DataLens: Scalable Privacy Preserving Training via Gradient Compression and Aggregation

Boxin Wang*
boxinw2@illinois.edu
University of Illinois at Urbana-Champaign
Illinois, USA

Fan Wu*
fanw6@illinois.edu
University of Illinois at Urbana-Champaign
Illinois, USA

Yunhui Long*
ylong4@illinois.edu
University of Illinois at Urbana-Champaign
Illinois, USA

Luka Rimanic
luka.rimanic@inf.ethz.ch
ETH Zürich
Zürich, Switzerland

Ce Zhang
ce.zhang@inf.ethz.ch
ETH Zürich
Zürich, Switzerland

Bo Li
lbo@illinois.edu
University of Illinois at Urbana-Champaign
Illinois, USA

ABSTRACT

Recent success of deep neural networks (DNNs) hinges on the availability of large-scale dataset; however, training on such dataset often poses privacy risks for sensitive training information. In this paper, we aim to explore the power of generative models and gradient sparsity, and propose a scalable privacy-preserving generative model DataLens, which is able to generate synthetic data in a differentially private (DP) way given sensitive input data. Thus, it is possible to train models for different down-stream tasks with the generated data while protecting the private information. In particular, we leverage the generative adversarial networks (GAN) and PATE framework to train multiple discriminators as "teacher" models, allowing them to vote with their gradient vectors to guarantee privacy.

Comparing with the standard PATE privacy preserving framework which allows teachers to vote on *one-dimensional* predictions, voting on the *high dimensional gradient vectors* is challenging in terms of privacy preservation. As dimension reduction techniques are required, we need to navigate a delicate tradeoff space between (1) the improvement of privacy preservation and (2) the slowdown of SGD convergence. To tackle this, we propose a novel dimension compression and aggregation approach TopAgg, which combines top-k dimension compression with a corresponding noise injection mechanism. We theoretically prove that the DataLens framework guarantees differential privacy for its generated data, and provide a novel analysis on its convergence to illustrate such a tradeoff on privacy and convergence rate, which requires non-trivial analysis as it requires a joint analysis on gradient compression, coordinate-wise gradient clipping, and DP mechanism. To demonstrate the practical usage of DataLens, we conduct extensive experiments on diverse datasets including MNIST, Fashion-MNIST, and high dimensional CelebA and Place365 datasets. We show that DataLens significantly outperforms other baseline differentially private data generative models. Our code is publicly available at https://github.com/AI-secure/DataLens.

CCS CONCEPTS

• **Security and privacy** → *Software security engineering*; • **Computing methodologies** → **Neural networks**.

KEYWORDS

Differential Privacy, Generative Models, Gradient Compression

ACM Reference Format:
Boxin Wang, Fan Wu, Yunhui Long, Luka Rimanic, Ce Zhang, and Bo Li. 2021. DataLens: Scalable Privacy Preserving Training via Gradient Compression and Aggregation. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS '21), November 15–19, 2021, Virtual Event, Republic of Korea.* ACM, New York, NY, USA, 23 pages. https://doi.org/10.1145/3460120.3484579

1 INTRODUCTION

Advanced machine learning methods, especially deep neural networks (DNNs), have achieved great success in a wide array of applications [22, 23, 60], mainly due to the fast development of hardware, their expressive representation power, and the availability of large-scale training datasets. However, one major concern that has risen in machine learning is that the training data usually contain a large amount of privacy sensitive information (e.g., human faces and medical records), which could be leaked via the trained machine learning models [51, 64]. *How to protect such private information while allowing high learning utility for the dataset* has attracted a lot of attention. Differentially private (DP) deep learning [2] proposes adding Gaussian noise to the clipped gradient during training, thus ensuring that the learned results are differentially private regarding the training data. However, its learning utility largely decreases with strong privacy requirements. A semi-supervised learning framework PATE [43, 44] is later proposed to improve the learning effectiveness at the presence of privacy noise, by leveraging the aggregation of noisy teacher models trained on

*Authors contributed equally to this research.

CCS '21, November 15–19, 2021, Virtual Event, Republic of Korea

ACM ISBN 978-1-4503-8454-4/21/11…$15.00
https://doi.org/10.1145/3460120.3484579

private datasets. It is shown that the PATE framework is able to improve the learning utility significantly while protecting data privacy. However, applying such privacy preserving training framework from the discriminative model to the generative model to guarantee that the generated data is differentially private is non-trivial given the potential high-dimensional gradient aggregation.

To further improve the flexibility of differentially private machine learning process, in this paper we aim to design a *privacy-preserving data generative model* which ensures that the data generator and the generated data, instead of only the predictions, are differentially private. This way, the generated data can then be used to train arbitrary models for different down-stream tasks with high flexibility. Having in mind that the generative adversarial networks (GAN) [19] achieved great success in terms of generating high quality data, it is natural to ask: *Is it possible to leverage the power of GAN in a way to generate data in a differentially private manner?* Some recent works have shown promising results on differentially private data generative models [37, 62]. However, most of them can only generate low dimensional data such as tabular data with weak privacy guarantees (i.e., $(\epsilon, \delta) - DP$ with small ϵ). The problem of *generating differentially private high dimensional data* (e.g., image) with strong privacy guarantees is still open, due to the fact that, in order to achieve strong privacy guarantees, the limited privacy budget is not enough to train a generative model to approximate any high dimensional perturbation.

In the meantime, an independent line of research concerning gradient compression in distributed training for communication efficiency [5, 7, 35, 57] shows that some noisy compression schemes such as only keeping the top-K elements of the gradient would achieve statistically similar convergence rate with vanilla training. This observation could potentially be a remedy for the above problem of high dimensionality — Intuitively, the noises introduced by these noisy compression schemes could also help protect privacy and combining them with traditional DP noise mechanism may allow us to add fewer amount of noise to achieve the same level of DP protection. This intuition inspired our work, which, to our best knowledge, is the *first* to marry these two lines of research on privacy and communication-efficient distributed learning to achieve both differential privacy guarantees and high model utility on high-dimensional data. As we will see, though intuitively feasible, taking advantage of this intuition is far from trivial.

Specifically, we propose a differentially private data generative model DataLens based on the PATE framework, which trains multiple discriminators as different *teacher* models to provide the back-propagation information in a differentially private way to the *student* generator. In addition, to tackle the high-dimensional data problem we mentioned above, we propose an effective noisy gradient compression and aggregation strategy TopAgg to allow each discriminator to vote for the top several dimensions in their gradients and then aggregate their noisy gradient sign to perform back-propagation. We prove the differential privacy guarantees for both the data generator and generated data for DataLens. Furthermore, to ensure the performance of the trained DP generative model, we provide a theoretical *convergence* analysis for the proposed gradient compression and aggregation strategy. In particular, to our best knowledge, this is the first convergence analysis considering the *coordinate-wise* gradient clipping together with gradient compression and DP noise mechanism.

Finally, we conduct extensive empirical evaluation on the utility of the generated based on DataLens comparing with several other baselines on image datasets such as MNIST, Fashion-MNIST, CelebA, and Place365, which is of much higher dimension than the tabular data used by existing DP generative models. We show that the generated data of DataLens can achieve the state-of-the-art utility on all datasets compared with baseline approaches. We also conduct a series ablation studies to analyze the visualization quality of the generated data, the data-dependent and data-independent privacy bounds, the impact of different components and hyper-parameters in DataLens, as well as different gradient compression methods.

In addition, to further evaluate the proposed compression and aggregation strategy TopAgg, which is the key building block in DataLens, we also discuss and evaluate TopAgg for the standard DP SGD training. We show that on both MNIST and CIFAR-10 datasets, TopAgg can achieve similar or even better model utility than the state of the art baseline approaches, which leads to an interesting future direction.

Technical Contributions. In this paper, we propose a general and effective differentially private data generative model for high-dimensional image data. We make contributions on both theoretical and empirical front.

- We propose an effective differentially private data generative model DataLens, which can be applied for generating high-dimensional image data with limited privacy budgets.
- We prove the privacy guarantees for DataLens, and conduct thorough theoretical analysis for the convergence of DataLens. We show that DataLens is able to make a good tradeoff between the privacy protection by adding DP noise and the slowdown of SGD convergence due to the added DP noise.
- We propose a novel noisy gradient compression and aggregation algorithm TopAgg by combining the top-k dimension compression and a specific DP noise injection mechanism. We also discuss the potential of adapting TopAgg to standard DP SGD training with evaluations.
- To illustrate tradeoff between differential privacy and convergence given gradient compression, we provide a novel theoretical analysis jointly considering gradient compression, coordinate-wise gradient clipping, and DP mechanism.
- We conduct extensive empirical evaluation on DataLens with four image datasets, including MNIST, Fasion-MNIST, CelebA, and Place365 datasets. We show that in term of the utility of generated data, DataLens significantly outperforms the state-of-the-art DP generative models.

2 PRELIMINARIES

Here we will first provide some background knowledge on differential privacy and data generative models. We then draw connections between the definitions we introduced here and our analysis on DataLens later.

2.1 Differential Privacy

(ε, δ)-differential privacy ((ε, δ)-DP) is currently an industry standard of privacy notion proposed by Dwork [16] It bounds the

change in output distribution caused by a small input difference for a randomized algorithm. The following definition formally describes this privacy guarantee.

Definition 1 ((ε, δ)-Differential Privacy [16])**.** A randomized algorithm $\mathcal{M}$ with domain $\mathbb{N}^{|\mathcal{X}|}$ is (ε, δ)-differentially private if for all $\mathcal{S} \subseteq \text{Range}(\mathcal{M})$ and for any neighboring datasets D and D':

$$\Pr[\mathcal{M}(D) \in \mathcal{S}] \leq \exp(\varepsilon) \Pr[\mathcal{M}(D') \in \mathcal{S}] + \delta.$$

Differential privacy is immune to post-processing. Formally, the composition of a data-independent mapping g with an (ε, δ)-DP mechanism $\mathcal{M}$ is also (ε, δ)-DP [17].

PATE Framework. Private Aggregation of Teacher Ensembles (PATE) is one of the DP mechanisms [43, 44] that provide the differential privacy guarantees for trained machine learning models. The PATE framework achieves DP by aggregating the prediction votes from several *teacher models*, which are trained on private data, as the input with DP noise for a *student model*, which serves as the final released prediction model with privacy protection. The privacy analysis [43] of PATE is derived using Laplacian mechanism and moments accountant technique based on Abadi et al. [2], which yields a tight privacy bound when the outputs of teacher models have high consensus over the topmost votes.

2.2 Data Generative Models

Data generative models aim to approximate the distribution of large datasets and thus generate diverse datasets following the similar data distribution., which can be used for data augmentation and further analysis. Recently, Generative Adversarial Network (GAN) [19] has been proposed as a deep learning architecture for training generative models. In particular, GAN consists of a generator Ψ that learns to generate synthetic records, and a discriminator Γ that is trained to tell real records apart from the fake ones. Given an input datase x and a sampled noise z, we train the discriminator Γ to maximize the likelihood of classifying the synthetic example from Ψ as drawing from the real distribution with the loss function $\mathcal{L}_\Gamma$ defined as: $\mathcal{L}_\Gamma = -\log \Gamma(x) - \log(1 - \Gamma(\Psi(z))$. The generator Ψ seeks to minimize the probability of the generated data being predicted as fake ones by the discriminator Γ with the loss function $\mathcal{L}_\Psi$ defined as: $\mathcal{L}_\Psi = -\log \Gamma(\Psi(z))$. Though GANs are able to generate high-quality data records given large training datasets, such generative models are prone to leak the information of training data [14]. This presents us the challenge on *how to prevent the training information leakage for generated data*, especially when the training data contains a large amount of privacy-sensitive information. In this paper, we aim to train differentially private generative models so that we can enjoy the benefits of generative models to generate unlimited amount of high-utility data for arbitrary downstream tasks, while protecting sensitive training information.

2.3 Gradient Compression

Gradient compression techniques, such as quantization, low-rank approximation, and sparsification, have been studied in the last decade [8, 12, 27, 34, 52, 55]. One surprising result is that stochastic gradient descent are often robust to these operations — one can often compress the data by orders of magnitude without significantly slow down the convergence. Most of existing efforts focus on saving the communication overheads in distributed training.

This paper is inspired by these previous research, however, focuses on a different problem — *can the saving of communication overheads provide benefits to differential privacy?* As we will see, by compressing the gradient in certain way, we are able to decrease the dimension of the gradient without significantly slow down the convergence. This can translate into fewer amount of noises that one needs to add to ensure DP. This intuition, however, requires careful design of the underlying algorithm and imposes novel challenges in theoretical understandings, which is the focus of this work.

3 THREAT MODEL & METHOD OVERVIEW

In this section, we will first introduce the threat model that we consider in this paper, then provide an overview of the proposed DataLens framework as a differentially private data generative model. We also provide an overview of the proposed noisy gradient compression and aggregation method TopAgg, which serves as one of the key building blocks in DataLens.

3.1 Threat Model and Goal

In practice, the machine learning models are usually trained by data containing a large amount of privacy sensitive information. Thus, given a trained model, an attacker is able to train some shadow models with partial data or leverage other strategies to infer the "membership" of a training instance [51], which leads to the leakage of sensitive information. For instance, if a person is known to have participated in a heart disease test, her privacy of having heart disease would be revealed. An attacker is also able to recover the training information via data recovery attacks [10, 11].

Differential privacy (DP) can protect against *membership inference attacks* and *training-data memorization* [11, 61]. Intuitively, differential privacy guarantees that when the input dataset differs by one record, the output distribution of a differentially private algorithm does not change by much. This definition reduces the risk of membership inference attacks and data recovery attacks given that it prevents the algorithm from memorizing individual record in the input training dataset.

In this paper, our **goal** is to ensure the differential privacy guarantees for training machine learning models, and therefore protect the privacy of training data. There has been a line of research focusing on providing differential privacy guarantees for the trained machine learning models by adding DP noise during training [2]. Here we mainly consider a more flexible case, where we will design a differentially private data generative model, which ensures that the *generated data* instead of the model's parameters are differentially private as proved in Theorem 3. Thus, as long as the data are generated, they can be used for training arbitrary down-stream learning tasks with differential privacy guarantees.

Note that besides privacy-preserving, it is also critical to make sure that the generated data is of high **utility**, and therefore we evaluate the *prediction accuracy* of models trained on the DP generated data and test their accuracy on real testset. Different with existing data generative models, "visual" quality of the generated

DP data is not the main goal of this paper, and we will provide evaluation on the visual quality of the generated data for understanding purpose in Section 5.2 and Table 3. We believe it is interesting future research to integrate other losses to further improve the visual quality of the generated data if it is part of the goal.

3.2 Method Overview

Here we briefly illustrate the proposed DataLens framework, as well as the novel noisy gradient compression and aggregation approach TopAgg which serves as a key building block in DataLens. The **goal** of DataLens is to generate high-dimensional data which will not leak private information in the training data. In terms of privacy preserving ML training, PATE [44] so far has achieved the state of the art performance, which motivates our privacy analysis. Figure 1 presents an overview for the structure of DataLens. This framework combines the algorithm TopAgg for high dimensional differentially private (DP) gradient compression and aggregation with GAN and the PATE framework. DataLens consists of an ensemble of teacher discriminators and a student generator. The teacher discriminators have access to randomly partitioned non-overlapping sensitive training data. In each *training iteration*, each teacher model produces a gradient vector to guide the student generator in updating its synthetic records. These gradient vectors from different teachers are compressed and aggregated using the proposed DP gradient aggregation algorithm TopAgg before they are sent to the student generator.

The DP gradient compression and aggregation step is crucial for the privacy protection and utility of the generator. Yet, it is challenging for the algorithm to both preserve high data utility and achieve a strong privacy guarantee. To achieve high data utility, the algorithm needs to preserve the correct gradient directions of the teacher models. As for the privacy guarantee, privacy composition over a high dimensional gradient vector often consumes high privacy budget, resulting in a weaker privacy guarantee.

To address this problem, prior work uses random projection to project the gradient vector onto lower dimensions [37]. However, this approach introduces excessive noise to the gradient directions and greatly undermines the utility of the model, making it hard to analyze the convergence.

