IMDB

Attack	Model	$\epsilon = 0.5$	$\epsilon = 1.0$	$\epsilon = 2.0$
BlackBox-Shadow	DPSGD	0.499	0.499	0.500
	DPSIGNSGD	**0.500**	**0.498**	**0.494**
	DPSGD-TS	0.502	0.502	0.499
	DPSIGNSGD-TS	**0.500**	**0.498**	**0.494**
WhiteBox-Partial	DPSGD	0.500	0.500	0.500
	DPSIGNSGD	**0.500**	**0.500**	**0.500**
	DPSGD-TS	0.500	0.500	0.500
	DPSIGNSGD-TS	**0.500**	**0.500**	**0.500**

5.4.2 *Result*

Table 6, Table 7, and Table 8 demonstrate that proposed bottlenecked differentially private algorithms are equally effective as their standard differentially private algorithms. Due to the internal use of scattering nets, we opted not to conduct tests with DPSGD-HF [36]. After the attack, the accuracy of all our methods is near 0.5, which is equivalent to random guessing. This indicates their effectiveness in defending against membership inference attacks.

5.5 *Discussion*

Through two standard optimization techniques (SGD and Adam), three real-life datasets (MNIST, FMNIST, and IMDB), and two sets of evaluations (Classification and Membership attack) we demonstrate that the concept of bottlenecked backpropagation by ignoring the magnitude of the gradients can train deep neural networks which are more efficient. The results confirm our hypothesis that by ignoring magnitude we can not only avoid the noise level in update but also reduce sensitivity towards private data. Interestingly to note, the improvement over the baseline for a relatively complex dataset (IMDB) is much larger than the rest. On all datasets, it can be seen that the models are completely confused about guessing the membership as proof of resilience against membership inference attacks.

6 Conclusion

Our study introduces a bottlenecked backprop technique which uses only sign of the gradients to update the neural network parameters. This reduces the sensitivity of the training procedure. We implement the bottlenecked version on SGD and Adam optimizers two most popular optimization techniques in deep learning. We employ Rényi DP and sampled the Gaussian mechanism to ensure differential privacy of the proposed algorithms. We compare the proposed methods with the state of the art to observe that under the same privacy budget, the proposed algorithms are significantly more accurate as well as resilient to membership inference attacks. Future research avenues may explore further refinements and extensions of bottlenecked backprop for diverse neural network architectures and applications.

References

[1] 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*, page 308–318, 2016.

[2] G. Andrew, O. Thakkar, B. McMahan, and S. Ramaswamy. Differentially private learning with adaptive clipping. advances. In *Neural Information Processing Systems 34 (2021), 17455–17466*, 2021.

[3] B. Balle and Y.-X. Wang. Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. In *International Conference on Machine Learning*, pages 394–403, 2018.

[4] R. Bassily, A. Smith, and A. Thakurta. risk minimization: Efficient algorithms and tight error bounds. In *2014 IEEE 55th annual symposium on foundations of computer science*, 2014.

[5] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar. signsgd: Compressed optimisation for non-convex problems. arXiv:1802.04434v3, 2018.

[6] M. G. J. H. Borja Balle, Gilles Barthe and T. Sato. Hypothesis testing interpretations and renyi differential privacy. In *Conference on Artificial Intelligence and Statistics. PMLR*, page 2496–2506, 2020.

[7] Z. Bu, J. Dong, Q. Long, and W. J. Su. Deep learning with gaussian differential privacy. *Harvard data science review 2020, 23*, page 10–1162, 2020.

[8] M. Bun and T. Steinke. Concentrated differential privacy: Simplifications, extensions, and lower bounds, 2016. URL https://doi.org/10.1007/978-3-662-53641-4_24.

[9] S. De, L. Berrada, J. Hayes, S. L. Smith, and B. Balle. Unlocking high-accuracy differentially private image classification through scale. arXiv preprint arXiv:2204.13650 (2022), 2022.

[10] R. Du, Q. Ye, Y. Fu, H. Hu, J. Li, C. Fang, and J. Shi. Differential aggregation against general colluding attackers. In *Proceedings of the IEEE International Conference on Data Engineering*, 2023.

[11] J. Duan, Q. Ye, and H. Hu. Utility analysis and enhancement of ldp mechanisms in high-dimensional space. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, pages 407–419. IEEE, 2022.

[12] C. Dwork, A. Roth, et al. *The algorithmic foundations of differential privacy*, volume 9, 3-4, 211-407.Issues 3-4. Foundations and Trends® in Theoretical Computer Science, 2014.

[13] C. Dwork, A. Roth, et al. *The algorithmic foundations of differential privacy*, volume 9, 3-4, 211-407. Foundations and Trends® in Theoretical Computer Science, 2014.

[14] T. V. Erven and P. Harremos. Rényi divergence and kullback-leibler divergence. IEEE Transactions on Information Theory 60, 7 (2014), 3797–3820), 2014.

[15] J. Fu, Q. Ye, H. Hu, Z. Chen, L. Wang, K. Wang, and X. Ran. Dpsur: Accelerating differentially private stochastic gradient descent using selective update and release. arXiv:2311.14056, 2023.

