Improving Differentially-Private Deep Learning with Gradients Index Pruning

Anonymous authors
Paper under double-blind review

Abstract

Differential Privacy (DP) provides a formal privacy guarantee preventing adversaries from inferring an individual record from populations. Differentially Private Stochastic Gradient Descent (DPSGD), the widely used method to train a model satisfying DP, inserts randomized noise to the gradients in each iteration but leads to significant accuracy decline, particularly on large and deep models. Facing the curse of dimensionality in differentially-private deep learning, we propose a Gradient Index Pruning (GIP) mechanism, which prunes gradients by a novel index perturbation scheme, to preserve important components of the gradients while reducing their sizes. Our mechanism does not alter the model, but merely adds a noisy top-k pruning step before the conventional gradients noise insertion in DPSGD. It is proven that GIP satisfies DP, yet improves accuracy over DPSGD. We also present theoretical analysis to show GIP indeed introduces less perturbation to the training. Experiments on a variety of models and datasets have demonstrated that GIP exceeds the state-of-the-art differentially-private deep learning methods by around $1-2\%$ accuracy boost.

1 Introduction

Recent work has shown that trained neural networks may leak/memorize information of the training data, posing great threats to the sensitive training data. Differential Privacy (DP) serves both as a measure to quantitatively describe the upper bound of the information leak, and mechanisms to ensure any individual sample's impact on the model is negligible. Differentially-private models have shown to be defensive against multiple privacy attacks, such as membership inference attacks (Rahman et al., 2018; Sablayrolles et al., 2019; Yu et al., 2021b), gradient matching attacks (Zhu et al., 2019), input reconstruction attacks (Carlini et al., 2019), and data poisoning attacks (Ma et al., 2019), etc.

Differentially-private stochastic gradient descent (DPSGD) has become a popular framework in differentially-private deep learning. By inserting randomized noise on clipped gradients, and composing the privacy loss through iterations, DPSGD provides DP guarantees for the private training dataset on the output model. However, the conventional method suffers significant accuracy loss due to overwhelming noise perturbation, especially on deep and wide networks. For example, training CIFAR10 with DPSGD on Wide-ResNet 16-4 (2.7M parameters) merely reaches 56.8% testing accuracy at $(1, 10^{-5})$-DP compared to 94.8% without DP, by the most recent results in (De et al., 2022).

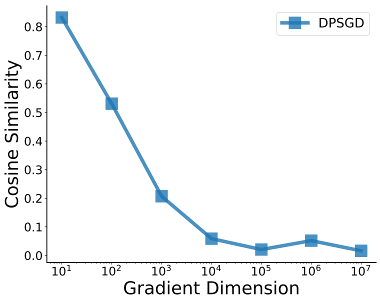

Figure 1: Cosine similarity between the differentially-private gradient vectors and their unperturbed counterparts of varied lengths. The value quickly decreases to 0 with the dimension.

Research on DPSGD has been seeking breakthroughs in improving practical utility while maintaining theoretical privacy guarantees. A series of works (Balle & Wang, 2018; Dong et al., 2021; Yang et al., 2022; Xiang et al., 2019) focus on the sufficient conditions, or the necessary and sufficient conditions for (ϵ, δ)-differential privacy, proposing stricter lower bounds for the noise variance. Another line of works investigates the accountant method (Abadi et al., 2016; Dong et al., 2021; Mironov et al., 2019), *i.e.,* composing DP over multiple iterations by tighter analyses on the higher moments of the privacy loss variable, or on the tradeoff function of type I and type II errors. Papernot et al. (2021) discovered a general family of bounded activation functions to bound the gradient sensitivity and improve the model accuracy. Recently, De et al. (2022) reveal strong evidence that hyperparameters such as batch size and learning rate are vital to DPSGD on large models. While works enhancing DPSGD from perspectives of noise bounds, accountant methods, and training hyperparameters seem to reach their limits, works of Tramèr & Boneh (2021); Yu et al. (2021b); Zhang et al. (2021) have discussed the impact of gradients sizes, which is a more practical aspect of DPSGD, leaving much room for improvement. Yu et al. (2021b; 2022); Tramèr & Boneh (2021) propose to replace the full gradients with their low-rank approximation, or to update an incremental set of weights atop a fixed pre-trained model. These methods indeed shrink the size of gradients but have to modify the original model structure at accuracy losses.

It is our key observations that 1) Lin et al. (2018) found that compression can be done on gradients with almost no impact on accuracy in SGD, 2) the magnitude of the differentially-private noise is mostly larger than that of the gradient itself, and 3) the larger the gradient size, the less likely the noisy gradients would agree with the original descent direction. To verify 2), we reproduce the experiment of training WRN16-4 on CIFAR10 in De et al. (2022), and calculate the ratio between gradient norm and noise norm in each step. The ratio remains around 0.01 during the entire training process. We verify 3) by an experiment applying $(\epsilon, \delta) = (1, 10^{-5})$-differential privacy to the gradient vectors (simulated by randomized noise sampled from $\mathcal{N}(0, 1)$) of different lengths, and show the cosine similarity between the differentially-private vectors and the unperturbed counterparts. The results in Fig. 1 show that as the gradient dimension grows, the cosine similarity value rapidly declines, suggesting the 'curse of dimensionality' in DPSGD — deep models endure much larger perturbation error at the same privacy guarantee.

Hence we are motivated to propose a new method for DPSGD with gradient pruning, requiring no model change. The intuition is to have DP mechanisms alter the gradient descent direction in each update as little as possible. We decouple the representation of the batch gradient into indices and values so that suitable mechanisms can be applied for each. The state-of-the-art Gaussian DP is used for values, while a novelly designed noisy top-k pruning method is for index perturbation. The top k elements of the gradient are selected with differential privacy. The model update is the combination result of the value distortion and index perturbation. Through theoretical analysis, we give evidence that our gradient index perturbation method introduces less noise. Intuitively, this is explained by a reduced noise dimension, and that the error brought by pruning is much smaller than by noise, since the variance of the noise is typically larger than that of the gradient.

Highlights of our contributions are as follows. *First,* we propose a new DPSGD method with gradient pruning, which effectively improves model utility while keeping the DP property. *Second,* a novel index perturbation mechanism, in combination with the value distortion, gives much smaller theoretical errors than previous works. *Finally,* experiments on a variety of models and datasets have verified that our method improves the accuracy of DPSGD by $1 - 2\%$ compared to the state-of-the-art.

2 Related Work

Differential privacy has been developed both as privacy guarantees and algorithms towards protecting individual training data records in deep learning. Particularly, differentially-private stochastic gradient descent (DPSGD) has been widely studied, and it often poses an acute problem in balancing the tradeoff between privacy and accuracy. Most works (Yu et al., 2021a;b; Tramèr & Boneh, 2021; Yu et al., 2022) point out that the high dimensions of the deep neural network is the culprit — excessive amount of noise is inserted leading to performance failure. Representative solutions typically reduce the size of the model updates (gradients) to alleviate the noise perturbation. For example, instead of applying differential privacy to full gradients, Yu et al. (2021b)propose a **low-rank approximation** to weight matrices, and perform differentially-private update in the projected subspace. Other works (Yu et al., 2022; Tramèr & Boneh, 2021) pre-train a large fraction of the model on public datasets and merely **fine-tunes an incremental set** of weights by differential privacy. Datalens (Wang et al., 2021) presents a similar idea of **gradient pruning** in training differentially-private generative models. Zhang et al. (2021) propose pruning in NLP tasks for dimensionality reduction by investigating model-specific sparsity. However, these works have their own drawbacks:

Low-rank approximation based methods rely on the public dataset or historical information to obtain the subspace for decomposition. Yu et al. (2021a) decomposes the gradients into a low-dimensional component as well as low-parity errors by the matrix projection. Yu et al. (2021b) divides the matrix into two parts by SVD decomposition, and the directional matrix is obtained from the historical gradient information. However, the approximation error of the directional matrix, whether being obtained from external data or historical information, cannot be strictly controlled. Hence the method would be invalid on standalone private datasets. Freezing the main body of a model and fine-tuning the incremental set degrades the accuracy performance compared to fine-tuning the entire model (Wang et al., 2021; De et al., 2022). Although Wang et al. (2021) prune the gradient of each sample, it does not aim to reduce the dimension of the additive noise, and thus a large amount of noise is still inserted. Zhang et al. (2021) is based on the property of gradient sparsity in NLP tasks, however, such a property may not hold on other tasks, e.g., the computer vision tasks. In contrast, our method does not rely on any auxiliary dataset, or any pre-training step to realize DPSGD. We bring down the size of the additive noise to reduce its impact.