In DataLens, we propose a novel algorithm TopAgg for high dimensional DP gradient compression and aggregation. Our main insight hinges on gradient sparsification as indicated in recent work on communication-efficient distributed learning [4, 5]: we can apply aggressive lossy compression on the gradient vectors without slowing down SGD convergence. In this paper, we identify a specific lossy compression scheme under which we can leverage more efficient DP mechanism, thus increasing the utility significantly.

In particular, the proposed gradient compression and aggregation algorithm TopAgg takes the top-k entries in a gradient vector and compresses them via stochastic sign gradient quantization [24]. This step significantly reduces the dimensionality of a gradient vector while preserving the most valuable gradient direction information. After the compression, we perform DP gradient aggregation over the sign gradient vectors with a corresponding noise injection mechanism. Since the gradient vectors have been compressed, the aggregation algorithm has a much *lower sensitivity*, which leads to a tighter privacy bound. We have also provide a theoretical analysis for the convergence of TopAgg in Section 4.3, which to our best knowledge is the first convergence analysis considering the *coordinate-wise* gradient clipping together with gradient compression and DP noise mechanism.

4 DATALENS: SCALABLE PRIVACY PRESERVING GENERATIVE MODEL

We first present our privacy preserving data generative model DataLens, then perform a rigorous analysis on its privacy guarantee and convergence, and demonstrate the privacy-utility trade-off controlled by the proposed gradient compression method. We also briefly discussion how to adapt the proposed noisy gradient compression and aggregation algorithm TopAgg from DataLens to standard SGD training.

4.1 DataLens Training

We now present the main algorithms used in DataLens. It consists of three parts: an ensemble of teacher discriminators, a student generator, and a DP gradient aggregator. First, we introduce the algorithm for training the student generator and teacher discriminators. Then, we introduce the novel high-dimensional DP gradient compression and aggregation algorithm TopAgg (Algorithm 3). This algorithm consists of two parts: a top-k gradient compression algorithm (`TopkStoSignGrad`, Algorithm 2) that compresses the gradient vectors while preserving the important gradient directions; and a DP gradient aggregation algorithm that aggregates teacher gradient vectors with differential privacy guarantees.

Training DP Generator via Teacher Discriminator Aggregation. On the high level, as shown in Figure 1 the teacher discriminators are trained on non-overlapping sensitive data partitions to distinguish between real and synthetic data. The student generator produces synthetic records, sends them to the teachers for label querying, and uses the aggregated gradient from the teacher discriminators to improve its generated synthetic records. The DP gradient aggregator ensembles the teachers' gradient vectors and adds DP noise for privacy guarantees. The detailed algorithm for this process is included in the Algorithm 1.

To begin with, we randomly partition the sensitive training dataset into non-overlapping subsets of the same size. Each partition is associated with one teacher discriminator. Then, we iteratively update the student generator and the teacher discriminators. Each iteration consists of the following four steps:

Step 1: Training teacher discriminators. The student generator Ψ produces a batch of synthetic records. Each teacher discriminator Γ_i updates the weights based on standard discriminator loss $\mathcal{L}_{\Gamma_i}$ to reduce its loss on distinguishing the synthetic records from real records in its training data partition.

Step 2: Generating and compressing teacher gradient vectors. Each teacher discriminator Γ_i computes a gradient vector $g^{(i)}$ of the discriminator loss $\mathcal{L}_{\Gamma_i}$ with regard to the synthetic records. Such gradient vector contains the information that could guide the student generator to improve its synthetic records aiming to increase the generated data utility (*i.e.*, classification accuracy of trained models).

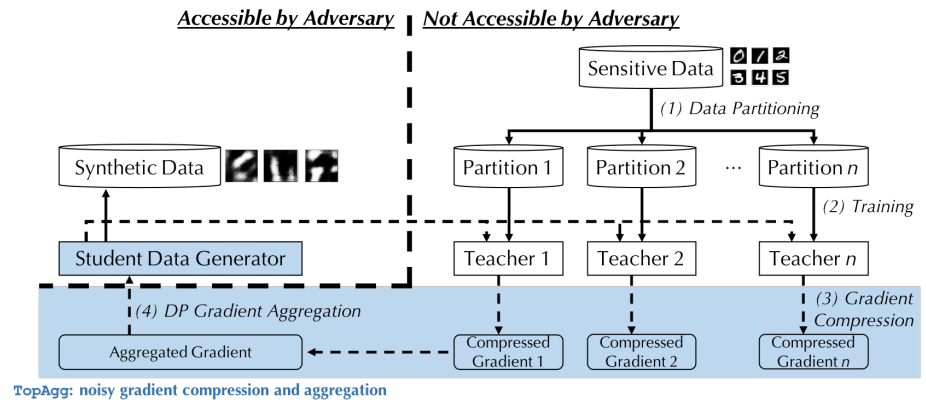


Figure 1: Overview of DataLens. DataLens consists of an ensemble of teacher discriminators and a student generator. DataLens provides a novel algorithm TopAgg for *high dimensional DP gradient compression and aggregation.* TopAgg consists of two parts: (1) top-k and sign gradient compression that selects the top k gradient dimensions, and (2) DP gradient aggregation for high-dimensional sparse gradients. The solid arrows denote the data flow, while the dash arrows denote the gradient flow.

Algorithm 1 - Training the Student Generator.

Input: batch size m, number of teacher models N, number of training iterations T, gradient clipping constant c, top-k, noise parameters σ, voting threshold β, disjoint subsets of private sensitive data $d_1, d_2, \ldots, d_N$, learning rate γ
for number of training iterations $\in [T]$ **do**
  ▷ *Phase I: Pre-Processing*
  Sample m noise samples $\mathbf{z} = (\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_m)$
  Generate fake samples $\Psi(\mathbf{z}_1), \Psi(\mathbf{z}_2), \ldots, \Psi(\mathbf{z}_m)$
  for each synthetic image $\Psi(\mathbf{z}_j)$ **do**
    ▷ *Phase II: Private Computation and Aggregation*
    for each teacher model Γ_i **do**
      Sample m data samples $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m)$ from d_i
      Update the teacher discriminator Γ_i by descending its stochastic gradient on $\mathcal{L}_{\Gamma_i}$ on both fake samples and real samples
      Calculate the gradient $\mathbf{g}_j^{(i)} = -\frac{\partial \log \Gamma_i(a)}{\partial a}\Big|_{a=\Psi(\mathbf{z}_j)}$ of the teacher discriminator loss $\mathcal{L}_{\Gamma_i}$ w.r.t. the sample $\Psi(\mathbf{z}_j)$.
    end for
    $\mathbf{g}_j \leftarrow (\mathbf{g}_j^{(1)}, \mathbf{g}_j^{(2)}, \ldots, \mathbf{g}_j^{(N)})$
    $\tilde{\mathbf{g}}_j \leftarrow \texttt{DPTopkAgg}\left(T, \mathbf{g}_j, c, k, \sigma, \beta\right)$
    ▷ *Phase III: Post-Processing*
    $\hat{\mathbf{x}}_j \leftarrow \Psi(\mathbf{z}_j) + \gamma \tilde{\mathbf{g}}_j$
  end for
  Update the student generator Ψ by descending its stochastic gradient on $\hat{\mathcal{L}}_\Psi(\mathbf{z}, \hat{\mathbf{x}}) = \frac{1}{m}\sum_{j=1}^m (\Psi(\mathbf{z}_j) - \hat{\mathbf{x}}_j)^2$ on $\hat{\mathbf{x}} = (\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \ldots, \hat{\mathbf{x}}_m)$
end for

Step 3: DP gradient compression and aggregation. In order to perform efficient DP mechanism for the teacher gradient vectors, we propose TopAgg to compress the teacher gradient vectors first and then aggregate them. We perform gradient aggregation over the teachers' gradient vectors with a corresponding noise injection algorithm that guarantees differential privacy. The final aggregated noisy gradient vector is then passed to the student generator. Details will be discussed in the next subsection.

Step 4: Training the student generator. The student generator learns to improve its synthetic records by back-propagating the aggregated DP gradient vectors produced by the teacher ensemble. We define the loss function for the student generator as $\hat{\mathcal{L}}_\Psi(\mathbf{z}, \hat{\mathbf{x}}) = \frac{1}{m}\sum_{j=1}^m (\Psi(\mathbf{z}_j) - \hat{\mathbf{x}}_j)^2$, where $\mathbf{z}_j$ is the noise sample, $\Psi(\mathbf{z}_j)$ is the synthetic data, and $\hat{\mathbf{x}}_j = \Psi(\mathbf{z}_j) + \gamma\tilde{\mathbf{g}}_j$ is the synthetic data plus the aggregated DP gradient vectors from the teacher discriminators. Since $-\frac{\partial \hat{\mathcal{L}}_\Psi(\mathbf{z},\hat{\mathbf{x}})}{\partial \hat{\mathbf{x}}} = \frac{2\gamma}{m}\sum_{j=1}^m \tilde{\mathbf{g}}_j$, descending the stochastic gradient on $\hat{\mathcal{L}}_\Psi(\mathbf{z}, \hat{\mathbf{x}})$ would propagate the aggregated DP gradient vectors from the teacher discriminators to the student generator.

Top-k Gradient Compression via Stochastic Sign Gradient. In the Step 3. *gradient compression and aggregation*, each teacher model compresses its dense, real-valued gradient vector into a sparse sign vector with k nonzero entries. We first present and discuss the gradient compression function: $\texttt{TopkStoSignGrad}(\mathbf{g}, c, k)$ (Algorithm 2).

Inspired by the recent results on signSGD [7] and gradient compression in communication efficient distributed learning [5, 57], we design a gradient compression algorithm that reduces a gradient vector in two steps. First, we select the top-k dimensions in each teacher gradient $\mathbf{g}$ and set the remaining dimensions to zero. This step reduces the dimensionality of the gradient vector and allows us to achieve a tighter privacy bound during DP gradient aggregation. Then, we clip the gradient at each dimension with threshold c, normalize the top-k gradient vector, and perform stochastic gradient sign quantization. Specifically, we first select the top-k dimensions of the gradient. Let $\hat{g}_j$ be the j-*th* dimension selected from the gradient vector g, we then clip each selected dimension as $\hat{g}_j = \min(\max(\hat{g}_j, -c), c)$. After normalization, we assign the stochastic gradient sign $\tilde{g}_j$ based on the following rule:

$$\tilde{g}_j = \begin{cases} 1, & \text{with probability } \frac{1+\hat{g}_j}{2}; \\ -1, & \text{with probability } \frac{1-\hat{g}_j}{2}. \end{cases} \quad (1)$$

We can see that $\tilde{g}_j$ is an unbiased estimator of $\hat{g}_j$. As a result, we transform a dense, real-valued gradient vector into a sparsified $\{-1, 0, 1\}$-valued vector, which allows more effective differentially private gradient aggregation.

High Dimensional DP Gradient Aggregation. In the gradient aggregation step, we perform differentially private aggregation

Algorithm 2 - Gradient Compression on Top-k Dimensions via Stochastic Sign Gradient (`TopkStoSignGrad`). This algorithm takes in a gradient vector of a teacher model $\mathbf{g}^{(i)}$ and returns the compressed gradient vector $\tilde{\mathbf{g}}^{(i)}$.

Input: Gradient vector $\mathbf{g}^{(i)}$, gradient clipping constant c, top-k
$\mathbf{h}^{(i)} \leftarrow$ arg-topk($|\mathbf{g}^{(i)}|, k$)
▷ *the top-k indices of the absolute value of gradient* $\hat{\mathbf{g}}^{(i)}$
$\mathbf{g}_j^{(i)} = \min(\max(\mathbf{g}_j^{(i)}, -c), c)$ for each dimension j in $\mathbf{g}^{(i)}$
▷ *Clip each dimension of* $\mathbf{g}^{(i)}$ *so that* $-c \le \mathbf{g}_j^{(i)} \le c$.
$\hat{\mathbf{g}}^{(i)} \leftarrow \mathbf{g}^{(i)} / \|\mathbf{g}^{(i)}\|_\infty$ ▷ *gradient normalization to (-1, 1)*
$\tilde{\mathbf{g}}^{(i)} \leftarrow \mathbf{0}$ ▷ *initialization of the compressed sparse gradient vector*
for each top-k index j in $\mathbf{h}^{(i)}$ **do**

$$\tilde{g}_j^{(i)} = \begin{cases} 1, & \text{with probability } \frac{1+\hat{g}_j^{(i)}}{2} \\ -1, & \text{with probability } \frac{1-\hat{g}_j^{(i)}}{2} \end{cases}$$

end for
Return: $\tilde{\mathbf{g}}^{(i)}$

on the compressed teachers' gradient vectors. Specifically, we want to guarantee that the change of any teacher gradient vector will not considerably shift the output distribution of the aggregation. Algorithm 3 presents the aggregation algorithm.

After compression, each gradient vector is a sparse sign vector with k nonzero entries. Therefore, we propose a novel algorithm that converts gradient aggregation into a voting problem. Specifically, the gradient signs can be viewed as votes for the gradient directions. Each teacher can vote for k gradient dimensions. For each dimension in the top-k selection, they vote either the positive direction (*i.e.*, $\tilde{g}_j = 1$) or the negative direction (*i.e.*, $\tilde{g}_j = -1$).

We apply Gaussian mechanism [41] with post-processing thresholding to aggregate the gradient votes. First, we take the sum of the gradient vectors and inject Gaussian noise following distribution $\mathcal{N}(0, \sigma^2)$. Then, we check whether the noisy vote for each gradient direction is greater than a threshold. This thresholding step guarantees that we only select the gradient directions with high agreement rate among the teacher models. To reach an agreement, the following two conditions need to be satisfied. First, the gradient dimension is ranked as top-k for the majority of the teachers. Second, these teachers also agree on the sign of the gradients along these dimensions. With thresholding, we remove the influence of outliers among the teachers. Intuitively, since the selected directions have higher votes, they are unlikely to be changed by the DP noise injection mechanism to preserve utility.

In particular, the Top-k stochastic sign gradient quantization and DP gradient aggregation approaches together form a novel DP gradient compression and aggregation algorithm TopAgg (Algorithm 3), which serves as a key building block in DataLens. These joint operators are the first time to be adopted in a data generated model, and we will provide the convergence analysis for these joint operators in Section 4.3.

4.2 Differential Privacy Analysis for DataLens

In this section, we analyze the differential privacy bound for the proposed DataLens framework, and we leverage the Rényi differential privacy in our analysis. We also compare the data-dependent privacy bound and the data-independent privacy bound, and we show

Algorithm 3 - Differentially Private Gradient Compression and Aggregation (TopAgg). This algorithm takes gradients of teacher models and returns the compressed and aggregated differentially private gradient vector.

Input: Teacher number N, gradient vectors of teacher models $\mathcal{G} = \{\mathbf{g}^{(1)}, \ldots, \mathbf{g}^{(N)}\}$, gradient clipping constant c, top-k, noise parameters σ, voting threshold β
▷ *Phase I: Gradient Compression*
for each teacher's gradient $\mathbf{g}^{(i)}$ **do**
  $\tilde{\mathbf{g}}^{(i)} \leftarrow$ `TopkStoSignGrad`$(\mathbf{g}^{(i)}, c, k)$
end for
▷ *Phase II: Differential Private Gradient Aggregation*
$\tilde{\mathbf{g}}^* \leftarrow \sum_{i=1}^{N} \tilde{\mathbf{g}}^{(i)} + \mathcal{N}(0, \sigma^2)$
▷ *Phase III: Gradient Thresholding (Post-Processing)*
for each dimension $\tilde{g}_j^*$ of $\tilde{\mathbf{g}}^*$ **do**

$$\bar{g}_j = \begin{cases} 1, & \text{if } \tilde{g}_j^* \ge \beta N; \\ -1, & \text{if } \tilde{g}_j^* \le -\beta N; \\ 0, & \text{otherwise.} \end{cases}$$

end for
Return: $\bar{\mathbf{g}}$

the data-independent one is more suitable for analyzing DataLens.

Rényi Differential Privacy. We utilize Rényi Differential Privacy (RDP) to perform the privacy analysis since it supports a tighter composition of privacy budget and can be applied to both data-independent and data-dependent settings. First, we review the definition of RDP and its connection to DP.