[16] A. Golatkar, A. Achille, Y.-X. Wang, A. Roth, M. Kearns, and S. Soatto. Mixed differential privacy in computer vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8376–8386, 2022.

[17] B. J. Grosz and S. Kraus. A firm foundation for private data analysis. *Commun.ACM*, 54(1):86–95, 2011.

[18] X. X. Jianxin Wei, Ergute Bao and Y. Yang. Dpis: An enhanced mechanism for differentially private sgd with importance sampling. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 2855–2899, 2022.

[19] R. Jin, Y. Huang, X. He, H. Dai, and T. Wu. Stochastic-sign sgd for federated learning with theoretical guarantees, 2020.

[20] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[21] Y. LeCun, C. Cortes, and C. Burges. Mnist handwritten digit database. M. ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist 2 (2010), 2010.

[22] J. Lee and D. Kifer. Concentrated differentially private gradient descent with adaptive per-iteration privacy budget. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, pages 1656–1665, 2018.

[23] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, , and C. Potts. Learning word vectors for sentiment analysis. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA. The Association for Computer Linguistics,*, pages 142–150, 2011.

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

[25] I. Mironov, K. Talwar, and L. Zhang. Rényi differential privacy of the sampled gaussian mechanism. *arXiv preprint arXiv: Learning*, 2019.

[26] I. Mironov, K. Talwar, and L. Zhang. Rényi differential privacy of the sampled gaussian mechanism, 2019.

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

[28] N. Papernot, S. Song, I. Mironov, A. Raghunathan, K. Talwar, and Úlfar Erlingsson. Scalable private learning with pate. Preprint arXiv:1802.08908, 2018.

[29] N. Papernot, A. Thakurta, S. Song, S. Chien, and Úlfar Erlingsson. Tempered sigmoid activations for deep learning with differential privacy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 9312–9321, 2021.

[30] N. Phan, X. Wu, H. Hu, and D. Dou. Adaptive laplace mechanism: Differential privacy preservation in deep learning. In *2017 IEEE international conference on data mining (ICDM). IEEE*, pages 385–394, 2017.

[31] V. Pichapati, A. T. Suresh, F. X. Yu, S. J. Reddi, and S. Kumar. Adaclip: Adaptive clipping for private sgd. arXiv preprint arXiv:1908.07643 (2019), 2019.

[32] H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951.

[33] A. Salem, Y. Zhang, M. Humbert, P. B. amd Mario Fritz, and M. Backes. Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. In *Network and Distributed Systems Security (NDSS) Symposium*, 2019.

[34] K. C. Shuang Song and A. D. Sarwate. tochastic gradient descent with differentially private updates. In *2013 IEEE global conference on signal and information processing. In 2. IEEE, 245–248*, 2013.

[35] W. Z. Tianqing Zhu, Gang Li and S. Y. Philip. *Differential Privacy and Applications*, volume 69. Springer, 2017.

[36] F. Tramer and D. Boneh. Differentially private learning needs better features (or much more data). In *International Conference on Learning Representations*, 2020.

[37] K. S. V. Veen, R. Seggers, P. Bloem, and G. Patrini. Three tools for practical differential privacy, 2018.

[38] D. Wang, Y. Liu, W. Tang, F. Shang, H. Liu, Q. Sun, and L. Jiao. Signadam++: Learning confidences for deep neural networks. In *2019 International Conference on Data Mining Workshops (ICDMW)*, pages 186–195. IEEE, 2019.

[39] H. Xiao, K. Rasul, and R. Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv e-prints (2017), arXiv–1708, 2017.

[40] Z. Xu, S. Shi, A. X. Liu, J. Zhao, and L. Chen. An adaptive and fast convergent approach to differentially private deep learning. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, page 1867–1876, 2020.

[41] X. Yang, H. Zhang, W. Chen, and T.-Y. Liu. Normalized/clipped sgd with perturbation for differentially private non-convex optimization. *arXiv preprint arXiv:2206.13033*, 2022.

[42] Q. Ye, H. Hu, K. Huang, M. H. Au, and Q. Xue. Stateful switch: Optimized time series release with local differential privacy. In *IEEE INFOCOM 2023-IEEE Conference on Computer Communications*, pages 1–10. IEEE, 2023.

[43] Q. Ye, H. Hu, X. Meng, H. Zheng, K. Huang, C. Fang, and J. Shi. Privkvm*: Revisiting key-value statistics estimation with local differential privacy. *IEEE Transactions on Dependable and Secure Computing*, pages 17–35, Jan 2023. doi: 10.1109/tdsc.2021.3107512.

[44] S. W. Yingxue Zhou and A. Banerjee. Bypassing the ambient dimension: Private sgd with gradient subspace identification. In *International Conference on Learning Representations*, 2020.

[45] D. Yu, H. Zhang, W. Chen, and T.-Y. Liu. Do not let privacy overbill utility: Gradient embedding perturbation for private learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8376–8386, 2022.

[46] X. Zhang, S. Ji, and T. Wang. Differentially private releasing via deep generative model (technical report). *arXiv e-prints*, 2018.

[47] Y. Zhang, Q. Ye, R. Chen, H. Hu, and Q. Han. Trajectory data collection with local differential privacy. *Proceedings of the VLDB Endowment*, 16(10):2591–2604, 2023.