Other lines of works on DPSGD focus on the selection of hyperparameters and training techniques (De et al., 2022) rather than the network size. We show that our method could improve the state-of-the-art further by non-trivially applying gradients pruning on top of it.

3 Preliminaries

We give a brief review of the concepts of differential privacy, DPSGD and Mallows model.

Definition 1 (**(ε, δ)-Differential Privacy** (Dwork et al., 2006)). *A randomized mechanism M satisfies (ε, δ)-differential privacy if for any neighboring datasets X and X' differing by at most one unit, and for any possible output $\mathcal{O}$,*

$$\Pr(\mathcal{M}(X) \in \mathcal{O}) \le e^{\varepsilon} \Pr(\mathcal{M}(X') \in \mathcal{O}) + \delta. \tag{1}$$

In the special case of $\delta = 0$, we call $\mathcal{M}$ ε-differentially private.

DPSGD. In a deep learning task, the sensitive training dataset $X = [x_1, x_2, \ldots, x_N]$ requires to be protected in T iterations of stochastic gradient descent. In each iteration, a batch of data B of size $|B|$ will be randomly selected to compute the gradient for weights $W : g = 1/|B| \sum_{x \in B} g(W, x)$, where $g(W, x)$ represents the gradient of the individual data x and the single g represents the average gradient of the batch. Since DP requires that the sensitivity of the outcome is bounded, conventional DPSGD conducts per-sample clipping on the gradients of each x:

$$\bar{g}(W, x) = g(W, x) / \max(1, \frac{\|g(W, x)\|_2}{C}), \tag{2}$$

to ensure that the sensitivity of the gradient is C, and thus the batch gradient becomes $\bar{g} = 1/|B| \sum_{x \in B} \bar{g}(W, x)$. DPSGD mechanism $\mathcal{M}$ inserts noise to the batch gradient: $\mathcal{M}(\bar{g}) = \bar{g} + Z$, where $Z \sim N(0, \sigma^2 C^2 E)$ has the same shape with $\bar{g}$. σ is a constant decided by the privacy budget (ϵ, δ) and C is the sensitivity of the gradient, also the clipping value.

Mallows model (Mallows, 1958) is a popular probabilistic model for permutations. The permutation of a set S is a bijection $S \mapsto S$. The mode of the distribution is given by the reference permutation I_0, and the probability of a permutation increases as it is 'closer' to I_0 as measured by rank distance metrics (*e.g.,* L_1 distance). The dispersion parameter θ controls the shape of the distribution.

Definition 2. *For dispersion parameter θ, reference permutation $I_o \in \mathrm{S}$, and rank distance measure $\boldsymbol{d} : \mathrm{S} \times \mathrm{S} \mapsto \mathbb{R}$,*

$$\mathbb{P}_{\theta, \boldsymbol{d}}(I : I_0) = \frac{1}{\psi(\theta, \boldsymbol{d})} e^{-\theta \boldsymbol{d}(I, I_0)} \tag{3}$$

is the Mallows model where $\psi(\theta, \boldsymbol{d}) = \sum_{I \in \mathrm{S}} e^{-\theta \boldsymbol{d}(I, I_0)}$ is a normalization term and $I \in \mathrm{S}$.

Without specification, we use L_1-norm distance as $\boldsymbol{d}$ throughout the paper.

4 METHODOLOGY

In this section, we analyze the impact of DP noise to SGD and present our method GIP in improving model utility for DPSGD. Considering the high-dimensional characteristics of gradients, we analyze the influence of perturbation noise to gradients from the dimension perspective. By the clipping step in DPSGD, the maximum L_2 norm of the batch gradient $\bar{g}$ is C. Meanwhile, the amount of additive noise in expectation is $\mathbb{E}_Z\|\mathcal{M}(\bar{g})-\bar{g}\|_2^2 = \sigma^2 C^2 d$ where d is the dimension of the flattened $\bar{g}$. Hence in each iteration, the perturbation error of DPSGD grows linearly with the dimension of the flattened gradient. Given the fact that most gradients can be compressed with little impact on accuracy, we are motivated to design an effective pruning step to shrink the gradient size, yet without altering the descent direction too much.

4.1 GRADIENT INDEX PRUNING

We propose a gradient pruning method based on indices selection. Conventionally, gradients can be approximated by its low-rank component, or its most prominent set of elements. By applying differentially-private update using the approximation rather than the full gradients, the expected amount of noise is reduced while the steepest descent direction of the loss is unavoidably affected. Moreover, the factorization and the prominent set selection are privacy-leaking, and thus would consume additional privacy budget.

Top-k pruning. We choose to preserve the top k ($k \in (0, 1]$) elements of the gradient under the DP constraint. It means to retain the largest k elements of the gradient arranged by the absolute values. Under the same pruning amount, the compressed gradient by top-k pruning is most likely to keep the descent direction unaltered. Unfortunately, the pruning criterion suggests the use of the private gradient data, which poses a source of privacy leakage. Hence the pruning step should be differentially-private. We propose to **decouple indices from values** in gradients so that different DP operators can be adopted for the two. For the kept values after pruning, the state-of-the-art Rényi DP (RDP) (Mironov, 2017; Mironov et al., 2019) is used whereas the indices are perturbed by our designed gradient index pruning method.

A key observation is that most of the gradients can be compressed without compromising accuracy. Hence we represent the gradients by indices $I \in \{0, 1\}^d$ and the actual values. Index 1 denotes the corresponding value is non-zero and 0 suggests otherwise. The index representation is a sequence of 1s and 0s denoting the positions of non-zero values. To compose its DP scheme, we first introduce index sensitivity:

Definition 3. *The sensitivity of the index sequence $I \in S$ is defined as*

$$s_1(I) = \sup_{d(X,X')=1} \|I(g) - I(g')\|_1,$$

where $\|\cdot\|_1$ is the L_1 norm and g, g' are the gradients caculated from neighboring input datasets X and X', respectively.

Corresponding to top-k pruning, we denote the subset of Is where k (percentage) of 1s are retained in each sequence as S_k, and the index sequence obtained from top-k method as $I_0 \in S_k$. It is a non-trivial design of S_k as **without top k, the index sensitivity could be as large as the full length of the gradient**, leading to an almost random perturbation. Within S_k, the index sensitivity is $\min\{2kd, 2d - 2kd\}$ at most. Our index perturbation mechanism is defined as:

Definition 4. *Given an index sequence $I_0 \in S_k$, a random mechansim $\mathcal{M}_p : S_k \mapsto S_k$ is defined as*

$$\mathcal{M}_p(I_0) = I, \textit{ with } \mathbb{P}_{\theta,\boldsymbol{d}}(I : I_0) = \frac{1}{\psi(\theta, \boldsymbol{d})} e^{-\theta \boldsymbol{d}(I, I_0)}. \tag{4}$$

The random variable I follows the Mallows model with parameters θ and distance metric $\boldsymbol{d}(\cdot, \cdot)$.

Given the index perturbation mechanism $\mathcal{M}_p$, we have the following theorem:

Theorem 1 (Index privacy)**.** *Given an index sequence of gradient $g : I(g) \in S_k$ pruned by top-k method and $\mathcal{M}_p(I(g)) = I$, where I follows the Mallows model $\mathbb{P}_{\theta,\boldsymbol{d}}(I : I(g))$. $\mathcal{M}_p$ is ϵ-differentially-private if and only if $\theta \leq \frac{\epsilon}{s_1(I)}$.*

The proof is provided in Appendix A.1.