Definition 2 ((λ, α)-RDP [41])**.** A randomized mechanism $\mathcal{M}$ is said to guarantee (λ, α)-RDP with $\lambda > 1$ if for any neighboring datasets D and D',

$$D_\lambda\left(\mathcal{M}(D) \| \mathcal{M}\left(D'\right)\right) = \frac{1}{\lambda - 1} \log \mathbb{E}_{x \sim \mathcal{M}(D)} \left[\left(\frac{\mathbf{Pr}[\mathcal{M}(D) = x]}{\mathbf{Pr}\left[\mathcal{M}\left(D'\right) = x\right]} \right)^{\lambda - 1} \right] \le \alpha.$$

For any given probability $\delta > 0$, (λ, α)-RDP implies $(\varepsilon_\delta, \delta)$-differential privacy with ε_δ bounded by the following theorem. The definition of *neighboring dataset* in this work follows the standard definition used in PATE framework [43] and DP-SGD framework [2]. As noted in Abadi et al. [2], the neighboring datasets would differ in a single entry, that is, one image instance is *present* or *absent* in one dataset compared with the other taking image as an example.

Theorem 1 (From RDP to DP [41])**.** *If a mechanism $\mathcal{M}$ guarantees (λ, α)-RDP, then $\mathcal{M}$ guarantees $(\alpha + \frac{\log 1/\delta}{\lambda - 1}, \delta)$-differential privacy for any $\delta \in (0, 1)$.*

In the remaining of this section, we first use RDP to analyze the privacy bound of DataLens, and then derive the final DP bound in Theorem 3. We will first analyze the data-independent and data-dependent privacy bounds.

Data-Independent Privacy Bound. In our PATE based data generative framework, the teacher discriminators have access to the sensitive training data and the student generator learns about the sensitive data from the teachers through the gradient aggregation algorithm. Therefore, if the gradient aggregation algorithm preserves DP or RDP, the same privacy guarantee applies to the student

generator based on the post-processing theorems. Hence, we focus on deriving the privacy bound for the gradient aggregation algorithm (TopAgg).

Let $\tilde{\mathcal{G}} = (\tilde{\mathbf{g}}^{(1)}, \ldots, \tilde{\mathbf{g}}^{(N)})$ be the set of compressed teacher gradient vectors, where $\tilde{\mathbf{g}}^{(i)}$ is the compressed gradient of the i-th teacher. We define sum aggregation function

$$f_{sum}(\tilde{\mathcal{G}}) = \sum_{i=1}^{N} \tilde{\mathbf{g}}^{(i)},$$

and, by applying Gaussian mechanism, we have

$$\tilde{\mathbf{G}}_\sigma f_{sum}(\tilde{\mathcal{G}}) = f_{sum}(\tilde{\mathcal{G}}) + \mathcal{N}\left(0, \sigma^2\right) = \sum_{\tilde{\mathbf{g}} \in \tilde{\mathcal{G}}} \tilde{\mathbf{g}} + \mathcal{N}\left(0, \sigma^2\right).$$

For any real-valued function f, the Gaussian mechanism provides the following RDP guarantee:

Theorem 2 (RDP Guarantee for Gaussian Mechanism [41]). *If f has ℓ_2-sensitivity s, then the Gaussian mechanism $\mathbf{G}_\sigma f$ satisfies $\left(\lambda, s^2\lambda/\left(2\sigma^2\right)\right)$-RDP.*

Thus, to calculate the RDP guarantee for $\tilde{\mathbf{G}}_\sigma f_{sum}(\tilde{\mathcal{G}})$, we first need to calculate the ℓ_2 sensitivity [16] of the aggregation algorithm.

Lemma 1. *For any neighboring top-k gradient vector sets $\tilde{\mathcal{G}}$, $\tilde{\mathcal{G}}'$ differing by the gradient vector of one teacher, the ℓ_2 sensitivity for f_{sum} is $2\sqrt{k}$.*

Proof. The ℓ_2 sensitivity is the maximum change in ℓ_2 norm caused by the input change. For each of the top-k dimension, a teacher could take one of the following two changes: (1) vote for the opposite direction, which flips the gradient sign of one entry; (2) vote for a different dimension, which reduces the vote of one entry and increases the vote on another. The former changes ℓ_2 norm by 2, and the latter by $\sqrt{2}$. In the worst case, the teacher flips all the top-k gradient signs, the change in ℓ_2 norm equals $\sqrt{2^2 k} = 2\sqrt{k}$. □

Theorem 3. *The TopAgg algorithm (Algorithm 3) guarantees $(\frac{2k\lambda}{\sigma^2} + \frac{\log 1/\delta}{\lambda - 1}, \delta)$-differential privacy for all $\lambda \geq 1$ and $\delta \in (0, 1)$.*

Proof. The DPTopkAgg algorithm can be decomposed into applying gradient thresholding on the output of the sum aggregation Gaussian mechanism $\mathbf{G}_\sigma f_{sum}$. $\mathbf{G}_\sigma f_{sum}$ guarantees $(\lambda, 2k\lambda/\sigma^2)$-RDP (Lemma 1 & Theorem 2), and thus this theorem is the result of applying the post-processing theorem of RDP and Theorem 1. □

Data-Dependent Privacy Bound. The parameters ε in Definition 1 and α in Definition 2 are called *the privacy budget* of a randomized mechanism. When ε and α are dependent of the input dataset D, the privacy bound is data-dependent. In the following section, we compare the data-independent privacy bound in Theorem 3 with a data-dependent privacy bound proposed by Papernot et al. [44]. We prove that, when the algorithm has high dimensional outputs, the data-independent privacy bound (Theorem 3) is tighter and achieves better utility.

First, we revisit the data-dependent RDP bound for randomized algorithms [44]:

Theorem 4 (Data-Dependent RDP Bound [44]). *Let $\mathcal{M}$ be a randomized algorithm with (μ_1, α_1)−RDP and (μ_2, α_2)−RDP guarantees and suppose that there exists a likely outcome $\bar{\mathbf{g}}^*$ given a dataset D and a bound $\tilde{q} \leq 1$ such that $\tilde{q} \geq \Pr[\mathcal{M}(D) \neq \bar{\mathbf{g}}^*]$. Additionally, suppose that $\lambda \leq \mu_1$ and $\tilde{q} \leq e^{(\mu_2 - 1)\alpha_2} / \left(\frac{\mu_1}{\mu_1 - 1} \cdot \frac{\mu_2}{\mu_2 - 1}\right)^{\mu_2}$. Then, for any neighboring dataset D' of D, we have:*

$$D_\lambda\left(\mathcal{M}(D) \| \mathcal{M}(D')\right) \leq \frac{1}{\lambda - 1} \log\Big((1 - \tilde{q}) \cdot \boldsymbol{A}(\tilde{q}, \mu_2, \alpha_2)^{\lambda - 1} + \tilde{q} \cdot \boldsymbol{B}(\tilde{q}, \mu_1, \alpha_1)^{\lambda - 1}\Big),$$

where

$$\boldsymbol{A}(\tilde{q}, \mu_2, \alpha_2) \triangleq (1 - \tilde{q}) / \left(1 - (\tilde{q} e^{\alpha_2})^{\frac{\mu_2 - 1}{\mu_2}}\right), \qquad \boldsymbol{B}(\tilde{q}, \mu_1, \alpha_1) \triangleq e^{\alpha_1} / \tilde{q}^{\frac{1}{\mu_1 - 1}}.$$

The parameters μ_1 and μ_2 are optimized to get a data-dependent RDP guarantee for any order λ.

The above data-dependent RDP bound is tighter than the data-independent bound in Theorem 3 when $\tilde{q} \ll 1$. Since $\tilde{q} \geq \Pr[\mathcal{M}(D) \neq \bar{\mathbf{g}}^*]$, the data-dependent bound improves upon the data-independent bound only when the algorithm's output distribution peaks at a likely outcome $\bar{\mathbf{g}}^*$. Papernot et al. [44] demonstrated that the data-dependent privacy bound improves the utility of the PATE framework when teachers vote on one-dimensional predictions. However, we observe that this bound does *not* always guarantee a better utility for algorithms with high dimensional outputs. Specifically, with the increase of the output dimensionality, there is a diminishing benefit from using the data-dependent privacy bound in Theorem 4.

Below, we demonstrate the observation that the data-independent privacy bound can achieve better utility with the aggregation and thresholding steps in TopAgg. Let $\mathcal{M}(\tilde{\mathcal{G}}, N, \beta)$ represent the composition of these two steps, where $\tilde{\mathcal{G}}$ is the compressed gradient vector set, N is the number of teachers, and β is the voting threshold.

Theorem 5. *For any $\bar{\mathbf{g}}^* \in \{0, 1\}^d$, we have*

$$\Pr[\mathcal{M}(\tilde{\mathcal{G}}, N, \beta) \neq \bar{\mathbf{g}}^*] = 1 - \prod_{\{j | \bar{g}_j^* = 1\}} \left(1 - \Phi\left(\frac{\beta N - f_j}{\sigma}\right)\right) \prod_{\{j | \bar{g}_j^* = -1\}} \Phi\left(\frac{\beta N - f_j}{\sigma}\right) \prod_{\{j | \bar{g}_j^* = 0\}} \operatorname{erf}\left(\frac{\beta N - f_j}{\sqrt{2}\sigma}\right)$$

where Φ is the cumulative distribution function of the normal distribution, erf *is the error function, and f_j is the j-th dimension of the gradient vector sum $\sum_{i=1}^{N} \tilde{\mathbf{g}}^{(i)}$ without the noise injection.*

Theorem 5 shows that the bound $\tilde{q} \geq \Pr[\mathcal{M}(\tilde{\mathcal{G}}, N, \beta) \neq \bar{\mathbf{g}}^*]$ increases with the increasing output dimensionality of $\mathcal{M}$. Since the Gaussian mechanism adds independent Gaussian noise along each dimension, this noise flattens out the probability distribution around the likely outcome $\bar{\mathbf{g}}^*$, and consequently reduces the peak probability for $\Pr[\mathcal{M}(\tilde{\mathcal{G}}, N, \beta) = \bar{\mathbf{g}}^*]$. Therefore, when $\mathcal{M}$ has a high dimensional output, it is very unlikely for the distribution of the algorithm's output to have a spike at any certain point (i.e. $\tilde{q} \ll 1$). Since the data-dependent privacy bound improves upon the data-independent bound only when $\tilde{q} \ll 1$, it is unlikely to benefit algorithms with high-dimensional output. Based on this understanding, we use Theorem 3 (the data-independent privacy bound) for the privacy analysis in DataLens. We also provide empirical evaluation of the data-dependent and data-independent privacy bounds in Figure 2 in Section 5.3.

4.3 Convergence Analysis of TopAgg

Why does top-k and sign compression help the DP data generation process? In this section, we provide theoretical analysis on the

convergence to present the *intuition* behind our proposed gradient compression and aggregation algorithm TopAgg. Note that, as directly analyzing the convergence of GAN is technically challenging [40] and beyond the scope of this paper, we focus on an abstract model in which each teacher provides an *unbiased* gradient estimator for SGD with loss function $F_n(x)$ given input x. We believe that this is a plausible assumption since in our setting each teacher has access to a random non-overlapped partition of the input data.

Understanding the convergence behavior of stochastic gradient descent in the context of differential privacy is a challenging problem. At the first glance, the DP noise might look like just another variance term over the stochastic gradient; however, it is the other operations such as the **normalization** and **clipping** of gradients that make the analysis much harder. In fact, it is not until recently [15, 45, 53] that researchers developed some results to analyze the behavior of DP-SGD with gradient *norm* clipping (often limited to scaling L^2 norm instead of truncating). In our context, this problem becomes even more challenging, as we need to consider not only *element-wise* gradient clipping, but also top-K compression, an operator that introduces *bias*, instead of *variance* to our gradient estimator.

Setup and Assumptions. We focus on the following setting in which our goal is to minimize $f(x) = \frac{1}{N}\sum_{n\in[N]} F_n(x)$ over $\mathbb{R}^d$. Recall that the update rule is

$$x_{t+1} = x_t - \frac{\gamma}{N}\sum_{n\in[N]}\left(Q(\texttt{clip}(\texttt{top-k}(F'_n(x_t)), c), \xi_t) + \mathcal{N}(0, Ak)\right), \quad (2)$$

for some constant $A > 0$ and a clipping constant $c > 0$, with clipping performed *coordinate-wise*. Here we rephrased the stochastic sign quantization using $Q(x,\xi) = \xi(x)$, when $x \geq 0$, and $Q(x,\xi) = -\xi(-x)$, when $x < 0$, where $\xi(x) \sim Ber(x)$, element-wise. Thus the term $Q(\texttt{top-k}(F'_n(x_t)), \xi_t)$ is equivalent to Algorithm 2, which takes in a gradient vector of a teacher model's gradient $F'_n(x_t)$ and returns the compressed gradient vector. Furthermore, $\mathcal{N}(0, Ak)$ is the noise added to ensure differential privacy, since we know from Theorem 3 that when DPTopkAgg satisfies (λ, α)-RDP, the variance of Gaussian noise is $\sigma^2 = 2k\lambda/\alpha$, which is proportional to k.

Following previous work, we make a set of standard assumptions [5, 49, 57]. We assume that f has L-Lipschitz gradient, that all F_n are smooth and that we have a bounded gradient, meaning that there exists $M > 0$ such that $\frac{1}{N}\sum_{n\in[N]}\left\|F'_n(x)\right\|^2 \leq M^2$. Furthermore, we assume bounded stochastic variance per coordinate, meaning that for every $i \in [d]$ there exists $\sigma_i > 0$ such that $\frac{1}{N}\sum_{n\in[N]}|F'_n(x) - \nabla f(x)|^2 \leq \sigma_i^2$, for all $x \in \mathbb{R}^d$. With respect to compression, we see that Q is unbiased in our case, i.e. $\mathbb{E}_\xi[Q(x,\xi)] = x$, for all x, and of bounded variance, i.e. $\mathbb{E}_\xi\left[\|Q(x,\xi) - x\|^2\right] \leq \tilde{\sigma}^2$, for some $\tilde{\sigma} > 0$, and all x. Finally, with respect to top-k, we assume (see [5]) that there exists a non-increasing sequence $1 \geq \tau_1 \geq \ldots \geq \tau_d = 0$, such that for all $k \in [d]$ and all $x \in \mathbb{R}^d$, one has $\|F'_n(x) - \texttt{top-k}(F'_n(x))\| \leq \tau_k\|F'_n(x)\|$. Given these assumptions, we have the following result:

Theorem 6. *(Convergence of top-k Mechanism with/without Gradient Quantization) Suppose that the above assumptions hold, and let $k \in [d]$. Then after T updates using the learning rate γ, one has*

$$\left(\frac{\min\{c,1\}}{d+2}\right)\frac{1}{T}\sum_{t\in[T]}\min\{\mathbb{E}\|\nabla f(x_t)\|^2, \mathbb{E}\|\nabla f(x_t)\|_1\}$$
$$\leq \min\{\tau_k M^2, c(d-k)M\} + L\gamma Ak + (f(x_0) - f(x^*))/(T\gamma)$$
$$+ \max\{\|\sigma\|^2 + \|\sigma\|M, 2\|\sigma\|_1\} + 2L\gamma(\tilde{\sigma}^2 + \min\{c^2, M^2\}). \quad (3)$$

Moreover, if no quantization is used, i.e. $Q(x,\xi) = x$ for all x, then one can improve the last term to $L\gamma\min\{c^2, M^2\}$.

Proof Sketch. The full proof is given in Appendix D, whereas here we explain main ingredients. Intuitively, clipping gradients yields a dichotomy between gradient performing as the usual gradient descent versus the signed gradient descent (as in [9]) of magnitude c. We start with a well-known fact that f having L-Lipschitz gradients implies $f(x_{t+1}) - f(x_t) \leq \langle\nabla f(x_t), x_{t+1} - x_t\rangle + \frac{L}{2}\|x_{t+1} - x_t\|^2$, which allows one to look at the convergence rate step by step. Upon inserting the update rule (2), we split the argument into two cases based on, for $i \in [d]$ and $A_i := \{n \in [N] : |F'_n(x)| \geq c\}$,

$$\texttt{clip}(F'_n(x)_i, c) = c \cdot \texttt{sign}(F'_n(x)_i) \cdot \mathbf{1}\{n \in A_i\} + F'_n(x) \cdot \mathbf{1}\{n \notin A_i\}.$$

Using a proof by contradiction, we show that the error terms cannot beat the main term for clipped and non-clipped gradients simultaneously. In doing so, the error terms on the RHS of (3) originate from the following: $\min\{\tau_k M^2, c(d-k)M\}$ comes from applying the top-k mechanism on top of clipped gradients, $2L\gamma Ak$ originates from the variance of the noise attributed to differential privacy, $f_{0,*}/T\gamma$ comes from the telescoping property when summing over all steps. The term $\max\{\|\sigma\|^2 + \|\sigma\|M, 2\|\sigma\|_1\}$ comes from the clipping dichotomy (also contributing to the term $\min\{c, 1\}$ on the LHS), whereas $2L\gamma(\tilde{\sigma}^2 + \min\{c^2, M^2\})$ is the variance of quantization step. The without quantization case follows the similar approach, up to the non-existence of randomness in the quantization case, yielding a simpler proof.

Discussion: Why Does Top-K Help? The above result depicts the following tradeoff. As k gets *smaller* the error caused by top-k quantization gets larger, leading to two effects:

(1) The term $\min\{\tau_k M^2, c(d-k)M\}$, introduced through the *bias* of top-k compression, gets larger;
(2) The $2L\gamma Ak$ term, introduced by the differential privacy noise, however, gets smaller.

Given a finite number of iterations T, in the worst case the bias introduced through the term τ_k dominates when the gradients are evenly distributed over coordinates, yielding that the top-k compression can significantly slow down the convergence rate in the worst case. However, previous works [5, 57] empirically verify that under certain real distribution of gradient dimensions, the top-k compression does not introduce a large bias, yielding justification for top-k compression, especially when the original dimension d is of very high dimension. For example, if we assume that the gradient follows the *Weibull distribution* $W(\rho_1, \rho_2)$, for some $\rho_1 > 0$ and $0 < \rho_2 < 1$, following recent work in gradient compression [18], then τ_k are, on expectation, distributed as $\tau_k \propto \exp(-(k/\rho_1 d)^{\rho_2}) - \exp(-1)$, which for small ρ_2 grows significantly slower than the contribution of the noise due to differential privacy (linear in k) decreases, as k decreases. Thus, the convergence-privacy tradeoff for algorithm TopAgg can be clearly characterized. It is obvious that given the convergence guarantee, the compression step could save the privacy budget and therefore improve the utility (i.e. smaller DP noise is

added) for training on high-dimensional data, as long as the chosen k is not too small.

4.4 Discussion: TopAgg for SGD Training

In addition to the DP generative model, the proposed DP gradient compression and aggregation algorithm TopAgg, which is a key building block of DataLens, is also generalizable for the standard DP SGD training by applying the gradient compression and aggregation in the DP SGD training process. However, since the DP SGD algorithm has already achieved high data utility, the improvement with TopAgg is empirically marginal, and we will defer the details on how to adapt TopAgg to training a differentially private deep neural network and the corresponding evaluation in Appendix A.

5 EXPERIMENTAL EVALUATION

In this section, we present the experimental evaluation of DataLens for generating differentially private data with high utility. We compare DataLens with state-of-the-art differentially private generative models and evaluate the data utility and visual quality on high-dimensional image data such as CelebA face and Places365 to demonstrate the effectiveness and scalability of DataLens.

5.1 Experimental Setup

We compare the generated data utility of DataLens with three state-of-the-art baselines: DP-GAN [59], PATE-GAN [62], GS-WGAN [13], and G-PATE [37] on four image datasets.

Datasets. To demonstrate the advantage of DataLens as being able to generate high dimensional differentially private data, we focuse on high dimensional image datasets, including MNIST [30], Fashion-MNIST [58], CelebA datasets [36], and Places365 dataset [65]. MNIST and Fashion-MNIST dataset contain grayscale images of 28×28 dimensions. Both datsets have 60,000 training examples and 10,000 testing examples. The CelebA dataset contains 202,599 color images of celebrity faces. We use the official preprocessed version with face alignment and resize the images to $64 \times 64 \times 3$. Places365 dataset is consisted of 1.8M high resolution color images of diverse scene categories. We select three level-2 classes to compose a dataset of size 120,000 and resize the images to $64 \times 64 \times 3$.

We create two CelebA datasets based on different attributes: CelebA-Gender is a binary classification dataset with gender as the label, while CelebA-Hair uses three hair color attributes (black/ blonde/ brown) as classification labels. The training and testing set is split following the official partition as [36]. Since DP-GAN and PATE-GAN did not evaluate their framework on high dimensional image datasets, we run their open-source code and compare with the proposed DataLens framework.

Models. Both the teacher discriminator and the student generator of DataLens uses the same architecture as DC-GAN [47]. The latent variables sampled from Gaussian distribution are 50-dimensional for MNIST, 50-dimensional ($\varepsilon = 1$) and 64-dimensional ($\varepsilon = 10$) for Fashion-MNIST, 100-dimensional for CelebA datasets, and 100-dimensional for Places365. For $\varepsilon = 1$, we set top-k=200 for MNIST and Fashion-MNIST, top-k=700 for CelebA and Places365. For $\varepsilon = 10$, we set top-k=350 for MNIST and Fashion-MNIST, top-k=500 for CelebA, and top-k=700 for Places365. Ablation studies and discussions on comprehensive hyper-parameter analysis can be found in Section 5.3.

Baselines. For baseline models, DP-GAN uses standard WGAN and adds Gaussian noise on the gradients during training to achieve differential privacy. Both PATE-GAN and G-Pate leverage PATE framework to generate differentially private images based on different teacher aggregation strategies. Since DP-GAN and PATE-GAN did not evaluate or report their frameworks on (high-dimensional) image datasets, we run their open-source code of DP-GAN[1] and PATE-GAN[2] and compare with our DataLens framework. For GS-WGAN, we use its open-source implementation[3] to train DP generative models. For large $\varepsilon = 10$, we can reproduce the performance on MNIST and Fashion-MNIST. Under small $\varepsilon = 1$ setting, we tried our best to tune the hyper-parameters of GS-WGAN; however, we observe GS-WGAN is unable to converge given the limited privacy constraints, especially when presented with higher-dimensional data (CelebA, Places365) which is confirmed with the authors.

Evaluation Metrics. We follow standard evaluation pipelines [13, 37, 62] and evaluate DataLens as well as baselines in terms of *data utility* and *visual quality* under different privacy constraints. Specifically, *data utility* is evaluated by training a classifier with the generated data and testing the classifier on real test dataset. We consider the *testing accuracy* on the test set as the indicator for the utility of the synthetic data for downstream tasks. To evaluate the visual quality of generated data for understanding purpose, we consider *Inception Score (IS)* [31] and *Frechet Inception Distance (FID)* [50], which are standard metrics of visual quality in GAN literature. We also provide the images generated by DataLens in Appendix Figure 4 for visualization.

5.2 Experimental Results

In this section, we evaluate DataLens on different datasets. We first compare the generated data utility for DataLens and four other state of the art DP generative model baselines. We then explore the performance of DataLens under limited privacy budgets (*i.e.*, $\epsilon < 1$), which is a challenging while important scenario. We then evaluate the visual quality of the generated data, followed by a range of ablation studies on the data-dependent and data-independent privacy analysis, impacts of different hyper-parameters and components in DataLens, as well as different compression methods. We show that the proposed DataLens not only outperforms all baselines, but also demonstrates additional advantages especially when the privacy budget is small.

Data Utility Evaluation. We first compare DataLens with four baselines under two privacy budget settings $\varepsilon = 1, \delta = 10^{-5}$ and $\varepsilon = 10, \delta = 10^{-5}$ on five high dimensional image datasets, following the standard evaluation pipeline.

From Table 1, we can see that DataLens shows substantially higher performance than all baseline methods especially when $\varepsilon = 1$. In particular, the performance improvement on MNIST under $\varepsilon = 1$ is more than 13%. Even for high dimensional datasets like CelebA-Hair and Places365 whose dimensionality is 16 times larger than MNIST, DataLens achieves 10% higher performance improvement than the state of the art, which demonstrates its advantages

[1] Code at https://github.com/illidanlab/dpgan
[2] Code at https://bitbucket.org/mvdschaar/mlforhealthlabpub/src/master/alg/pategan/
[3] Code at https://github.com/DingfanChen/GS-WGAN

Table 1: Performance of different differentially private data generative models on Image Datasets: Classification accuracy of the model trained on the generated data and tested on real test data under different ε ($\delta = 10^{-5}$).

Dataset \ Methods	DC-GAN ($\varepsilon = \infty$)	ε	DP-GAN	PATE-GAN	G-PATE	GS-WGAN	DataLens
MNIST	0.9653	$\varepsilon = 1$	0.4036	0.4168	0.5810	0.1432	**0.7123**
		$\varepsilon = 10$	0.8011	0.6667	**0.8092**	0.8075	0.8066
Fashion-MNIST	0.8032	$\varepsilon = 1$	0.1053	0.4222	0.5567	0.1661	**0.6478**
		$\varepsilon = 10$	0.6098	0.6218	0.6934	0.6579	**0.7061**
CelebA-Gender	0.8149	$\varepsilon = 1$	0.5330	0.6068	0.6702	0.5901	**0.7058**
		$\varepsilon = 10$	0.5211	0.6535	0.6897	0.6136	**0.7287**
CelebA-Hair	0.7678	$\varepsilon = 1$	0.3447	0.3789	0.4985	0.4203	**0.6061**
		$\varepsilon = 10$	0.3920	0.3900	0.6217	0.5225	**0.6224**
Places365	0.7404	$\varepsilon = 1$	0.3200	0.3238	0.3483	0.3375	**0.4313**
		$\varepsilon = 10$	0.3292	0.3796	0.3883	0.3725	**0.4875**

Table 2: Performance Comparison of different differentially private data generative models on Image Datasets under small privacy budget which provides strong privacy guarantees ($\varepsilon \leq 1$, $\delta = 10^{-5}$).

ε	MNIST					Fashion-MNIST				
	DP-GAN	PATE-GAN	G-PATE	GS-WGAN	DataLens	DP-GAN	PATE-GAN	G-PATE	GS-WGAN	DataLens
0.2	0.1104	0.2176	0.2230	0.0972	**0.2344**	0.1021	0.1605	0.1874	0.1000	**0.2226**
0.4	0.1524	0.2399	0.2478	0.1029	**0.2919**	0.1302	0.2977	0.3020	0.1001	**0.3863**
0.6	0.1022	0.3484	0.4184	0.1044	**0.4201**	0.0998	0.3698	0.4283	0.1144	**0.4314**
0.8	0.3732	0.3571	0.5377	0.1170	**0.6485**	0.1210	0.3659	0.5258	0.1242	**0.5534**
1.0	0.4046	0.4168	0.5810	0.1432	**0.7123**	0.1053	0.4222	0.5567	0.1661	**0.6478**

on high dimensional data than other baseline DP generative models. Specifically, we note that GS-WGAN can only converge under large privacy budget ($\varepsilon = 10$) for gray-scale datasets (MNIST and FashionMNIST), as GS-WGAN needs 20k epochs and small noise to converge. In comparison, DataLens can converge within 100 epochs due to the fast convergence rate brought by top-k operation for high-dimensional datasets. As a result, under the limited privacy budget ($\varepsilon = 1$) or given high dimensional facial datasets (*e.g.*, CelebA), GS-WGAN is unable to converge and therefore generate low-utility data, making the classifier accuracy close to random guessing; while DataLens can generate high-utility data even with limited privacy budget.

Evaluation under small privacy budget. To further demonstrate the advantage of DataLens as being able to generate high-utility images under *small* privacy budgets (*i.e.*, higher privacy protection guarantees), we conduct ablation studies on MNIST and Fashion-MNIST under $\varepsilon \leq 1$. The experimental results are shown in Table 2.

We find that DataLens achieves the best results compared with the baselines given such tight privacy constraints. With increasing privacy budgets, different DP models gradually converge and the accuracy increases. We note that DataLens converges the fastest and achieves more than 20% accuracy even under smallest privacy budget $\varepsilon = 0.2$ for both MNIST and Fashion-MNIST datasets; while the baseline models barely converge and the accuracy is similar to random guess. The observed experimental results also support our theoretical analysis that the proposed TopAgg algorithm can introduce a smaller bias and provide high-utility gradient information for the student generator to converge, demonstrating that our method is particularly effective under limited privacy budgets.

Visual Quality Evaluation. We present the quantitative visual quality evaluation of DataLens and baselines in Table 3 based on Inception Score (IS) under different privacy constraints: $\varepsilon = 1, \delta = 10^{-5}$ and $\varepsilon = 10, \delta = 10^{-5}$. Since CelebA-gender and CelebA-face are from the same distribution of face images and have a lot of overlapping, we mainly consider CelebA-Gender to represent the visual quality for CelebA face dataset.

We observe that DataLens consistently outperform the baselines in terms of visual quality while ensuring the rigorous privacy protection when $\varepsilon = 1$, which suggests that DataLens can converge faster than the state-of-the-art baselines. Specifically, the generated differentially private MNIST images achieve the inception score of 4.37, improving the strongest baseline G-PATE by more than 20%. When $\varepsilon = 10$, we find that DataLens can be outperformed by GS-WGAN on MNSIT and Fashion-MNIST, but still outperforms all baselines on high-dimensional CelebA datasets. We believe the reason is that while top-k operation can help with faster converge the most important information and yield high-utility data, it may lose some detailed and trivial gradient information for image reconstruction. We note that visual quality and data utility are two orthogonal metrics, and DataLens consistently generates data with the highest utility. We provide the evaluation of FID in Appendix Table 12, given that FID is evaluated based on models trained with ImageNet which may not be suitable for evaluating datasets such

Table 3: Quality evaluation of images generated by different differentially private data generative models on Image Datasets: we use Inception Score (IS) to measure the visual quality of the generated data under different ε ($\delta = 10^{-5}$).

Dataset	Real data	ε	DP-GAN	PATE-GAN	G-PATE	GS-WGAN	DataLens
MNIST	9.86	1	1.00	1.19	3.60	1.00	**4.37**
		10	1.00	1.46	5.16	**8.59**	5.78
Fashion-MNIST	9.01	1	1.03	1.69	3.41	1.00	**3.93**
		10	1.05	2.35	4.33	**5.87**	4.58
CelebA	1.88	1	1.00	1.15	1.11	1.00	**1.18**
		10	1.00	1.16	1.12	1.00	**1.42**

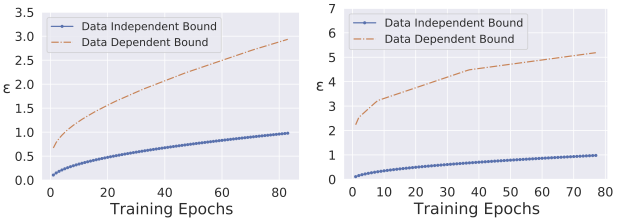


Figure 2: Ablation studies on the data dependent bound v.s. data independent bound on MNIST (left) and CelebA-Hair (right). The data independent bound always yields tighter privacy bound than the data dependent analysis, given high dimensionality of gradients.

as MNIST. In addition, we believe it would be an interesting future direction to add additional loss terms for improving the visual quality of the generated data with privacy guarantees.

5.3 Ablation Studies

In this section, we conduct a series of ablation studies to further understand the improvements of DataLens, including the empirical exploration of the data-dependent and data-independent privacy bounds, the hyper-parameter impacts, the comparison with different gradient compression methods, as well as the impacts of each component in DataLens pipeline.

Data-Independent Bound v.s. Data-Dependent Bound. We compute the data-independent privacy bound and the data dependent privacy bound to validate the theoretical comparison in Section 4.2. Figure 2 presents the privacy budget consumption over each training epoch computed by the data-independent bound and the data-dependent bound, respectively. We set $\sigma = 5000$ for MNIST and Fashion-MNIST, and $\sigma = 9000$ for CelebA-Hair and CelebA-Gender. The training is stopped when the privacy budget ε computed by the data independent bound reaches 1. As shown in Figure 2, the data-independent bound is always tighter than the data-dependent one on the high-dimensional datasets. We also notice that models on MNIST and Fashion-MNIST have a similar data-dependent bound, and so are models on CelebA-Hair and CelebA-Gender. These results align with our theoretical analysis in Theorem 4 and Theorem 5. Due to the high dimensionality of the gradients and the Gaussian noise, there is unlikely to be a spike in the probability distribution over the likely outcomes of the gradient aggregation step. Consequently, the data-dependent privacy bound is loose and mostly determined by the dimension of the gradients.