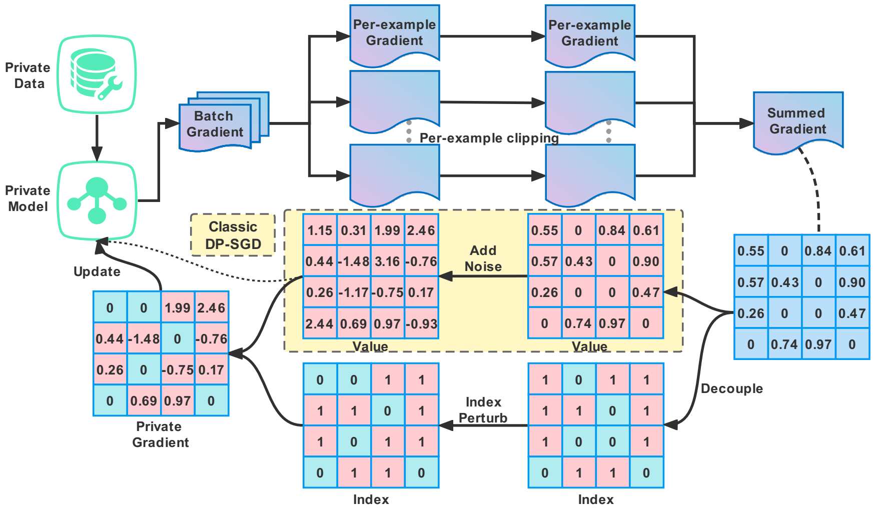

Figure 2: Overview of differentially-private SGD with gradients index pruning.

Algorithm 1 Noisy Top-k Pruning

```
Input: (a) flattened gradient vector $\mathbf{g}$ of dimension $d$, (b) pruning ratio $k$, (c) group size $\ell$.
Ensure: Perturbed index sequence $I$.
 1: Divide the gradient $\mathbf{g}$ into groups $\mathbf{g}^{(i)}$ of equal length $\ell$.
 2: for $\mathbf{g}^{(i)}$ in all groups of $\mathbf{g}$ do
 3:     Get $t_i$ as the $100k$-th percentile of the elements in $\{|\mathbf{g}_j^{(i)}|\}$
 4:     for $j \in [\ell]$ do
 5:         $I(\mathbf{g}^{(i)})_j = \begin{cases} 1, & \text{if } |\mathbf{g}_j^{(i)}| > t_i, \\ 0, & \text{if } |\mathbf{g}_j^{(i)}| \leq t_i. \end{cases}$
 6:     end for
 7:     Sample $I^{(i)}$ from $\mathbb{P}_{\theta^{(i)},\boldsymbol{d}}\left(I : I(\mathbf{g}^{(i)})\right)$.
 8:     Reinstall all $I^{(i)}$ in the original position to obtain $I$.
 9: end for
10: return $I$
```

4.2 Differentially-Private SGD with Pruning

We present differentially-private SGD with gradients index pruning in this section. The overall framework is shown in Fig. 2. Per-example gradients are computed in each iteration, and they are clipped individually before being summed up over a batch. The resulting batch gradients are decoupled into values and indices. Gaussian differential privacy mechanism is applied to the values, whereas the indices are perturbed by our noisy top-k pruning method. The private gradients are computed by multiplying the results of the two parts, and are used for model update.

The noisy top-k pruning is given in Alg. 1. We sample an index sequence following Mallows model given the original top-k gradients. However, in practice, due to the computational constraints, we cannot directly use Mallows model in the high-dimensional case. Hence we split the gradients into groups $\{\mathbf{g}^{(i)}\}$ of smaller scales by their original order in the gradients, and select the top k elements in each $\mathbf{g}^{(i)}$ by the absolute values. For example, $|\mathbf{g}_j^{(i)}|$ denotes the absolute value of the j-th element of gradient $\mathbf{g}^{(i)}$. In step 7 $I^{(i)}$ will be sampled from $\mathbb{P}_{\theta^{(i)},\boldsymbol{d}}\left(I : I(\mathbf{g}^{(i)})\right)$ and we present the sampling algorithm in AppendixA.2. The perturbation results $I^{(i)}$ are put together to obtain the index sequence I. Accordingly, the differential privacy budget is split up for each group. The parameter $\theta^{(i)}$ of Mallows model of each group should satisfy $\theta^{(i)} \leq \epsilon_2^{(i)}/s_1(I)$ for the resulting I to be $\epsilon_2 = \sum_i \epsilon_2^{(i)}$-differentially-private.

Now we illustrate the overall DPSGD algorithm in Alg. 2 which follows the general framework in Abadi et al. (2016). The gradients of each example are clipped and summed up in line 8. Perturbed index I is obtained by calling Alg. 1. We follow the privacy accountant method in De et al. (2022) to compose DP over iterations by RDP and convert it to (ϵ, δ)-DP by Theorem 21 in Balle et al. (2020).

Lemma 1 (Privacy accountant)**.** *Given training iterations T, batch sampling ratio q and standard deviation σ for Gaussian noise, DPSGD satisfies $(\epsilon_1(T, q, \sigma), \delta(T, q, \sigma))$-differential privacy.*

With the lemma, we can prove the following theorem.

Theorem 2. *Given the clipping value C, the index sensitivity $s_1(I)$, and T, q, σ defined in Lemma 1, if $\theta^{(i)} \leq \epsilon_2^{(i)}/s_1(I)$, $\epsilon = \epsilon_1(T, q, \sigma) + T\sum_i \epsilon_2^{(i)} > 0$, and $\delta = \delta(T, q, \sigma) \in (0, 1)$, Alg. 2 is (ϵ, δ)-differentially private.*

Proof. The Gaussian mechanism applied to values satisfies $(\epsilon_1(T, q, \sigma), \delta(T, q, \sigma))$-differential privacy by Lemma 1. The noisy top-k pruning meets ϵ_2-differential privacy according to Thm. 1, where we compose privacy budget for all $I^{(i)} \in I$ over T iterations as $T\sum_i \epsilon_2^{(i)}$. Thm. 2 is straightforward by taking basic composition from Dwork et al. (2014) of the two mechanisms. □

It is worth noting that line 9 of Alg. 2 can be replaced by any index pruning methods. To show the power of our design, we give a naive random-k pruning mechanism which does not consume any privacy budget. In **random-k pruning**, k of the gradient elements are randomly selected as the index sequence I. Since it does not rely on any private knowledge, this step is privacy-free. However, the gradient descent direction is altered by random sampling. In latter sections, we will compare our top-k pruning method against baselines including this naive one. It is obvious to have the following privacy guarantee held:

Proposition 1 (Differentially-private random-k pruning)**.** *By replacing line 9 of Alg. 2 with random-k pruning, Alg. 2 satisfies $(\epsilon_1(T, q, \sigma), \delta(T, q, \sigma))$-differential privacy.*

The proof is straightforward and thus is omitted.