Ablation studies on hyper-parameters. As we can see, DataLens contains several hyper-parameters: the number of the teacher models, the top-k, the threshold β, the standard deviation σ of injected Gaussian noise, and the gradient clipping constant c. We evaluate a set of hyper-parameters as shown in Table 4. For other parameters, we use: for MNIST and Fashion-MNIST datasets, we set $\sigma = 5000$ when $\varepsilon = 1$ and $\sigma = 900$ when $\varepsilon = 10$; for CelebA datasets of higher dimensionality, we set $\sigma = 9000$ when $\varepsilon = 1$ and $\sigma = 700$ when $\varepsilon = 10$. We set the gradient clipping constant $c = 10^{-5}$ in all experiments. We also follow the default DC-GAN model configuration[4] and set the batch size the same as the disjoint data partition size.

From Table 4, we observe that training more teacher discriminators will give us better performance in general, as it can save more privacy budgets. However, with more teacher discriminators, each discriminator will have access to a smaller amount of training data, thus leading to a slightly worse performance. We observe this trade-off on the CelebA-Gender dataset, where the optimal number of teachers is 6000. Choosing a proper top-k and β is bit tricky: as stated in the discussion of Section 4.3, if top-k is too small, the model converges slower and is likely to converge to a bad solution. On the other hand, if top-k is too large, we will introduce a larger DP noise and the model can soon reach the privacy budget limit given the high sensitivity. In this paper, we search for the best top-k via grid search. Another observation is that we usually need to set a high threshold β to smooth out the noisy gradient entries from the DP noise, though if the threshold is too high it is likely to ignore the top-k voted gradients information. This tradeoff leads us to choose a threshold β between $\frac{\sigma}{2N}$ and $\frac{\sigma}{N}$. Finally, we also note that the clipping value c has a large impact on the model convergence. We observe given a fixed top-k, a reasonably smaller c generally yields better convergence rate and data utility. This is aligned with our theorem, because if c gets smaller, the convergence bias from the first term $c(d-k)M$ will get smaller.

We note that the peformance improvements from DataLens does not necessarily comes from the fact that we have more hyper-parameters, since compared to other baseline methods using PATE framework such as G-PATE and PATE-GAN, DataLens only introduces one more hyper-parameter top-k for gradient compression. Moreover, as shown in Table 4, DataLens can outperform all the baselines on CelebA datasets over a wide range of different hyper-parameters in practice.

Ablation Studies on the Gradient Compression Methods. Here we analyze the impact of our top-k gradient compression method in TopAgg compared with other compression methods in previous works, *e.g.*, D^{2}P-Fed and FetchSGD. In particular, for D^{2}P-Fed, we replace our Algorithm 2 (`TopkStoSignGrad`) that uses stochastic sign compression with it, which essentially uses k-level gradient quantization and random rotation for gradient pre-processing. The detailed algorithm is shown in Algorithm 6 and Algorithm 5 in Appendix C.2. For FetchSGD, we uses the same stochastic sign compression as we leverage sign signal as teacher voting in PATE framework. During aggregation, we use Count Sketch data structure, and use top-k and unsketch operation to retrieve the aggregated gradient. The detailed algorithm is shown in Algorithm 7

[4]Details can be found at https://github.com/carpedm20/DCGAN-tensorflow.

Table 4: Impact of different hyper-parameters: the number of Teachers, top-k and threshold β under $\varepsilon = 1$ and $\delta = 10^{-5}$. We search for the optimal parameter combinations and report the best accuracy by controlling the parameter in each cell.

(a) Hyper-parameters Search for MNIST and Fashion-MNIST

	Top-k			# of Teachers		
	100	200	300	2000	3000	4000
MNIST	0.5889	**0.7123**	0.6753	0.5841	0.7061	**0.7123**
Fashion	0.5738	**0.6478**	0.6088	0.5608	0.5952	**0.6478**
β	0	0.1	0.3	0.5	0.7	0.9
MNIST	0.6361	0.6450	0.6890	0.6921	**0.7123**	0.6956
Fashion	0.5859	0.6103	0.6060	0.6122	0.6213	**0.6478**

(b) Hyper-parameters Search for CelebA-Hair and CelebA-Gender

	Top-k			# of Teachers		
	500	700	900	4000	6000	8000
CelebA-Gender	0.6922	**0.7058**	0.6811	0.6378	**0.7058**	0.6936
CelebA-Hair	0.5792	**0.6061**	0.5769	0.5669	0.5835	**0.6061**
β	0.5	0.6	0.7	0.8	0.85	0.9
CelebA-Gender	0.6440	0.6789	0.6922	0.6861	**0.7058**	0.6381
CelebA-Hair	0.4957	0.5669	0.5612	0.6022	0.5835	**0.6061**

Table 5: Accuracy Comparison of different gradient compression methods (TopAgg, D^{2}P-Fed, FetchSGD). We report the test classification accuracy of models trained with data generated with each technique under $\varepsilon = 1$ and $\delta = 10^{-5}$.

Dataset \ Methods	TopAgg	D^{2}P-Fed	FetchSGD
MNIST	**0.7123**	0.1424	0.6935
Fashion-MNIST	**0.6478**	0.1667	0.6387
CelebA-Gender	**0.7058**	0.4445	0.6552
CelebA-Hair	**0.6061**	0.2893	0.4926

in Appendix C.2. From Table 5, we note that D^{2}P-Fed and FetchSGD are outperformed by our TopAgg in terms of data utility, which is mainly due to the increase of the consumption of privacy budget and the introduction of additional noise during aggregation. Concretely, D^{2}P-Fed uses m-level gradient quantization, which increases the sensitivity of quantized gradients from $2\sqrt{k}$ to $m\sqrt{k}$. Without top-k mechanism, D^{2}P-Fed compression quickly reaches the limit of the privacy budget, and thus the model barely converges. Although FetchSGD uses a similar top-k mechanism during compression, the adoption of Count Sketch data structure introduces additional noisy information when approximating the aggregated gradient, and therefore hurts the utility of the generated data.

Moreover, we record the running time of DataLens and adapted gradient compression methods D^{2}P-Fed and FetchSGD on one Tesla T4 GPU under the best parameters of MNIST, Fashion-MNIST, CelebA-Hair, and Celeb-Gender. The average running for each epoch is shown in Table 6. The time consumption for D^{2}P-Fed is significantly higher than TopAgg due to the k-level quantization step as well as the rotation step for gradient transformation. The time consumption for FetchSGD is significantly higher than TopAgg due to the Count Sketch data structure overhead.

Runtime Analysis. We record the running time of our framework on one RTX-2080 Ti GPU under the best parameter (4000 teacher) of MNIST for $\varepsilon = 1$ for three runs. We then only change the number of teacher discrinimators to 2000 and record the running time again. The average running time for each epoch given different teacher discriminators are 149.92s for 2000 teachers and 322.17s for 4000 teachers, respectively. The student generator converges within 100 epochs, thus the total training time is around $4-8$ hours for MNIST under $\varepsilon = 1$. The runtime scales almost **linear** to the number of teachers, so adopting a larger number of teachers will not bring much computation overhead. In contrast, the average training time for DP-GAN and G-PATE takes around $26-34$ hours for MNIST under $\varepsilon = 1$. Moreover, GS-WGAN requires hundreds of GPU hours to pretrain one thousand non-private GAN as the warm-up steps.

Table 6: Running Time Comparison of different gradient compression methods (TopAgg, D^{2}P-Fed, FetchSGD). We report the average training time per epoch on different datasets under $\varepsilon = 1$ and $\delta = 10^{-5}$.

Dataset \ Methods	TopAgg	D^{2}P-Fed	FetchSGD
MNIST	338.34 s	492.43s	785.34 s
Fashion-MNIST	**340.84s**	471.02s	775.35s
CelebA-Gender	**1196.60s**	3683.22s	2622.40s
CelebA-Hair	**1120.59s**	8092.50 s	2620.63s

Table 7: Ablation studies on the impact of different components of DataLens pipeline on Image Datasets: We report the test classification accuracy of models trained with data generated based on different variants of DataLens under $\varepsilon = 1$, $\delta = 10^{-5}$. The first row of each data groups presents the performance of DataLens.

Dataset \ Component	Top-k	Stochastic Quantization	Aggregation Thresholding	Accuracy
MNIST	✓	✓	✓	**0.7123**
	✗	✓	✓	0.5170
	✓	✗	✓	0.6741
	✓	✓	✗	0.6361
Fashion-MNIST	✓	✓	✓	**0.6478**
	✗	✓	✓	0.4775
	✓	✗	✓	0.6159
	✓	✓	✗	0.5859
CelebA-Gender	✓	✓	✓	**0.7058**
	✗	✓	✓	0.6134
	✓	✗	✓	0.6889
	✓	✓	✗	0.6860
CelebA-Hair	✓	✓	✓	**0.6061**
	✗	✓	✓	0.3318
	✓	✗	✓	0.5325
	✓	✓	✗	0.5504

Ablation Studies on the Impact of Different Components in DataLens. To further understand where the improvements

of DataLens come from, we investigate how each component in DataLens pipeline contributes to the generated data utility improvement in Table 7 on four high-dimensional image datasets.

In particular, we consider the following components: (1) top-k, (2) stochastic gradient quantization, and (3) gradient thresholding, and evaluate how they impact the data utility by adding or removing each component. We note that the top-k procedure is the most important component based on results in Table 7, since removing this step will largely increase the privacy consumption, leading models fail to converge when given limited privacy budget. Gradient quantization and thresholding are also useful techniques though less critical, contributing to the $3\% - 7\%$ of the utility improvement as shown in Table 7.

6 RELATED WORK

DP Generative Models. In order to generate data with differential privacy guarantees, several works have been conducted to develop DP generative models for low-dimensional data such as tabular data. Some of them apply differential privacy to traditional data generation algorithms, such as Bayesian networks [63], synthetic data generation from marginal distributions [46], and the multiplicative weights approach [20]. Although these methods have demonstrated good performances on low dimensional datasets, they suffer from either low data utility or high sampling complexity on high dimensional data, and therefore they are usually not suitable for the high-dimensional image datasets discussed in this paper.

Another line of work adapts DP-SGD to GAN. DPGAN [59] achieves differential privacy by adding Gaussian noise to the discriminator gradients during the training process. DP-CGAN [54] uses a similar approach to guarantee DP and trains a conditional GAN to generate both synthetic data and labels. GS-WGAN [13] uses the Wasserstein loss and sanitizes the data-dependent gradients of the generator to improve data utility. However, these approaches still suffer from low data utility when applied to high dimensional datasets due to privacy budget explosion.

PATE-GAN [62] combines the PATE framework with GAN. It trains multiple teacher discriminators and uses them to update the student discriminator. However, in this framework, it essentially applies PATE to train the discriminator within a GAN. Both the teacher and students models are discriminators and the interaction between the generator and discriminator is not adapted for the teacher-student framework. Thus, PATE-GAN is also only evaluated on low dimensional tabular data and suffers the similar problem under limited privacy budget. G-PATE [37] improves upon PATE-GAN by directly training a student generator using the teacher discriminators. It uses the random projection algorithm to reduce the gradient dimension during training, which is challenging to analyze its convergence. By combining the PATE framework with top-k gradient compression, DataLens demonstrates a significant utility improvement upon PATE-GAN and G-PATE on high dimensional datasets, with theoretical analysis on its convergence.

DP SGD Training. DPDL [2] is the first work that applies the notion of Differential Privacy to the SGD training to prevent deep neural models from exposing private information of training data. DPDL also proposes to compute the privacy cost of the training by moments accountant, which proves to be a tighter bound than the strong composition theorem. McMahan et al. [39] adopts the notion of Rényi differential privacy, which extends and generalizes the moment accountant to multi-vector queries. Thus, the Rényi differential privacy analysis enables the framework to provide privacy for heterogeneous sets of vectors and is widely adopted by current open-source DP library (Tensorflow Privacy and Pytorch Opacus) implementation. However, the high-dimensional data issue is still present in these algorithms given the fact that the privacy budget can be consumed quickly when aggregating these gradients in a differentially private manner.

Gradient Compression. Communication efficient distributed learning has attracted intensive interests recently. Popular techniques include gradient compression [4, 5, 49, 57], decentralization [28, 33], and asynchronization [32] (see [6]). The essence of these methods is to reason about the *noise* introduced via relaxations in the system design. cpSGD [3] is proposed as a binomial DP-mechanism specifically designed for stochastic k-level gradient quantization [38] to allow low-precision communication after adding DP noises. Extending this work, D^2P-Fed [56] instead applies the discrete Gaussian mechanism to the same k-level quantization and achieves a stronger privacy guarantee. Similarly, Kairouz et al. [25] combine discrete Gaussian mechanism with k-level quantization to facilitate federated learning with differential privacy and secure aggregation. In comparison, DataLens uses PATE framework to give rigorous privacy guarantee and apply sign compression as teacher voting to save privacy budget. FetchSGD [48] focuses on communication-efficiency in the federated learning setting, and proposes Count Sketch data structure and top-k operation for fast gradient compression and aggregation. However, FetchSGD lacks the discussion for privacy guarantee. In this paper, we explore the relationship between privacy and gradient compression in a different scenario and illustrate how gradient compression can help to achieve better utility in privacy preserving algorithms over high dimensional data. We propose TopAgg by combining stochastic sign [24] with top-k gradient compression. Our empirical results show that TopAgg outperforms state-of-art gradient compression algorithms on improving model utility with differential privacy guarantee.

7 CONCLUSION

Overall, we propose a novel and effective differentially private data generative model DataLens, which is applicable to high-dimensional data compared with existing approaches. In addition, we propose a novel algorithm TopAgg to perform gradient compression and aggregation. We provide the DP analysis as well as convergence analysis for the proposed model. Extensive empirical experiments demonstrate that DataLens substantially outperforms the existing DP generative models on different especially high-dimensional image datasets, even under limited privacy budget.

ACKNOWLEDGEMENT

We thank the anonymous reviewers for their constructive feedback. We also thank Dingfan Chen and many others for the helpful discussion. This work is partially supported by the NSF grant No.1910100, NSF CNS 20-46726 CAR, and Amazon Research Award.

REFERENCES

[1] 2020. Opacus – Train PyTorch models with Differential Privacy. https://opacus.ai/

[2] Martín Abadi, Andy Chu, Ian J. Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep Learning with Differential Privacy. *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security* (2016).

[3] Naman Agarwal, Ananda Theertha Suresh, Felix Xinnan X Yu, Sanjiv Kumar, and Brendan McMahan. 2018. cpSGD: Communication-efficient and differentially-private distributed SGD. In *Advances in Neural Information Processing Systems*.

[4] Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. 2017. QSGD: Communication-Efficient SGD via Gradient Quantization and Encoding. In *Advances in Neural Information Processing Systems*.

[5] Dan Alistarh, Torsten Hoefler, Mikael Johansson, Nikola Konstantinov, Sarit Khirirat, and Cedric Renggli. 2018. The Convergence of Sparsified Gradient Methods. In *Advances in Neural Information Processing Systems*.

[6] Tal Ben-Nun and Torsten Hoefler. 2019. Demystifying Parallel and Distributed Deep Learning: An In-Depth Concurrency Analysis. *ACM Comput. Surv.* (2019).

[7] Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. 2018. signSGD: Compressed Optimisation for Non-Convex Problems. In *Proceedings of the 35th International Conference on Machine Learning*.

[8] Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. 2018. signSGD: Compressed Optimisation for Non-Convex Problems. *Proceedings of the 35th International Conference on Machine Learning* 80 (2018), 560–569.

[9] Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. 2018. signSGD: Compressed Optimisation for Non-Convex Problems. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*. PMLR, 560–569.

[10] Nicholas Carlini, Samuel Deng, Sanjam Garg, Somesh Jha, Saeed Mahloujifar, Mohammad Mahmoody, Shuang Song, Abhradeep Thakurta, and Florian Tramer. 2020. An Attack on InstaHide: Is Private Learning Possible with Instance Encoding? *arXiv preprint arXiv:2011.05315* (2020).

[11] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. 2019. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th {USENIX} Security Symposium ({USENIX} Security 19)*. 267–284.

[12] Chia-Yu Chen, Jiamin Ni, Songtao Lu, Xiaodong Cui, Pin-Yu Chen, Xiao Sun, Naigang Wang, Swagath Venkataramani, Vijayalakshmi (viji) Srinivasan, Wei Zhang, and Kailash Gopalakrishnan. 2020. ScaleCom: Scalable Sparsified Gradient Compression for Communication-Efficient Distributed Training. *Adv. Neural Inf. Process. Syst.* 33 (2020), 13551–13563.

[13] Dingfan Chen, Tribhuvanesh Orekondy, and Mario Fritz. 2020. GS-WGAN: A Gradient-Sanitized Approach for Learning Differentially Private Generators. *Neural Information Processing Systems (NeurIPS)* (2020).

[14] Dingfan Chen, Ning Yu, Yang Zhang, and Mario Fritz. 2020. GAN-Leaks: A Taxonomy of Membership Inference Attacks against Generative Models. In *CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020*.

[15] Xiangyi Chen, Steven Z. Wu, and Mingyi Hong. 2020. Understanding Gradient Clipping in Private SGD: A Geometric Perspective. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 13773–13782.

[16] Cynthia Dwork. 2008. Differential privacy: A survey of results. In *International conference on theory and applications of models of computation*. Springer, 1–19.

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

[18] Fangcheng Fu, Yuzheng Hu, Yihan He, Jiawei Jiang, Yingxia Shao, Ce Zhang, and Bin Cui. 2020. Don't Waste Your Bits! Squeeze Activations and Gradients for Deep Neural Networks via TinyScript. In *International Conference on Machine Learning*.

[19] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.

[20] Moritz Hardt, Katrina Ligett, and Frank McSherry. 2012. A simple and practical algorithm for differentially private data release. In *Advances in Neural Information Processing Systems*. 2339–2347.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

[22] Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nezihe Merve Gürel, Bo Li, Ce Zhang, Costas J. Spanos, and Dawn Xiaodong Song. 2019. Efficient Task-Specific Data Valuation for Nearest Neighbor Algorithms. *Proc. VLDB Endow.* 12 (2019), 1610–1623.

[23] Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nick Hynes, Nezihe Merve Gürel, Bo Li, Ce Zhang, Dawn Xiaodong Song, and Costas J. Spanos. 2019. Towards Efficient Data Valuation Based on the Shapley Value. In *AISTATS*.

[24] Richeng Jin, Yufan Huang, Xiaofan He, Huaiyu Dai, and Tianfu Wu. 2020. Stochastic-Sign SGD for Federated Learning with Theoretical Guarantees. *arXiv preprint arXiv:2002.10940* (2020).

[25] Peter Kairouz, Ziyu Liu, and Thomas Steinke. 2021. The Distributed Discrete Gaussian Mechanism for Federated Learning with Secure Aggregation. *arXiv preprint arXiv:2102.06387* (2021).

[26] J. Kiefer and J. Wolfowitz. 1952. Stochastic Estimation of the Maximum of a Regression Function. *Annals of Mathematical Statistics* 23 (1952), 462–466.

[27] Anastasia Koloskova, Sebastian Stich, and Martin Jaggi. 2019. Decentralized Stochastic Optimization and Gossip Algorithms with Compressed Communication. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, Long Beach, California, USA, 3478–3487.

[28] Anastasia Koloskova, Sebastian Stich, and Martin Jaggi. 2019. Decentralized Stochastic Optimization and Gossip Algorithms with Compressed Communication. In *Proceedings of the 36th International Conference on Machine Learning*.

[29] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. [n. d.]. CIFAR-10 (Canadian Institute for Advanced Research). ([n. d.]). http://www.cs.toronto.edu/~kriz/cifar.html

[30] Yann LeCun. 1998. The MNIST database of handwritten digits. *http://yann.lecun.com/exdb/mnist/* (1998).

[31] Chunyuan Li, Hao Liu, Changyou Chen, Yuchen Pu, Liqun Chen, Ricardo Henao, and Lawrence Carin. 2017. ALICE: Towards Understanding Adversarial Learning for Joint Distribution Matching. *Advances in Neural Information Processing Systems* 30 (2017), 5495–5503.

[32] Xiangru Lian, Yijun Huang, Yuncheng Li, and Ji Liu. 2015. Asynchronous Parallel Stochastic Gradient for Nonconvex Optimization. In *Advances in Neural Information Processing Systems*.

[33] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. 2017. Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent. In *Advances in Neural Information Processing Systems*.

[34] Hyeontaek Lim, David G Andersen, and Michael Kaminsky. 2019. 3LC: LIGHTWEIGHT AND EFFECTIVE TRAFFIC COMPRESSION FOR DISTRIBUTED MACHINE LEARNING. In *Proceedings of the 2nd SysML Conference*.

[35] Ji Liu and Ce Zhang. 2020. Distributed Learning Systems with First-Order Methods. *Foundations and Trends® in Databases* 9, 1 (2020), 1–100.

[36] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2015. Deep Learning Face Attributes in the Wild. In *Proceedings of International Conference on Computer Vision (ICCV)*.

[37] Yunhui Long, Suxin Lin, Zhuolin Yang, Carl A Gunter, and Bo Li. 2019. Scalable differentially private generative student model via pate. *arXiv preprint arXiv:1906.09338* (2019).

[38] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*. PMLR, 1273–1282.

[39] H. Brendan McMahan, Galen Andrew, Ulfar Erlingsson, Steve Chien, Ilya Mironov, Nicolas Papernot, and Peter Kairouz. 2019. A General Approach to Adding Differential Privacy to Iterative Training Procedures. arXiv:1812.06210 [cs.LG]

[40] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. 2018. Which Training Methods for GANs do actually Converge?. In *Proceedings of the 35th International Conference on Machine Learning*.

[41] Ilya Mironov. 2017. Renyi differential privacy. In *Computer Security Foundations Symposium (CSF), 2017 IEEE 30th*. IEEE, 263–275.

[42] Ilya Mironov, Kunal Talwar, and Li Zhang. 2019. R'enyi Differential Privacy of the Sampled Gaussian Mechanism. *arXiv preprint arXiv:1908.10530* (2019).

[43] Nicolas Papernot, Martín Abadi, Ulfar Erlingsson, Ian Goodfellow, and Kunal Talwar. 2017. Semi-supervised knowledge transfer for deep learning from private training data. In *International Conference on Learning Representations*.

[44] Nicolas Papernot, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Ulfar Erlingsson. 2018. Scalable Private Learning with PATE. In *International Conference on Learning Representations*.

[45] Venkatadheeraj Pichapati, Ananda Theertha Suresh, Felix X. Yu, Sashank J. Reddi, and Sanjiv Kumar. 2019. AdaCliP: Adaptive Clipping for Private SGD. *CoRR* abs/1908.07643 (2019). arXiv:1908.07643 http://arxiv.org/abs/1908.07643

[46] Wahbeh Qardaji, Weining Yang, and Ninghui Li. 2014. Priview: practical differentially private release of marginal contingency tables. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. ACM, 1435–1446.

[47] Alec Radford, Luke Metz, and Soumith Chintala. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015).

[48] Daniel Rothchild, Ashwinee Panda, Enayat Ullah, Nikita Ivkin, Ion Stoica, Vladimir Braverman, Joseph Gonzalez, and Raman Arora. 2020. Fetchsgd: Communication-efficient federated learning with sketching. In *International Conference on Machine Learning*. PMLR, 8253–8265.

[49] Christopher De Sa, Ce Zhang, Kunle Olukotun, and Christopher Ré. 2015. Taming the Wild: A Unified Analysis of HOG WILD! -Style Algorithms. In *Proceedings of*

the 28th International Conference on Neural Information Processing Systems.
[50] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved techniques for training GANs. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*. 2234–2242.
[51] R. Shokri, M. Stronati, C. Song, and V. Shmatikov. 2017. Membership Inference Attacks Against Machine Learning Models. In *2017 IEEE Symposium on Security and Privacy (SP)*. 3–18.
[52] Hanlin Tang, Shaoduo Gan, Ce Zhang, Tong Zhang, and Ji Liu. 2018. Communication Compression for Decentralized Training. In *Advances in Neural Information Processing Systems 31*, S Bengio, H Wallach, H Larochelle, K Grauman, N Cesa-Bianchi, and R Garnett (Eds.). Curran Associates, Inc., 7652–7662.
[53] Om Thakkar, Galen Andrew, and H. Brendan McMahan. 2019. Differentially Private Learning with Adaptive Clipping. *CoRR* abs/1905.03871 (2019). arXiv:1905.03871 http://arxiv.org/abs/1905.03871
[54] Reihaneh Torkzadehmahani, Peter Kairouz, and Benedict Paten. 2019. Dp-cgan: Differentially private synthetic data and label generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*.
[55] Thijs Vogels, Sai Praneeth Karimireddy, and Martin Jaggi. 2020. Practical Low-Rank Communication Compression in Decentralized Deep Learning. *Adv. Neural Inf. Process. Syst.* 33 (2020).
[56] L Wang, R Jia, and D Song. 2020. D2P-Fed: Differentially private federated learning with efficient communication. *arxiv. org/pdf/2006.13039* (2020).
[57] Jianqiao Wangni, Jialei Wang, Ji Liu, and Tong Zhang. 2018. Gradient Sparsification for Communication-Efficient Distributed Optimization. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*.
[58] Han Xiao, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747* (2017).
[59] Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. 2018. Differentially Private Generative Adversarial Network. *arXiv preprint arXiv:1802.06739* (2018).
[60] Yunan Ye, Hengzhi Pei, Boxin Wang, Pin-Yu Chen, Yada Zhu, Ju Xiao, and Bo Li. 2020. Reinforcement-learning based portfolio management with augmented asset movement prediction states. In *Proceedings of the AAAI Conference on Artificial Intelligence*.
[61] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. 2018. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*. IEEE, 268–282.
[62] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. 2019. PATE-GAN: Generating Synthetic Data with Differential Privacy Guarantees. In *International Conference on Learning Representations*.
[63] Jun Zhang, Graham Cormode, Cecilia M Procopiuc, Divesh Srivastava, and Xiaokui Xiao. 2017. Privbayes: Private data release via bayesian networks. *ACM Transactions on Database Systems (TODS)* 42, 4 (2017), 25.
[64] Yuheng Zhang, Ruoxi Jia, Hengzhi Pei, Wenxiao Wang, Bo Li, and Dawn Song. 2020. The secret revealer: Generative model-inversion attacks against deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.
[65] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. 2017. Places: A 10 million Image Database for Scene Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2017).

A TOPAGG FOR DP SGD TRAINING

A.1 DP SGD Training Algorithm with TopAgg

We show the details of the DP SGD training using the TopAgg approach in Algorithm 4. We mainly adopt the DP SGD framework as in Abadi et al. [2]. In particular, at each step of the SGD, we compute the gradient for a random subset of examples, and then use the clipping norm to clip each gradient by their ℓ_2 norm. After performing a top-k compression of the gradients to select the sub-dimensions, we then take a sum of the compressed gradients, to which we inject the Gaussian noise subsequently. Finally, we update the model with the compressed DP gradient. Theoretically, since the ℓ_2 norm of the gradient vector is reduced after the top-k compression step, the amount of noise required to achieve the same level of DP guarantee becomes smaller, which implies a potentially better utility of the training algorithm.

Note that directly applying the `TopkStoSignGrad` algorithm (Algorithm 2) to SGD training does not yield a good utility due to information loss during gradient quantization. To overcome this problem, we specially adapt TopAgg to select and preserve a subset of the dimensions in the gradients based on the requirement imposed on the ℓ_2 norm. This new strategy is described as the function `NormTopK` in Algorithm 4. Concretely, given a gradient vector $\mathbf{g}$, we compute $\bar{\mathbf{g}}$ by selecting several dimensions in $\mathbf{g}$ with the highest absolute values to ensure that their squared sum is close to the target norm $k\|\mathbf{g}\|^2$ (here $0 < k < 1$) after the top-k compression, and preserving only the values in the selected dimensions. Thus, the compressed gradient $\bar{\mathbf{g}}$ satisfies the condition $\|\bar{\mathbf{g}}\|^2 \leq k\|\mathbf{g}\|^2$. The remaining dimensions in $\bar{\mathbf{g}}$ are set to 0 since they contain less information. In this way, we achieve the goal of gradient compression without suffering a significant distortion. We note that the ℓ_2 sensitivity of the gradient sum $\sum_i \hat{\mathbf{g}}_t(\mathbf{x}_i)$ is $\sqrt{k}C$ when adapting TopAgg for SGD training, since adding or removing one instance $\mathbf{x}$ in the training set would lead to the gradient sum differing by $\hat{\mathbf{g}}_t(\mathbf{x})$, whose ℓ_2 norm is bounded by $\sqrt{k}C$ due to the operations in `NormTopK`. Thus, the variance of the added Gaussian noise is $kC^2\sigma^2\mathbf{I}$.

Algorithm 4 - Differentially Private SGD training via Gradient Compression and Aggregation TopAgg

Input: Examples $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, Top-$k$-Portion parameter k, loss function $\mathcal{L}(\theta) = \frac{1}{n}\sum_i \mathcal{L}(\theta, \mathbf{x}_i)$. Parameters: batch size B, learning rate γ_t, noise scale σ, gradient clipping norm C, total number of epochs T.
function `NormTopK`($\mathbf{g}, k$)
  $norm \leftarrow \|\mathbf{g}\|^2$
  $target \leftarrow norm \cdot k$ ▷ *target norm after processing*
  $indices \leftarrow$ pick the dimensions in a decreasing order in terms of the squared norm at that dimension, so that the sum of the squared norms in those dimensions add up to right below $target$.
  $\tilde{\mathbf{g}} \leftarrow$ preserve the values of $\mathbf{g}$ at $indices$ and set value at other dimensions as 0
  return $\tilde{\mathbf{g}}$
end function

Initialize θ_0 randomly
for epoch $t \in [T]$ **do**
  Sample a batch of instances $\{\mathbf{x}_{t_i}\}_{i=1}^{B}$ each with sampling probability B/n.
  for each sample $\mathbf{x} \in \{\mathbf{x}_{t_i}\}_{i=1}^{B}$ **do**
    $\mathbf{g}_t(\mathbf{x}) \leftarrow \nabla_{\theta_t}\mathcal{L}(\theta_t, \mathbf{x})$ ▷ *compute gradient*
    $\tilde{\mathbf{g}}_t(\mathbf{x}) \leftarrow \mathbf{g}_t(x)/\max\left(1, \frac{\|\mathbf{g}_t(\mathbf{x})\|}{C}\right)$ ▷ *clip gradient*
    $\hat{\mathbf{g}}_t(\mathbf{x}) \leftarrow$ `NormTopK`$(\tilde{\mathbf{g}}_t(\mathbf{x}), k)$ ▷ *compress gradient*
  end for
  $\bar{\mathbf{g}}_t \leftarrow \frac{1}{B}\left(\sum_i \hat{\mathbf{g}}_t(\mathbf{x}_i) + \mathcal{N}(0, k\sigma^2C^2\mathbf{I})\right)$ ▷ *add noise*
  $\theta_{t+1} \leftarrow \theta_t - \gamma_t\bar{\mathbf{g}}_t$ ▷ *gradient descent*
end for
Output θ_t and compute the overall privacy cost (ε, δ) using Moments Accountant

A.2 Evaluation of TopAgg for DP SGD

In this section, we demonstrate the universality of the proposed DP gradient compression and aggregation algorithm TopAgg in

DataLens, and in particular, the feasibility of applying it to DP SGD training by evaluating its performance on two standard image classification tasks for evaluating DP SGD mechanisms. We first describe the experimental setup. Then we provide extensive evaluation results on a wide range of privacy budgets. Overall, the results show that the TopAgg enabled DP SGD training achieves similar or even better performance on model utility compared with the state-of-the-art Gaussian DP mechanism based on the Moment Account [39] (we will call it GM-DP in the rest of the paper). We also show that TopAgg would bring additional advantages under limited privacy budgets, where the utility gap between the DP model and the vanilla model is large for traditional DP approaches.

A.2.1 Experimental Setup.

We evalute TopAgg for DP SGD training on two datasets, and compare its performance with two baseline frameworks, including the differentially private deep learning (DPDL) [2] and GM-DP [39].

Datasets. We experiment with two datasets commonly used in DP SGD research: MNIST [30] and CIFAR-10 [29]. Both datasets are standard image classification datasets. The description of MNIST is provided in Section 5.1. Similarly, we use 60,000 instances for training and 10,000 for testing. CIFAR-10 consists of 60,000 32×32 colored images of 10 classes. Among all, 50,000 instances are used in training and 10,000 are used in testing.