Algorithm 2 Differentially-Private SGD with Pruning

```
Input: (a) privacy parameters $\epsilon, \delta$, (b) training samples $\{x_1, x_2, \ldots, x_N\}$, (c) model weights $W$,
    loss function $\mathcal{L}(W) = 1/N \sum_i \mathcal{L}(W, x_i)$, (d) hyperparameters: learning rate $\eta_t$, noise scale $\sigma$,
    batch size $|B|$, clipping value $C$, (e) pruning ratio $k$ and the pruning method $\mathcal{M}_p(\cdot, k)$ with ratio
    $k$.
 1: Initialize $W_0$ randomly
 2: for $t \in [T]$ do
 3:     Take a random sample $L_t$ with sampling probability $|B|/N$
 4:     for $i \in [L_t]$ do
 5:         Compute $\mathbf{g}_t(x_i) \leftarrow \nabla_{W_t} \mathcal{L}(W_t, x_i)$
 6:         $\overline{\mathbf{g}}_t(x_i) \leftarrow \mathbf{g}_t(x_i) / \max\left(1, \frac{\|\mathbf{g}_t(x_i)\|_2}{C}\right)$
 7:     end for
 8:     Accumulate the clipped gradients over a batch $\mathbf{g}_t = \sum_i^{|B|} \overline{\mathbf{g}}_t(x_i)$
 9:     Sample index sequence $I$ from $\mathcal{M}_p(\mathbf{g}_t, k)$ by Alg. 1
10:     Add noise $\tilde{\mathbf{g}}_t = \frac{1}{|B|}[\mathbf{g}_t + Z] \odot I$, where $Z \sim \mathcal{N}(0, \sigma^2 C^2)$ and $\odot$ is the Hadamard product.
11: end for
12: $W_{t+1} \leftarrow W_t - \eta_t \tilde{\mathbf{g}}_t$
13: return $W_T$
```

5 Analysis and Comparison

This section presents the analysis of the perturbation errors in our method and makes comparison with other works. We analyze the Mean Square Error(MSE) introduced in a single iteration by DPSGD with pruning. As gradient descent is taken in each iteration, we consider the less error included, the less the update deviates from the original descent direction, which leads to a smaller accumulated error in the end. As the mechanism design relies on the gradient distribution (imagine how would random-k and top-k perform at all 1s gradient vector), the perturbation analysis has to take into account the gradient distribution. We further observed from Chen et al. (2020) that, the gradient distribution gradually becomes more symmetric throughout the training process of MNIST and CIFAR10, and its center is at 0 in both. Hence, we made the following assumption to facilitate the analysis:

Assumption 1. *The batch gradient* $\mathbf{g} = \sum_{x_i \in B} \mathbf{g}(W, x_i) \in \mathbb{R}^d$ *in DPSGD follows* $\mathcal{N}(0, \sigma_g^2)$.

Under Assumption 1, we have $\mathbb{E}_{\mathbf{g}} \|\mathbf{g}\|_2 = d\sigma_g^2$, meaning that the expected L_2 norm of the gradient is bounded. We define MSE as:

$$MSE = \mathbb{E}\, \|[\mathbf{g} + Z] \odot I - \mathbf{g}\|_2^2 = \underbrace{\mathbb{E}\|\mathbf{g} \odot I - \mathbf{g}\|_2^2}_{\text{Pruning MSE}} + \underbrace{\mathbb{E}\|Z \odot I\|_2^2}_{\text{Noise MSE}}, \tag{5}$$

given the additive noise $Z \sim \mathcal{N}(0, \sigma^2 C^2)$, and the perturbed index sequence I. The error can be divided into the pruning error and the noise error, and one can easily see if the same pruning amount is applied, the noise error is the same:

$$\text{Noise MSE} = \mathbb{E}\|Z \odot I\|_2^2 = \sigma^2 C^2 kd \tag{6}$$

where $k \in (0, 1]$ is the pruning ratio. Therefore, we could compare the pruning error for different pruning methods.

Proposition 2. *Let* $\mathbf{g} \in \mathbb{R}^d$ *denote the batch gradient,* I_r *and* I_t *be the resulting index sequence of random-*k *and top-*k *pruning. Under Assumption 1, the pruning MSEs are:*

$$\begin{aligned} MSE_r &= \mathbb{E}_{\mathbf{g}, I_r} \|\mathbf{g} - \mathbf{g} \odot I_r\|_2^2 = (d - k \cdot d)\sigma_{\mathbf{g}}^2; \\ MSE_t &= \underbrace{(d - k \cdot d)\sigma_{\mathbf{g}}^2 \left[1 - k - \frac{\sqrt{2}}{\sqrt{\pi}} a \exp\left(-\frac{a^2}{2}\right)\right]}_{MSE_{t0}} + \underbrace{\sigma_{\mathbf{g}}^2 \left[1 + \frac{2a}{k\sqrt{2\pi}} \exp\left(-\frac{a^2}{2}\right)\right] F(\theta)}_{\textit{index perturbation error}}, \end{aligned} \tag{7}$$

where $a = \Phi^{-1}(1 - \frac{k}{2})$, $F(\theta) = \frac{1}{\psi(\theta, \mathbf{d})} \sum_{i=0}^{kd} \binom{kd}{i} \binom{(1-k)d}{i} i e^{-\theta 2i}$ *and* Φ *denotes the standard normal CDF. Further we have* $MSE_t \leq MSE_r$.

We reuse the denotations in Def. 2 above. Please see Appendix A.3 for the proof. In fact, MSE_t contains two parts: the error introduced by top-k selection MSE_{t0} and the index perturbation error. The former could be regarded as a lower bound for MSE_t. Obviously, we have $MSE_t > MSE_{t0}$ and the index perturbation error is the sacrifice for the differential privacy guarantee of indices. To show our advantage over DPSGD, we also list the MSE of DPSGD, which contains Noise MSE only: $MSE_d = \sigma^2 C^2 d$.

Corollary 1. *Under Assumption 1, if* $\sigma_{\mathbf{g}}^2 \leq \beta(C, k, d, \theta)\sigma^2$, *where* $\beta(C, k, d, \theta) = \frac{C^2(1-k)d}{(1-k)d\left[1-k-\frac{\sqrt{2}}{\sqrt{\pi}} a \exp\left(-\frac{a^2}{2}\right)\right] + \left[1 + \frac{2a}{k\sqrt{2\pi}} \exp\left(-\frac{a^2}{2}\right)\right] F(\theta)}$, *we have* $MSE_d > MSE_t$.

We will give a more specific example. Given the parameter setting of training WRN-16-4 on CIFAR10 with $(1, 10^{-5})$-DP in the following section, we get $\beta(C, k, d, \theta) = 0.9913$, suggesting the condition of Corollary 1 is easy to meet since σ is usually much larger than $\sigma_{\mathbf{g}}$. And we present how the value of $\beta(C, k, d, \theta)$ changes with k and d in Appendix B. Hence in most cases, we have $MSE_d > MSE_t$. We also compare GIP with other index perturbation methods, *e.g.,* PrivKV (Ye et al., 2019) and FedSel (Liu et al., 2020), in Appendix A.4.

6 EVALUATIONS

We conduct experiments in a variety of settings to demonstrate the performance of our method and baselines. Experiments are run on NVIDIA RTX3090 GPUs, and results are reported by averaging over five runs.

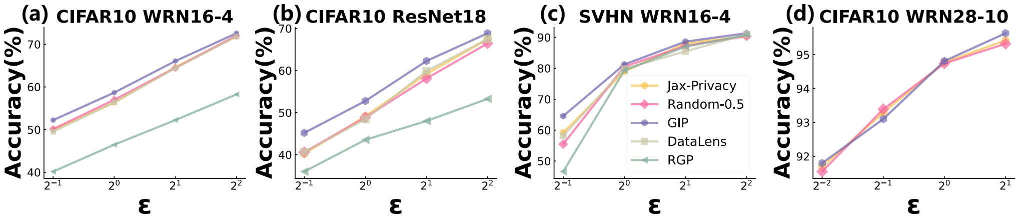

Figure 3: Comparison of GIP and other baselines over different datasets and models. Legends are shared.

6.1 Setup

Datasets and models. We choose the common image classification tasks on CIFAR-10 (Krizhevsky et al., 2009), SVHN (Netzer et al., 2011) and 1k-class ImageNet (Deng et al., 2009). The datasets of CIFAR-10, SVHN, ImageNet are divided into (45K, 5K), (68K, 5K), (1.27M, 10K) for (train set, test set), respectively. All training data is considered private. On CIFAR-10, we train Wide-ResNet (WRN)-16-4 and ResNet18 from scratch, and fine-tune the entire WRN-28-10 which has been pre-trained on the public ImageNet (De et al., 2022). On SVHN, we train WRN-16-4 from scratch. On ImageNet, a Normalizer-Free ResNet-50 (NF-ResNet-50) (Brock et al., 2021) is trained from scratch.