Models. For MNIST, we adopt a simple convolutional neural network following the default model architecture provided in the example in the open source Opacus library[5]. The network is consisted of two convolutional layers each followed by a max pooling layer, as well as two fully connected layers on the top.

For CIFAR-10, we follow the setting of DPDL, where we first pretrain the classifiers on public datasets, then freeze the parameters of feature extractor and finetune on the fully connected layers. In our paper, we use ResNet-18 [21] as the architecture of the classifier and load the model parameters pretrained on ImageNet[6]. We replace the fully connected layer with a randomly initialized linear head that takes features of 512-dimension extracted from ResNet feature extractor as input and outputs 10-dimensional prediction logits. During training, we freeze the parameters of ResNet feature extractor including the parameters of Batch Normalization layers to ensure that the feature extractor will not leak any privacy-sensitive data information.

We compare the performance of TopAgg with two state of the art DP SGD mechanisms: GM-DP [39] and DPDL [2]. In particular, we build upon Opacus [1], a PyTorch implementation of GM-DP that implements the DP SGD training scheme and privacy accountant method in [39], which enables convenient control of randomness in the framework. Our implementation of TopAgg for DP SGD training is also built upon the Opacus library. For DPDL which is the first work that proposed and evaluated the DP SGD training scheme, we directly compare with the results reported in Section 5.2 and Section 5.3 of the paper for fairness, which present the best model utility performance.

Evaluation Metrics. We adopt *model utility*, which is calculated as the *classification accuracy* of the trained models, as the evaluation metric for assessing the effectiveness of our algorithm

[5]Code at https://github.com/pytorch/opacus/blob/master/examples/mnist.py
[6]Publicly available at https://pytorch.org/docs/stable/torchvision/models.html

Table 8: Model utility when adapting TopAgg to DP SGD training on (a) MNIST and (b) CIFAR-10 with different privacy parameter ε and TopAgg parameter k. In all cases, $\delta = 10^{-5}$.

(a) MNIST

	TopAgg			GM-DP
	$k = 0.6$	$k = 0.7$	$k = 0.8$	
$\varepsilon = 0.05$	73.87 ± 4.77	75.81 ± 3.59	77.15 ± 2.89	78.40 ± 4.00
$\varepsilon = 0.10$	85.67 ± 1.48	**86.12** ± 1.39	85.88 ± 1.97	84.55 ± 0.98
$\varepsilon = 0.20$	89.84 ± 0.64	90.25 ± 0.40	90.80 ± 1.04	91.17 ± 0.37
$\varepsilon = 0.30$	91.21 ± 0.53	91.85 ± 0.21	92.35 ± 0.53	**93.31** ± 0.42
$\varepsilon = 0.50$	92.83 ± 0.52	93.50 ± 0.75	93.82 ± 0.46	**94.38** ± 0.24
$\varepsilon = 0.70$	94.41 ± 0.19	94.61 ± 0.24	94.65 ± 0.27	**95.08** ± 0.10
$\varepsilon = 1.00$	95.13 ± 0.33	95.22 ± 0.20	**95.42** ± 0.20	95.41 ± 0.26
$\varepsilon = \infty$	97.94 ± 0.19	98.58 ± 0.11	98.79 ± 0.09	**99.08** ± 0.04

(b) CIFAR-10

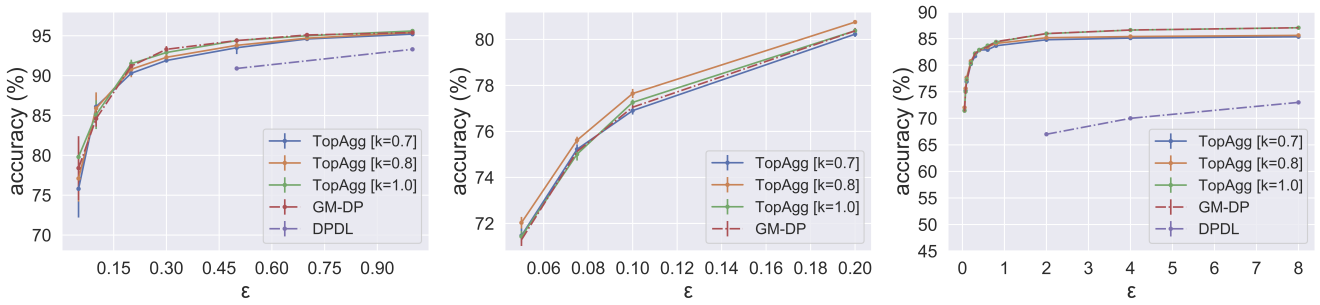
	TopAgg			GM-DP
	$k = 0.6$	$k = 0.7$	$k = 0.8$	
$\varepsilon = 0.025$	42.18 ± 0.89	44.32 ± 0.59	**45.87** ± 0.97	41.58 ± 2.01
$\varepsilon = 0.050$	65.10 ± 0.48	71.47 ± 0.32	**72.03** ± 0.25	71.29 ± 0.27
$\varepsilon = 0.075$	74.70 ± 0.26	75.22 ± 0.21	**75.62** ± 0.15	75.13 ± 0.19
$\varepsilon = 0.10$	75.56 ± 0.18	76.91 ± 0.17	**77.65** ± 0.20	77.05 ± 0.09
$\varepsilon = 0.20$	79.44 ± 0.07	80.23 ± 0.12	**80.76** ± 0.08	80.37 ± 0.13
$\varepsilon = 0.30$	80.19 ± 0.17	81.70 ± 0.08	82.15 ± 0.15	**82.19** ± 0.10
$\varepsilon = 0.40$	81.91 ± 0.10	82.86 ± 0.05	**82.82** ± 0.13	82.80 ± 0.09
$\varepsilon = 0.60$	82.56 ± 0.36	82.91 ± 0.20	83.44 ± 0.15	**83.46** ± 0.04
$\varepsilon = 0.80$	83.40 ± 0.06	83.67 ± 0.22	84.10 ± 0.07	**84.44** ± 0.07
$\varepsilon = 2.00$	84.34 ± 0.08	84.80 ± 0.18	85.18 ± 0.08	**85.97** ± 0.04
$\varepsilon = 4.00$	84.82 ± 0.04	85.11 ± 0.08	85.40 ± 0.08	**86.63** ± 0.05
$\varepsilon = 8.00$	85.13 ± 0.05	85.37 ± 0.04	85.62 ± 0.04	**87.05** ± 0.03
$\varepsilon = \infty$	85.19 ± 0.3	85.42 ± 0.02	85.80 ± 0.03	**87.85** ± 0.02

TopAgg. For each dataset, each privacy budget ε, and each `NormTopK` parameter k, we perform an extensive grid search for the combination of hyper-parameters (including gradient clipping norm C, noise scale σ, batch size B, and learning rate lr) for all methods for fair comparison. We then use the best hyper-parameters to start 10 runs with different random seeds for noise generation and report the averaged results for each method. For models trained under baseline frameworks, we follow the same parameter search protocol and parameter grid to obtain the reported results.

A.2.2 Experimental Results.

We compare TopAgg (with $k \in \{0.7, 0.8, 1.0\}$) with two baselines (GM-DP and DPDL) on DP SGD training under a wide variety of privacy constraints, as shown in Figure 3. For clarity of presentation, we leave the complete set of results in Table 8. For MNIST, we mainly evaluate small privacy budgets ($\varepsilon \leq 1.0$), since the performance gap of private and non-private models on MNIST is negligible for large ε. For CIFAR-10, specifically, we examine the regions of both small ε and large ε respectively in a more comprehensive manner.

We first note that GM-DP is a special case of our TopAgg when $k = 1.0$. In this case, the complete gradient is preserved after the Top-K step, and therefore the performance of TopAgg is equivalent to GM-DP. Still, we provide both results as a sanity check in Figure 3.



(a) MNIST (b) CIFAR-10 (under small ε) (c) CIFAR-10 (under full range of ε)

Figure 3: The performance of DataLens applied to SGD training on two datasets: (a) MNIST and (b-c) CIFAR-10. We run DataLens with a range of $k \in \{0.7, 0.8, 1.0\}$ and compare its performance with two baselines GM-DP and DPDL on a wide range of privacy budget ε.

Table 9: Model utility for applying norm-based TopAgg and dimension-based TopAgg in DP SGD training on MNIST dataset. ε is the privacy budget and k is the top-k parameter in TopAgg. In all cases, $\delta = 10^{-5}$.

	TopAgg (norm-based)			TopAgg (dim-based)			GM-DP
	$k = 0.6$	$k = 0.7$	$k = 0.8$	$k = 0.2$	$k = 0.4$	$k = 0.6$	
$\varepsilon = 0.05$	73.87 ± 4.77	75.81 ± 3.59	77.15 ± 2.89	74.25 ± 3.25	**78.45** ± 2.89	73.55 ± 1.97	78.40 ± 4.00
$\varepsilon = 0.10$	85.67 ± 1.48	86.12 ± 1.39	85.88 ± 1.97	83.65 ± 0.73	85.59 ± 1.39	**86.41** ± 0.70	84.55 ± 0.98
$\varepsilon = 0.20$	89.84 ± 0.64	90.25 ± 0.40	90.80 ± 1.04	89.10 ± 1.22	90.48 ± 0.73	89.30 ± 0.74	**91.17** ± 0.37

We observe that the results of TopAgg ($k = 1.0$) and GM-DP are indeed close in almost all scenarios regardless of the randomness of the algorithm (the green and red lines are generally overlapped).

In Figure 3 (a) for MNIST, different curves are intertwined, indicating that choosing different k values can only influence the model performance by a little margin. Thus, it does not hurt to adopt TopAgg with different k. The main reason is that, on MNIST, the model utility gap before and after adding DP noise is small, which does not provide space for smaller gradient compression ratios to improve the performance.

Moreover, for CIFAR-10 whose dimensionality is around 4 times larger than MNIST, we observe consistent performance improvements under the limited privacy budget ε in Figure 3 (b), which is also aligned with our observation in TopAgg for generative models, where our methods demonstrate a large margin over baselines under limited privacy budgets. Specifically, In Table 8 (b) under $\varepsilon = 0.025$, we observe that TopAgg with $k = 0.8$ can achieve the model accuracy of 45.87%, which is more than 4% higher than the baseline. This again verifies our theoretical analysis that TopAgg can help save the privacy budget consumption by compressing the gradient, and therefore substantially help the model convergence and improve the utility of the model.

With larger privacy budgets, Figure 3 (c) shows that TopAgg with very small k tends to have worse performance. It indicates that given small k, the bias introduced by top-k compression outweighs the bias introduced by low differential privacy noise. Without DP noise ($\varepsilon = \infty$), we observe that TopAgg with $k < 1$ has slightly worse performance than the GM-DP baseline, which is because without DP noise there is no longer the benefits of lower DP noise brought by top-k, while the bias introduced by gradient compression starts to hurt the performance moderately.

We further point out that DPDL has the worst performance among all as shown in Figure 3. This phenomenon is well understood given that the privacy analysis is looser in Abadi et al. [2] compared with the privacy analysis based on Rényi Differential Privacy [42] adopted in both our TopAgg and GM-DP here.

In addition, we empirically examine the hyper-parameters and the impact of gradient compression and noise injection in TopAgg on MNIST. We omit the detailed results in Appendix A. We observe that the bias induced by the DP noise is indeed larger than the gradient compression, which confirms our theoretical analysis of the convergence for TopAgg.

A.3 Ablation Studies on Hyper-parameters

DP-SGD algorithms (GM-DP and DPDL) contains several key parameters: the noise multiplier σ of the injected Gaussian noise, gradient clipping constant c, batch size that will affect the sampling rate q, and learning rate. TopAgg adds another important parameter k for `NormTopK` on top of the GM-DP framework. To search for the optimal hyper-parameters, we conduct comprehensive grid search. We list the optimal hyper-parameters under several different privacy budgets ε in Table 11 for MNIST and CIFAR-10.

A.4 Tradeoff between Gradient Compression and Noise Injection

In essence, TopAgg differs from the standard DP SGD training scheme and moment accountant method adopted in GM-DP [39] mainly in the introduction of the gradient compression parameter

Table 10: Results of the control experiments to explore the impact of gradient compression and noise injection.

(a) Experimental setup. Each cell is one experimental scenario.

	noise $\sim \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})$	noise $\sim \mathcal{N}(0, k\sigma^2 C^2 \mathbf{I})$	no noise
NormTopK	TopK-GM-DP	TopAgg	TopK-SGD
no compression	GM-DP [39]	–	clipped SGD

(b) Experimental results on MNIST dataset under small and big privacy budgets. (above) $\varepsilon = 0.1$ and $k = 0.8$; (below) $\varepsilon = 1.0$ and $k = 0.8$.

	noise $\sim \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})$	noise $\sim \mathcal{N}(0, k\sigma^2 C^2 \mathbf{I})$	no noise
NormTopK	83.86 ± 1.49	85.88 ± 1.97	91.96 ± 0.61
no compression	85.05 ± 1.85	–	94.26 ± 0.43
	noise $\sim \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})$	noise $\sim \mathcal{N}(0, k\sigma^2 C^2 \mathbf{I})$	no noise
NormTopK	94.95 ± 0.25	95.42 ± 0.20	98.03 ± 0.09
no compression	95.56 ± 0.17	–	98.94 ± 0.04

k. On the one hand, using a smaller k to compress the gradients leads to more biased results in the returned gradients, which can cause performance degradation. On the other hand, the gradient norm becomes smaller after TopAgg based training which enables the introduction of less noise to achieve the same level of privacy guarantee, and therefore less distortion to the prediction. Noticing the tradeoff, we ask, *is there a sweet spot where the performance increase caused by the injection of less noise surpasses the model utility degradation induced by the bias in gradient compression?*

To this end, we design a set of control experiments to analyze the impact of the two factors: 1) compression parameter of gradients and 2) amount of injected DP noise. We provide the setup of the control experiments in Table 10a. Concretely, we investigate 3 levels of noise injection corresponding to the requirement of 3 algorithms (non-private SGD [26], GM-DP [39], and our TopAgg), and 2 scenarios of gradient compression (no compression and our NormTopK compression). Note that only GM-DP and TopAgg satisfy the intended privacy requirements. We do not report the results for the combination of no compression and reduced noise, since it is blatantly non-private and does not offer additional insights. Rather, we investigate the combination of NormTopK and full noise which we title TopK-GM-DP, in a hope to build the bridge between GM-DP and TopAgg. We additionally examine TopK-SGD (the combination of NormTopK and no noise) to get the sense of an upper-bound of the performance after gradient compression. For the purpose of controlling variables, we control the clipping norm C and noise scale σ to be the same for all the scenarios, and train the non-private algorithms using the same number of iterations as the private ones, which is determined by the corresponding privacy budget.

The results of the control experiments on different privacy budgets are provided in Table 10 (b). We summarize our observations as follows. First, along each row or each column of Table 10 (b), the performance of the training schemes will experience an increase. Naturally, the non-private SGD training will give the best performance of all (despite norm clipping). This means that less noise and no compression will generally yield better results. Second, noise injection has a larger impact on the performance than gradient compression in both cases of small and big ε. Third, the reduction in the scale of injected noise will compensate the performance decrease caused by gradient compression, therefore resulting in a similar or slightly better performance of TopAgg when compared with GM-DP. Given these observations, we conclude that the impact of gradient compression is negligible compared with noise injection, and that it is beneficial to exploit gradient compression to trade for the reduction in injected noise to achieve a potentially better performance.