Baselines and metrics. We select the state-of-the-art DPSGD baselines including **RGP** (Yu et al., 2021b), **DataLens** (Wang et al., 2021) and **Jax-privacy** (De et al., 2022). RGP applies the low-rank approximation to the gradient in each iteration of DPSGD. DataLens prunes the gradients to reduce the clipping value, which also decreases the variance of the additive noise. JAX-privacy is the most recent work suggesting suitable DPSGD hyperparameters for deep models. We reproduce each baseline in the same setting as with their original paper which we believe is the optimal for their methods, and compare with GIP under the same privacy budget (ϵ, δ).

Hyperparameters. Our implementation is built on `Jax` and we adopt the training hyperparameters as that of Jax-Privacy (De et al., 2022). Clipping value is set to $C = 1$ by default. We fix the privacy parameter $\delta = 10^{-5}$ on CIFAR-10 and SVHN, $\delta = 8 \times 10^{-7}$ on ImageNet. We allocate the privacy budget for index perturbation in proportion to ϵ: $\epsilon_2 = 0.01\epsilon$ in all experiments. And we set the batch size, learning rate, augmentation multiplicity and training steps the same for Jax-Privacy, Random-k, GIP and DataLens following De et al. (2022) and the specific values are listed in Tab. 4,3 in Appendix B. Note that since the privacy budget used by GIP for Gaussian Mechanism is $\epsilon_1 = 0.99\epsilon$, which is smaller than ϵ used by Jax-Privacy and Random-k, and hence our GIP injects a larger amount of noise for the same number of update steps. For RGP, we follow the settings in Yu et al. (2021b) which are listed in Appendix B. In our experiments, if we set the pruning ratio to a fixed value, the performance is suboptimal since the gradients vary greatly, particularly at the start of the training, according to Sec. 5 from Chen et al. (2020). Therefore, it is not reasonable to get rid of most gradients at the beginning of the training. Hence we apply an exponential decay as well as a linear decay schedule for the pruning ratio k. For the group size ℓ in Alg. 1, we set $\ell = 256$, and the privacy budget is combined over groups.

6.2 Results

Comparison with baselines. We depict the accuracies of each model trained over different privacy budgets in Fig. 3. For GIP, we set the pruning ratio schedule as a linear decay from $k = 1.0$ to 0.1. Since the same pruning ratio leads to inferior results on Random-k, we select an optimal schedule particularly for the method: an exponential decay from $k = 1.0$ to 0.5. As RGP requires to re-define the convolutional layers, we adopt its original adaptation to WRNs in experiments.

Among all baselines, Jax-Privacy has the best performance as it is trained with carefully-tuned hyperparameters. Nevertheless, GIP improves Jax-Privacy by $2-3\%$ for models trained from scratch, and has a comparable performance in the pre-training case. Most notably, at $(0.5, 10^{-5})$-DP, GIP enhances accuracy by 5% (from 59.20% to 64.55%) from Jax-Privacy on WRN-16-4, SVHN. At low privacy budgets, *i.e.,* smaller ϵs, the improvement is more significant, mostly because a large σ is used in DPSGD at a small ϵ, which leads to a bigger gap between MSE_t and MSE_d by Corollary 1. DataLens and Random-k showed close performance to Jax-Privacy, deviating by at most 1%.

RGP performs poorly on CIFAR-10, WRN-16-4 and ResNet18, but has a performance close to Jax-Privacy on WRN-16-4, SVHN when $\epsilon \geq 1$. We consider RGP significantly reduces model size by the low-rank method (*e.g.,* from 2.732M to 0.032M on WRN-16-4), but unfortunately fails to handle deep models on relatively complex datasets, or in high privacy regime. The improvement of GIP on pre-trained models are limited as shown in Fig. 3(d), since all methods including Jax-privacy and random-k have performance close to the non-private version, leaving little room for improvement. More numerical results are presented in Appendix B.

Tab. 1 records the testing accuracy of training NF-ResNet-50 from scratch on ImageNet at different privacy levels. The optimal pruning schedule is selected for each method. We choose a linear decay and an exponential decay from $k = 1.0$ to 0.5 for GIP and random-k, respectively. On complicated and large dataset as ImageNet-1k, we observe a mild $0.1 - 0.4\%$ improvement of GIP over Jax-Privacy. Random-k is not consistently better than Jax-Privacy, indicating that random pruning may be harmful to the accuracy.

Running time. To find out the computational overhead of GIP, we compare its training time with baselines in Tab. 2. GIP incurs mild additional overhead compared to Jax-Privacy and Random-k, mainly due to its shuffling mechanism. DataLens is much slower while RGP is the fastest, since it is the only method actually changing the model size.

Table 1: Top-1 and top-5 accuracies (%) of ImageNet trained from scratch.

	Jax-Privacy		Random-k		GIP	
ϵ	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
0.25	0.99	3.84	1.01	3.59	**1.13**	**4.02**
0.5	2.39	7.97	**2.52**	8.20	2.47	**8.34**
0.75	4.20	12.75	3.98	12.52	**4.44**	**13.22**
0.8	4.76	**13.95**	4.54	13.45	**4.96**	13.88

Table 2: The training time on WRN-16-4, SVHN over 875 steps. RDP over 100 epochs.

Method	Total(h)	Batch(s)
Jax-Privacy	6.68	12.6
Random-0.5	6.6	12.4
GIP	7.08	13.33
DataLens	14.18	26.34
RGP	0.92	-

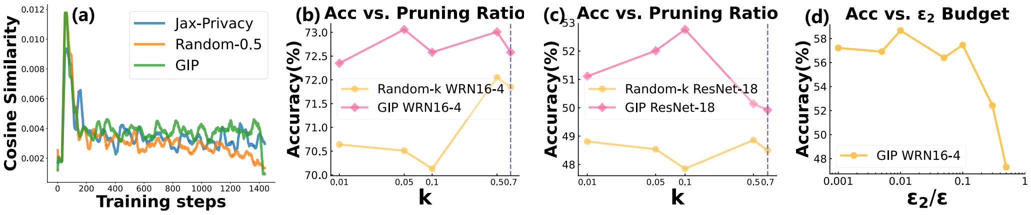

Figure 4: (a) Cosine similarity over training on CIFAR10, ResNet18. (b)(c) Ablation studies of CIFAR-10, WRN-16-4 and ResNet18 under different ks. (d) Ablation study of CIFAR-10 on WRN-16-4 under different budget allocations. $\epsilon_2 = T \sum_i \epsilon_2^{(i)}$ is for indices and the rest for values.

Case studies. To verify the reason why GIP is superior in accuracy, we record the cosine similarity between the DP gradient vectors and their unperturbed counterparts of different methods on CIFAR10, ResNet18 with $(1, 10^{-5})$-DP. From Fig. 4(a), we can tell the cosine similarities follow GIP $>$ Jax-Privacy $>$ Random-0.5 overall, which verifies that GIP effectively mitigates the variation in gradient descent direction. The overall decreasing trend of cosine similarity is due to that the gradient norm of most examples decay over training, while the noise norm does not not change over the course. Hence the direction of gradient does not agree any more in the latter phase of training. And the results agree with Fig. 3(b) where higher cosine similarity values correspond to higher accuracies.

We study how the varying hyperparameters would affect GIP's performance. First, we select different pruning ratio ks under $(4, 10^{-5})$-DP. The testing accuracies under different ks are shown in Fig. 4(b)(c). Here we selected $k \in \{0.01, 0.05, 0.1, 0.5, 0.7\}$. On WRN-16-4, the results of GIP mostly vary within 0.5% and the highest is at $k = 0.05$. The accuracy of random-k fluctuates greatly across different ratios with the highest occurring at $k = 0.5$. On ResNet18, GIP's accuracy peaks at $k = 0.1$ while random-k has the worst performance at that ratio. Overall, random-k performs poorly at low ratios (< 0.1), indicating that pruning alone would degrade accuracy at such ratios.

We also display how accuracies vary across different proportions of privacy budgets assigned to index perturbation in Fig. 4(d) under $(1, 10^{-5})$-DP. The trend is that, as that proportion of privacy budget grows, the accuracy first increases and then decreases, peaking at 1%. With the proportion going over 10%, the accuracy quickly decays, as the noise, rather than the indices selection, plays a more important role to performance.

7 Conclusion

We propose a new method GIP to improve the accuracy performance of DPSGD in deep models. The key is to prune the gradients to reduce the amount of additive noise, yet without altering the gradient descent direction too much. By decomposing the gradients representation into indices and values, GIP applies different DP mechanisms to the two components, and achieves an overall (ϵ, δ)-DP. We not only theoretically prove but also experimentally verify that GIP improves the DPSGD accuracy over the state-of-the-art.

References

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

Borja Balle and Yu-Xiang Wang. Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018.

Borja Balle, Gilles Barthe, Marco Gaboardi, Justin Hsu, and Tetsuya Sato. Hypothesis testing interpretations and renyi differential privacy. In *International Conference on Artificial Intelligence and Statistics*, pp. 2496–2506. PMLR, 2020.

Andrew Brock, Soham De, and Samuel L. Smith. Characterizing signal propagation to close the performance gap in unnormalized resnets. 2021.

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In Nadia Heninger and Patrick Traynor (eds.), *28th USENIX Security Symposium, USENIX Security 2019, Santa Clara, CA, USA, August 14-16, 2019*, pp. 267–284. USENIX Association, 2019.

Xiangyi Chen, Steven Z. Wu, and Mingyi Hong. Understanding gradient clipping in private sgd: A geometric perspective. *Advances in Neural Information Processing Systems*, pp. 13773–13782, 2020.

Soham De, Leonard Berrada, Jamie Hayes, Samuel L Smith, and Borja Balle. Unlocking high-accuracy differentially private image classification through scale. *arXiv preprint arXiv:2204.13650*, 2022.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

Jinshuo Dong, Aaron Roth, and Weijie Su. Gaussian differential privacy. *Journal of the Royal Statistical Society*, 2021.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pp. 265–284. Springer, 2006.

Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Yujun Lin, Song Han, Huizi Mao, Yu Wang, and William J Dally. Deep Gradient Compression: Reducing the communication bandwidth for distributed training. In *The International Conference on Learning Representations*, 2018.

Ruixuan Liu, Yang Cao, Masatoshi Yoshikawa, and Hong Chen. Fedsel: Federated sgd under local differential privacy with top-k dimension selection. In *International Conference on Database Systems for Advanced Applications*, pp. 485–501. Springer, 2020.

Yuzhe Ma, Xiaojin Zhu, and Justin Hsu. Data poisoning against differentially-private learners: Attacks and defenses. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, IJCAI'19, pp. 4732–4738, 2019. ISBN 9780999241141.

C. L. Mallows. Non-null ranking models. *Biometrika*, 44(1-2):114–130, 1958.

Ilya Mironov. Rényi differential privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pp. 263–275. IEEE, 2017.

Ilya Mironov, Kunal Talwar, and Li Zhang. Rényi differential privacy of the sampled gaussian mechanism. *CoRR*, abs/1908.10530, 2019.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

Nicolas Papernot, Abhradeep Thakurta, Shuang Song, Steve Chien, and Úlfar Erlingsson. Tempered sigmoid activations for deep learning with differential privacy. In *AAAI*, 2021.

Md. Atiqur Rahman, Tanzila Rahman, Robert Laganière, and Noman Mohammed. Membership inference attack against differentially private deep learning model. *Transactions on Data Privacy*, 11(1):61–79, 2018. URL http://www.tdp.cat/issues16/tdp.a289a17.pdf.

Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, Yann Ollivier, and Hervé Jégou. White-box vs black-box: Bayes optimal strategies for membership inference. In *International Conference on Machine Learning*, 2019.

Florian Tramèr and Dan Boneh. Differentially private learning needs better features (or much more data). *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YTWGvpFOQD-.

Boxin Wang, Fan Wu, Yunhui Long, Luka Rimanic, Ce Zhang, and Bo Li. Datalens: Scalable privacy preserving training via gradient compression and aggregation. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pp. 2146–2168, 2021.

Liyao Xiang, Jingbo Yang, and Baochun Li. Differentially-private deep learning from an optimization perspective. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pp. 559–567. IEEE, 2019.

Jungang Yang, Liyao Xiang, Ruidong Chen, Weiting Li, and Baochun Li. Differential privacy for tensor-valued queries. *IEEE Transactions on Information Forensics and Security*, 17:152–164, 2022. doi: 10.1109/TIFS.2021.3089884.

Qingqing Ye, Haibo Hu, Xiaofeng Meng, and Huadi Zheng. Privkv: Key-value data collection with local differential privacy. In *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 317–331. IEEE, 2019.

Da Yu, Huishuai Zhang, Wei Chen, and Tie-Yan Liu. Do not let privacy overbill utility: Gradient embedding perturbation for private learning. In *International Conference on Learning Representations*, 2021a.

Da Yu, Huishuai Zhang, Wei Chen, Jian Yin, and Tie-Yan Liu. Large scale private learning via low-rank reparametrization. In *International Conference on Machine Learning*, pp. 12208–12218. PMLR, 2021b.

Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, et al. Differentially private fine-tuning of language models. In *International Conference on Learning Representations*, 2022.

Huanyu Zhang, Ilya Mironov, and Meisam Hejazinia. Wide network learning with differential privacy. *CoRR*, abs/2103.01294, 2021. URL https://arxiv.org/abs/2103.01294.

Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. In *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 2019.

A THE PROOFS PART

A.1 PROOF OF THEOREM 1

Proof. Here we present the proof of the differential privacy for index perturbation mechanism. In order to achieve equation 1, the privacy loss and privacy budget ϵ should satisfy

$$\log \frac{\Pr(M_I(I(g)) \in \mathcal{O})}{\Pr(M_I(I(g')) \in \mathcal{O})} \leq \epsilon. \tag{8}$$

Substituting the probability density function expression into it, we can get

$$\log \frac{e^{-\theta \boldsymbol{d}(I, I(g))}}{e^{-\theta \boldsymbol{d}(I, I(g'))}} \leq \epsilon \Leftrightarrow \ \theta \left[\boldsymbol{d}(I, I(g')) - \boldsymbol{d}(I, I(g))\right] \leq \epsilon. \tag{9}$$

By the triangular inequality of distance, we can get

$$\theta \left[\boldsymbol{d}(I, I(g')) - \boldsymbol{d}(I, I(g))\right] \leq \theta \boldsymbol{d}(I(g'), I(g)) \leq \theta C_1(I) \tag{10}$$

Therefore, if the mechanism M_I satisfies ϵ-differential privacy, then θ must satisfies $\theta \leq \frac{\epsilon}{C_1(I)}$.

□

A.2 SAMPLING ALGORITHM OF MALLOWS MODEL

Sampling Algorithm of Mallows Model is demonstrated in Alg. 1. Although the Probability Density Function(PDF) of Mallows model has been shown in Definition 2, it is not easy to generate an index sample I that follows the distribution in the definition. Instead, we sample from a subspace of all d-dimensional permutations. We first sample a distance variable $\boldsymbol{d}(I, I_t)$, of which PDF is shown in equation 11. Here we set $S_k(I_t, i) = \{I | \boldsymbol{d}(I, I_t) = 2i, I \in S_k\}$.

$$\begin{aligned} &\sum_{I \in S_k} \frac{1}{\psi(\theta, \boldsymbol{d})} e^{-\theta \boldsymbol{d}(I, I_t)} = \frac{1}{\psi(\theta, \boldsymbol{d})} \sum_{i=0}^{kd} \sum_{I \in S_k(I_t, i)} e^{-\theta \boldsymbol{d}(I, I_t)} \\ =&\sum_{i=0}^{kd} \frac{1}{\psi(\theta, \boldsymbol{d})} \binom{kd}{i} \binom{(1-k)d}{i} e^{-\theta 2i} = \sum_{i=0}^{kd} \mathbb{P}\left[\boldsymbol{d}(I, I_t) = 2i\right] \end{aligned} \tag{11}$$

Let $\boldsymbol{d}(I, I_t)$ in the first step be i. In the second step, index sequence I_t is perturbed by randomly flipping i 1s to 0s and i 0s to 1s. The resulting sequence is the sampled I.

A.3 PROOF OF PROPOSITION 2

Proof. First, we calculate the MSE_r of random-k pruning method:

$$MSE_r = \mathbb{E}_{\mathbf{g}, I_r} \|\mathbf{g} - \mathbf{g} \odot I_r\|_2^2 = \mathbb{E}_{\mathbf{g}} \sum_{I_r^{(i)} = 0} \mathbf{g}_i^2 \tag{12}$$

Since the random-k method does not have any effect on the gradient $\mathbf{g}$, each element $\mathbf{g}_i$ still follows the Gaussian distribution $\mathcal{N}(0, \sigma_{\mathbf{g}}^2)$. Therefore, the MSE is the sum of all the expectation of $\mathbf{g}_i^2$:

$$MSE_r = (d - k \cdot d)\sigma_{\mathbf{g}}^2. \tag{13}$$

Second, for top-k pruning method, the pruning index I_t is calculated by $\mathbf{g}$ without perturbation. As we adopt gradient pruning, I_t should be recomputed. However, direct calculation is problematic due to the sorting problem involved in gradients. We thus use an alternative approach, where we treat each dimension of the gradient as the same random variable being sampled d times, and split these samples into two parts by the k quantiles of the absolute value. Therefore, in the distribution, one part has a weight of k, and the other part is $1 - k$. From this, we can get the k quantiles a as

$$\int_{-a}^{a} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) dx = 1 - k \tag{14}$$

As $\mathcal{N}(0,\sigma_{\mathbf{g}}^2)=\sigma_{\mathbf{g}}\mathcal{N}(0,1)$, we unify the quantiles for gradients on the standard normal distribution and compare $|\mathbf{g}_i|$ and $a\sigma_{\mathbf{g}}$ in the algorithms. We can then calculate a as

$$\int_{-a}^{a}\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{x^2}{2}\right)dx=2\Phi(a)-1\Longleftrightarrow a=\Phi^{-1}(1-\frac{k}{2}) \tag{15}$$

Next, we calculate that the value of MSE of $d-k\cdot d$ gradients $\mathbf{g}_i/\sigma_{\mathbf{g}}\in(-a,a)$ as

$$\begin{aligned}\mathbb{E}_{\mathbf{g}}\|\mathbf{g}-\mathbf{g}\odot I_t\|_2^2&=(d-k\cdot d)\sigma_{\mathbf{g}}^2\int_{-a}^{a}\frac{x^2}{\sqrt{2\pi}}\exp\left(-\frac{x^2}{2}\right)dx\\&=\frac{d\sigma_{\mathbf{g}}^2(1-k)}{\sqrt{2\pi}}\left[-x\exp\left(-\frac{x^2}{2}\right)\Big|_{-a}^{a}+\int_{-a}^{a}\exp\left(-\frac{x^2}{2}\right)dx\right]\\&=d\sigma_{\mathbf{g}}^2(1-k)\left[-\frac{2a}{\sqrt{2\pi}}\exp\left(-\frac{a^2}{2}\right)+1-k\right],\end{aligned} \tag{16}$$

By definition, the MSE in the algorithm after perturbation by Mallows Model is

$$MSE_t=\sum_{I\in S_k}\|\mathbf{g}-\mathbf{g}\odot I\|_2^2\mathbb{P}_{\theta,\boldsymbol{d}}(I,I_t) \tag{17}$$

We can divide all Is into different sets according to the distance between I and I_t. For I in the same set, we find that their probabilities are all equal. We split the index into two sets, $C_0(I)$ and $C_1(I)$, which represent the set with index 0 and the set with index 1 respectively. Then we have

$$\begin{aligned}&C_0(I)=\{j|I_j=0\},\ C_1(I)=\{j|I_j=1\}\\&|C_0(I)|=|C_0(I_t)|=k\cdot d,\ \ |C_1(I)|=|C_1(I_t)|=(1-k)\cdot d\end{aligned} \tag{18}$$

where $|\cdot|$ denotes the number of elements in the set. Therefore, the number of index differences between $C_1(I)$ and $C_1(I_t)$ is equal to the number of index differences in $C_0(I)$ and $C_0(I_t)$.

$$(1-k)\cdot d-|C_0(I)\cap C_0(I_t)|=k\cdot d-|C_1(I)\cap C_1(I_t)| \tag{19}$$

We then set the number of index differences in $C_0(I)$ and $C_0(I_t)$ as i and its range of variation is $0\le i\le k\cdot d$. Therefore, we can denote the distance $\boldsymbol{d}(I,I_t)$ and MSE by i:

$$\begin{aligned}&\boldsymbol{d}(I,I_t)=d-|C_0(I)\cap C_0(I_t)|-|C_1(I)\cap C_1(I_t)|=2i\\&MSE_t=\sum_{i=0}^{k\cdot d}\|\mathbf{g}-\mathbf{g}\odot I\|_2^2\frac{1}{\psi(\theta,\boldsymbol{d})}e^{-\theta 2i}\cdot|\{I|\boldsymbol{d}(I,I_t)=2i\}|\end{aligned} \tag{20}$$

The difference between I and I_t can be regarded as randomly selecting i elements from C_0 to be 1 and i elements from C_1 to be 0. Hence the number of $|S_k(I_t,i)=\{I|\boldsymbol{d}(I,I_t)=2i,I\in S_k\}|$ is

$$|S_k(I_t,i)|=\binom{kd}{i}\binom{(1-k)d}{i} \tag{21}$$

For $I\in S_k(I_t,i)$,

$$\mathbb{E}_{\mathbf{g}}\|\mathbf{g}-\mathbf{g}\odot I\|_2^2=\sum_{j\in C_0(I)}g_j^2=\sum_{j\in C_0(I)\cap C_0(I_t)}g_j^2+\sum_{j\in C_0(I)\cap C_1(I_t)}g_j^2 \tag{22}$$

Since $I\in S_k(I_t,i)$ has equal probability, the gradient index of the random sample in $C_0(I)\cap C_0(I_t)$ and $C_0(I)\cap C_1(I_t)$ follow the uniform distribution.

$$\begin{aligned}&\mathbb{E}_{\mathbf{g}}\sum_{j\in C_0(I)\cap C_0(I_t)}g_j^2+\sum_{j\in C_0(I)\cap C_1(I_t)}g_j^2\\=&MSE_{I_t}\cdot\frac{(1-k)d-i}{(1-k)d}+(d\sigma_g^2-MSE_{I_t})\cdot\frac{i}{kd}\\=&\sigma_g^2[(1-k)d-i]\,[1-\zeta]+\sigma_g^2(k+\zeta-k\zeta)\frac{i}{k}\\=&\sigma_g^2\left[(1-k)(1-\zeta)d+\frac{\zeta i}{k}\right]=MSE_{I_t}+\sigma_g^2\frac{\zeta i}{k},\end{aligned} \tag{23}$$

where we set $MSE_{I_t} = \mathbb{E}_{\mathbf{g}}\|\mathbf{g} - \mathbf{g} \odot I_t\|_2^2$, and $\zeta = \frac{2a}{\sqrt{2\pi}} \exp\left(-\frac{a^2}{2}\right) + k$ for short. Finally,

$$MSE_t = \sum_{i=0}^{k\cdot d}\left[MSE_{I_t} + \sigma_g^2 \frac{\zeta i}{k}\right]\frac{1}{\psi(\theta, \boldsymbol{d})}e^{-\theta 2i}|S_k(I_t, i)| = MSE_{I_t} + \sum_{i=0}^{k\cdot d}\sigma_g^2\frac{\zeta i}{k}\frac{1}{\psi(\theta, \boldsymbol{d})}e^{-\theta 2i}|S_k(I_t, i)|$$
$$= MSE_{I_t} + \sigma_g^2\left[1 + \frac{2a}{k\sqrt{2\pi}}\exp\left(-\frac{a^2}{2}\right)\right]\frac{1}{\psi(\theta, \boldsymbol{d})}\sum_{i=0}^{kd}\binom{kd}{i}\binom{(1-k)d}{i} ie^{-\theta 2i}, \tag{24}$$

where we substitue the sum in the last equation with equation 21. □

A.4 Comparison with other works

There are also some related works on privacy perturbation on gradient index, e.g. PrivKV(Ye et al., 2019), FedSel(Liu et al., 2020) and DataLens(Wang et al., 2021). Therefore, here we mainly compare the difference between the random response mechanism and the Mallows Model we employ.

The core idea of PrivKV and FedSel is to perturb the index by a random response mechanism, and to make the mechanism satisfy differential privacy by controlling the probability of index flipping. But both of these work in a distributed context with privacy guarantees using LDP. We found in our study that if the random response mechanism needs to be made to satisfy differential privacy, then poorer results will occur in the high-dimensional case.

Proposition 3. *Let $I_0 \in R^d$ is the index of gradient. The random response mechanism defined as*

$$I_{(j)} = \begin{cases} I_{0(j)} & w.p.\ p \\ 1 - I_{0(j)} & w.p.\ 1-p \end{cases} \tag{25}$$

will satisfy ϵ_2-DP if

$$p \leq \frac{e^{\frac{\epsilon_2}{C_1(I)}}}{1 + e^{\frac{\epsilon_2}{C_1(I)}}} \tag{26}$$

where $C_1(I)$ is the index sensitivty from Def. 3.

By this property, we can find that the random answering mechanism can also satisfy the requirement of differential privacy. However, the index sensitivity tends to be very large in DP-SGD, causing the random response mechanism does not work well. As an example, in Wide ResNet 16-4, d is approximately 10^6. To limit the size of index sensitivity, we performed per example pruning, keeping 1% of index for each gradient, in this case, we calculated that $C_1(I) = 2 \times 10^4$, and bringing this result into property 3, it is easy to conclude that $e^{\frac{\epsilon_2}{C_1(I)}}$ is close to 1 and the result is close to $p \leq \frac{1}{2}$. With such a flipping probability, the output of index is basically a random flipping with half being 0 and half being 1, which loses the meaning of pruning.

B Experiments and Results

B.1 The Discussion on Corollary 1

In Fig. 5, we illustrate how β varies with dimensionality d and pruning ratio k by Corollary 1. We choose $d \in [10, 200]$ as the max group size is selected to be 256 and $k \in [0.1, 0.9]$. We can see that the value of β is dominantly determined by k. A larger k and a larger d most likely lead to a greater β. We presented the value of $\log(\beta)$ in the figure, and the smallest β is 0.78 at $(k, d) = (0.1, 10)$. In the experiment in CIFAR-10, Wide ResNet 16-4, the value of σ_g^2/σ^2 is basically around 0.0001, which is much smaller than beta. Therefore, Corollary 1 holds in the majority of cases.

B.2 Hyperparameters

Here we present the detail information of hyperparameters in Tab. 3, 4. In implementing DataLens, we select the ‘Top-k-Portion parameter’ defined in Wang et al. (2021) as 0.8 for all the DataLens experiments. In implementing RGP, we select the low-rank approximation parameter to be $rank =$ 16. The RGP experiments were done with reference to Yu et al. (2021b) without further tricks. We set the learning rate as $\{2, 4, 2, 4\}$ in Tab. 3 which correspond to the varied $\sigma = \{0.5, 1, 2, 4\}$.

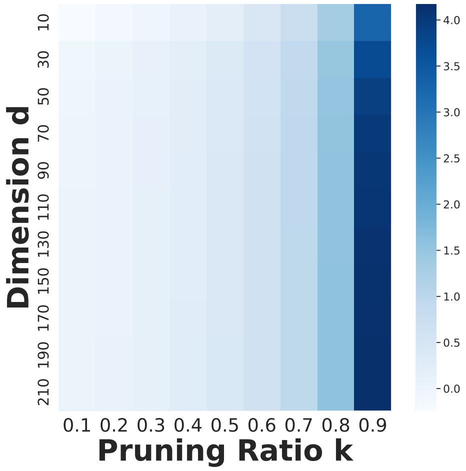

Figure 5: The value of $\beta(C, k, d, \theta)$ varies with k and d.

Table 3: Training hyperparameters.

	Learning rate	Batch size	Augmentation multiplicity
WRN-16-4	2	4096	16
ResNet18	$\{2, 4, 2, 4\}$	4096	16
WRN-28-10	1	16384	16
ImageNet	4	16384	4
RGP	1	1000	-

B.3 ACCURACIES

In Tab. 5, we display the numerical results of accuracy in each experiment for a clearer comparison. We highlight the data points in bold with the highest accuracy for each ϵ. We also provide comparison with Papernot et al. (2021); Tramèr & Boneh (2021) in Tab. 6 on Wide ResNet 16-4, CIFAR10.

Table 4: Privacy budget settings.

WRN-16-4	ϵ	0.5	1.0	2.0	4.0
ResNet 18	σ	13.0	10.0	6.0	4.0
WRN-28-10	ϵ	0.25	0.5	1.0	2.0
	σ	30.0	22.0	21.1	15.8
ImageNet	ϵ	0.2	0.4	0.6	0.8
	σ	2.5	2.5	2.5	2.5
RGP	ϵ	0.5	1.0	2.0	4.0
	epoch	30	100	150	200

Table 5: The detailed results in Fig. 3 of different methods and ϵs.

Testing accuracy(%)					
	ϵ	0.5	1	2	4
WRN-16-4 CIFAR-10	Jax-Privacy	49.97	56.3	64.64	71.81
	Random-0.5	50.12	57	64.45	72.05
	GIP	**52.22**	**58.67**	**66.07**	**72.58**
	DataLens	49.51	56.41	64.4	71.82
	RGP	36.25	46.46	52.23	59.74
	ϵ	0.5	1	2	4
ResNet-18 CIFAR-10	Jax-Privacy	40.23	49.1	59.18	67.43
	Random-0.5	40.6	48.86	58.1	66.44
	GIP	**45.19**	**52.76**	**62.26**	**68.85**
	DataLens	40.62	48.44	59.85	67.51
	RGP	35.99	43.54	48.01	53.26
	ϵ	0.5	1	2	4
WRN-16-4 SVHN	Jax-Privacy	59.2	78.95	88.02	90.38
	Random-0.5	55.46	80.53	87.22	90.36
	GIP	**64.55**	**81.23**	**88.61**	**91.32**
	DataLens	58.16	79.86	85.46	90.95
	RGP	46.59	79.78	87.02	90.72
	ϵ	0.25	0.5	1	2
WRN-28-10 CIFAR-10	Jax-Privacy	93.29	94.77	95.42	91.69
	Random-0.5	**93.39**	94.74	95.32	91.55
	GIP($k = 0.5$)	93.1	**94.81**	**95.63**	**91.8**

Table 6: Comparison with baselines under the same ϵs on Wide ResNet 16-4, CIFAR10.

Method	ϵ	Accuracy
Tramèr & Boneh (2021)	1	60.00%
	2	**66.84%**
	3	**69.30%**
Papernot et al. (2021)	7.53	66.20%
GIP	1	59.01%
	2	66.68%
	3	**70.87%**
	7.53	**80.39%**