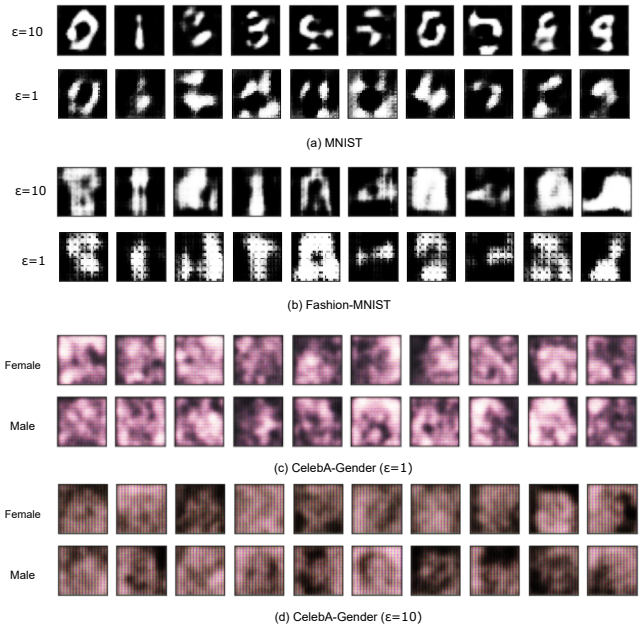


Figure 4: Visualization of generated images from DataLens

B VISUALIZATION OF IMAGE QUALITY

We visualize the private synthetic images for MNIST, Fashion-MNIST and CelebA, as shown in Figure 4. As expected the image has a lot of noise, since our goal is to generate data which can protect privacy and ensure high data utility in terms of training high performance models. It is interesting to see that these generated images are enough to train useful models, which lead to interesting future direction on what ML models actually learn from data.

C EXPERIMENTAL DETAILS

C.1 Visual Quality Evaluation

We evaluate both Inception Score and Frechet Inception Distance for DataLens and baselines over four datasets. We present the evaluation results in Table 12.

For Inception Score, in our experiments, we follow GS-WGAN and use the implementation[7] for Inception Score calculation with pretrained classifiers trained on real datasets (with test accuracy equal to 99%, 93%, 97% on MNIST, Fashion-MNIST, and CelebA-Gender).

For FID, we observe it is not necessarily consistent with Inception Score (e.g., for MNIST $\varepsilon = 1$, DataLens has better IS than G-PATE,

[7] https://github.com/ChunyuanLI/MNIST_Inception_Score

Table 11: Optimal hyper-parameters for TopAgg and GM-DP baseline on MNIST and CIFAR-10 with different privacy parameter ε and TopAgg parameter k. In all cases, $\delta = 10^{-5}$.

(a) MNIST ($\varepsilon = 0.05$)

	TopAgg			GM-DP
	$k = 0.6$	$k = 0.7$	$k = 0.8$	
c	10.0	5.0	5.0	3.0
σ	5.8	6.6	7.4	8.2
batch size	128	128	128	128
learning rate	0.01	0.01	0.01	0.01

(b) MNIST ($\varepsilon = 0.2$)

	TopAgg			GM-DP
	$k = 0.6$	$k = 0.7$	$k = 0.8$	
c	12.0	10.0	10.0	10.0
σ	3.6	3.6	3.6	3.4
batch size	512	512	512	512
learning rate	0.1	0.1	0.1	0.08

(c) MNIST ($\varepsilon = 1.0$)

	TopAgg			GM-DP
	$k = 0.6$	$k = 0.7$	$k = 0.8$	
c	16.0	12.0	10.0	14.0
σ	1.6	1.6	1.6	1.8
batch size	512	512	512	512
learning rate	0.08	0.1	0.1	0.04

(d) CIFAR-10 ($\varepsilon = 0.025$)

	TopAgg			GM-DP
	$k = 0.6$	$k = 0.7$	$k = 0.8$	
c	1	1.5	0.8	0.3
σ	6	7	6	7
batch size	24	32	24	32
learning rate	0.0008	0.001	0.001	0.002

(e) CIFAR-10 ($\varepsilon = 0.4$)

	TopAgg			GM-DP
	$k = 0.6$	$k = 0.7$	$k = 0.8$	
c	1	1	1	1.5
σ	2.5	2.5	2.5	3
batch size	96	96	96	96
learning rate	0.005	0.005	0.005	0.005

(f) CIFAR-10 ($\varepsilon = 8$)

	TopAgg			GM-DP
	$k = 0.6$	$k = 0.7$	$k = 0.8$	
c	0.5	0.5	0.5	2.5
σ	2.5	2.5	3	2.5
batch size	2048	2048	2048	2048
learning rate	0.2	0.2	0.2	0.04

Algorithm 5 - Gradient Compression via k-level Stochastic Gradient (`StoKlevelGrad`). This algorithm takes in a gradient vector of a teacher model $\mathbf{g}^{(i)}$ and returns the compressed gradient vector $\tilde{\mathbf{g}}^{(i)}$.

Input: Gradient vector $\mathbf{g}^{(i)}$, gradient clipping constant c, top-k
$\mathbf{g}_j^{(i)} = \min(\max(\mathbf{g}_j^{(i)}, -c), c)$ for each dimension j in $\mathbf{g}^{(i)}$
▷ *Clip each dimension of $\mathbf{g}^{(i)}$ so that $-c \le \mathbf{g}_j^{(i)} \le c$.*
$\mathbf{g}^{(i)} = R \times \mathbf{g}^{(i)}$ ▷ *Random Rotation*
$\hat{\mathbf{g}}^{(i)} \leftarrow \mathbf{g}^{(i)} / \left\|\mathbf{g}^{(i)}\right\|_\infty$ ▷ *gradient normalization to (-1, 1)*
$\tilde{\mathbf{g}}^{(i)} \leftarrow \mathbf{0}$
let $b[r] := -k/2 + 2r$ for every $r \in [0, k)$
let $m[r] := -c + \frac{2rc}{k-1}$ for every $r \in [0, k)$ ▷ *initialization of the compressed sparse gradient vector*
for each index j, and $b[r] \le \mathbf{g}_j^{(i)} \le b[r+1]$ **do**

$$\tilde{g}_j^{(i)} = \begin{cases} b[r+1], & \text{with probability } \frac{\mathbf{g}_j^{(i)} - m[r]}{m[r+1] - m[r]} \\ b[r], & o.w. \end{cases}$$

end for
Return: $\tilde{\mathbf{g}}^{(i)}$

but worse FID than G-PATE), which we think the reason is because FID is evaluated based on models trained with ImageNet which may not be suitable for evaluating datasets such as MNIST. In our experiments, we follow GS-WGAN and use the implementation[8] for FID calculation.

C.2 Adapting Other Gradient Compression Algorithms to DataLens

In this section, we illustrate how we adapt D^{2}P-Fed and FetchSGD to DataLens framework.

For D^{2}P-Fed, we replace our Algorithm 2 (`TopkStoSignGrad`) that uses stochastic sign compression with their method, which essentially uses k-level gradient quantization and random rotation for gradient pre-processing. The detailed algorithm is shown in Algorithm 6 and Algorithm 5.

For FetchSGD, we uses the same stochastic sign compression as we leverage sign signal as teacher voting in PATE framework. During aggregation, we use Count Sketch data structure, and use top-k and unsketch operation to retrieve the aggregated gradient. The detailed algorithm is shown in Algorithm 7.

Algorithm 6 - Differentially Private Gradient Compression and Aggregation (D^{2}P-Fed for DataLens). This algorithm takes gradients of teacher models and returns the compressed and aggregated differentially private gradient vector.

Input: Teacher number N, gradient vectors of teacher models $\mathcal{G} = \{\mathbf{g}^{(1)}, \ldots, \mathbf{g}^{(N)}\}$, gradient clipping constant c, top-k, noise parameters σ, voting threshold β
▷ *Phase I: Gradient Compression*
for each teacher's gradient $\mathbf{g}^{(i)}$ **do**
    $\tilde{\mathbf{g}}^{(i)} \leftarrow$ `StoKlevelGrad`$(\mathbf{g}^{(i)}, c, k)$
end for
▷ *Phase II: Differential Private Gradient Aggregation*
$\tilde{\mathbf{g}}^* \leftarrow \sum_{i=1}^{N} \tilde{\mathbf{g}}^{(i)} + \mathcal{N}(0, \sigma^2)$
▷ *Phase III: Gradient Thresholding (Post-Processing)*
for each dimension $\tilde{g}_j^*$ of $\tilde{\mathbf{g}}^*$ **do**

$$\bar{g}_j = \begin{cases} 1, & \text{if } \tilde{g}_j^* \ge \beta N; \\ -1, & \text{if } \tilde{g}_j^* \le -\beta N; \\ 0, & \text{otherwise.} \end{cases}$$

end for
Return: $\bar{\mathbf{g}}$

Algorithm 7 - Differentially Private Gradient Compression and Aggregation (FetchSGD for DataLens). This algorithm takes gradients of teacher models and returns the compressed and aggregated differentially private gradient vector.

Input: Teacher number N, gradient vectors of teacher models $\mathcal{G} = \{\mathbf{g}^{(1)}, \ldots, \mathbf{g}^{(N)}\}$, gradient clipping constant c, top-k, noise parameters σ, voting threshold β
S = CountSketchAggregator()
▷ *Phase I: Gradient Compression*
for each teacher's gradient $\mathbf{g}^{(i)}$ **do**
    $\tilde{\mathbf{g}}^{(i)} \leftarrow$ `TopkStoSignGrad`$(\mathbf{g}^{(i)}, c, k)$
    S += Sketch$(\tilde{\mathbf{g}}^{(i)})$
end for
▷ *Phase II: Differential Private Gradient Aggregation*
$\tilde{\mathbf{g}}^* \leftarrow$ top-k(unSketch(S)) + $\mathcal{N}(0, \sigma^2)$
Return: $\tilde{\mathbf{g}}^*$

[8] https://github.com/google/compare_gan

Table 12: Quality evaluation of images generated by different differentially private data generative models on Image Datasets: Inception Score (IS) and Frechet Inception Distance (FID) are calculated to measure the visual quality of the generated data under different ε ($\delta = 10^{-5}$).

(a) $\varepsilon = 1$

Dataset	Metrics	Real data	DP-GAN	PATE-GAN	G-PATE	GS-WGAN	DataLens
MNIST	IS ↑	9.86	1.00	1.19	3.60	1.00	**4.37**
	FID ↓	1.04	470.20	231.54	**153.38**	489.75	186.06
Fashion-MNIST	IS ↑	9.01	1.03	1.69	3.41	1.00	**3.93**
	FID ↓	1.54	472.03	253.19	214.78	587.31	**194.98**
CelebA	IS ↑	1.88	1.00	1.15	1.11	1.00	**1.18**
	FID ↓	2.38	485.92	434.47	302.45	437.33	**297.73**

(b) $\varepsilon = 10$

Dataset	Metrics	Real data	DP-GAN	PATE-GAN	G-PATE	GS-WGAN	DataLens
MNIST	IS ↑	9.86	1.00	1.46	5.16	**8.59**	5.78
	FID ↓	1.04	304.86	253.55	150.62	**58.77**	173.50
Fashion-MNIST	IS ↑	9.01	1.05	2.35	4.33	**5.87**	4.58
	FID ↓	1.54	433.38	229.25	171.90	**135.47**	167.68
CelebA	IS ↑	1.88	1.00	1.16	1.12	1.00	**1.42**
	FID ↓	2.38	485.41	424.60	323.95	432.58	**320.84**

D PROOFS

We now prove Theorem 5 and Theorem 6.

Proof of Theorem 5. We have

$$\begin{aligned}
&\Pr[\mathcal{M}(\tilde{\mathcal{G}}, T, \beta) \neq \bar{\mathbf{g}}^*] \\
=&1 - \Pr[\mathcal{M}(\tilde{\mathcal{G}}, T, \beta) = \bar{\mathbf{g}}^*] \\
=&1 - \prod_{\{j|\bar{g}_j^*=1\}} \Pr[f_j + n_j \geq \beta T] \prod_{\{j|\bar{g}_j^*=-1\}} \Pr[f_j + n_j \leq -\beta T] \prod_{\{j|\bar{g}_j^*=0\}} \Pr[-\beta T < f_j + n_j < \beta T] \\
=&1 - \prod_{\{j|\bar{g}_j^*=1\}} \Pr[n_j \geq \beta T - f_j] \prod_{\{j|\bar{g}_j^*=-1\}} \Pr[n_j \leq -\beta T - f_j] \prod_{\{j|\bar{g}_j^*=0\}} \Pr[-\beta T - f_j < n_j < \beta T - f_j] \\
=&1 - \prod_{\{j|\bar{g}_j^*=1\}} \left(1 - \Phi\left(\frac{\beta T - f_j}{\sigma}\right)\right) \prod_{\{j|\bar{g}_j^*=-1\}} \Phi\left(\frac{\beta T - f_j}{\sigma}\right) \prod_{\{j|\bar{g}_j^*=0\}} \operatorname{erf}\left(\frac{\beta T - f_j}{\sqrt{2}\sigma}\right),
\end{aligned}$$

where the last equality holds because n_j follows the normal distribution with mean 0 and variance σ^2, concluding the proof.

Proof of Theorem 6. We begin by fixing $t \in [T]$. The assumption that f has L-Lipschitz gradient, *i.e.*, $\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$, implies, through a well-known argument, that

$$f(x_{t+1}) - f(x_t) \leq \langle \nabla f(x_t), x_{t+1} - x_t \rangle + \frac{L}{2}\|x_{t+1} - x_t\|^2,$$

Recall that $x_{t+1} - x_t = -\frac{\gamma}{N}\sum_{n\in[N]} \left(Q(\texttt{clip}(\texttt{top-k}(F'_n(x_t)), c), \xi_t) + \mathcal{N}(0, Ak)\right)$. Taking the expectation over the quantization and the insertion of data-privacy noise yields

$$\begin{aligned}
\mathbb{E}_{\mathcal{N}}\mathbb{E}_{\xi_t} f(x_{t+1}) - f(x_t) \leq -\frac{\gamma}{N}\sum_{n\in[N]} \underbrace{\langle \nabla f(x_t), \mathbb{E}_{\xi_t}\left[Q(\texttt{clip}(\texttt{top-k}(F'_n(x_t), c), \xi_t)\right]\rangle}_{I_n(x_t)} - \underbrace{\frac{\gamma}{N}\sum_{n\in[N]} \langle \nabla f(x_t), \mathbb{E}_{\mathcal{N}}\left[\mathcal{N}(0, Ak)\right]\rangle}_{=0} \\
+ \frac{L\gamma^2}{N}\sum_{n\in[N]} \underbrace{\mathbb{E}_{\xi_t}\left\|Q(\texttt{clip}(\texttt{top-k}(F'_n(x_t)), c), \xi_t)\right\|^2}_{J_n(x_t)} + \underbrace{\frac{L\gamma^2}{N}\sum_{n\in[N]} \mathbb{E}_{\mathcal{N}}\|\mathcal{N}(0, Ak)\|^2}_{=L\gamma^2 Ak},
\end{aligned}$$

where we used the Cauchy-Schwarz inequality $(a_1 + \ldots + a_n)^2 \leq n(a_1^2 + \ldots + a_n^2)$ for $n = 2N$.

For $I_n(x_t)$ note that

$$-\frac{\gamma}{N}\sum_{n\in[N]} I_n(x_t) = -\frac{\gamma}{N}\langle \nabla f(x_t), \mathbb{E}_{\xi_t}\left[Q(\texttt{clip}(\texttt{top-k}(F_n'(x_t)), c), \xi_t)\right]\rangle$$

$$\left\{\mathbb{E}_{\xi}\left[Q(x,\xi)\right] = x\right\} \qquad = -\frac{\gamma}{N}\sum_{n\in[N]}\langle \nabla f(x_t), \texttt{clip}(\texttt{top-k}(F_n'(x_t)), c)\rangle$$

$$= -\frac{\gamma}{N}\sum_{n\in[N]}\underbrace{\langle \nabla f(x_t), \texttt{clip}(F_n'(x_t), c)\rangle}_{I_n^{(1)}} + \frac{\gamma}{N}\sum_{n\in[N]}\underbrace{\langle \nabla f(x_t), \texttt{clip}(F_n'(x_t)), c) - \texttt{clip}(\texttt{top-k}(F_n'(x_t)), c)\rangle}_{I_n^{(2)}}$$

Claim 1. For $\alpha = \frac{1}{d+2}$ and under the assumptions from Theorem 6 one has

$$-\frac{\gamma}{N}\sum_{n\in[N]} I_n^{(1)} \leq \gamma \max\{-\alpha\|\nabla f(x_t)\|^2 + \|\sigma\|^2 + \|\sigma\|M, \quad -\alpha c\|\nabla f(x_t)\|_1 + 2c\|\sigma\|_1\}. \tag{4}$$

Proof of Claim 1. For the ease of notation, let $x = x_t$ and $g_n(x) = F_n'(x)$. First note that, per coordinate $i \in [d]$,

$$\texttt{clip}(g_n(x)_i, c) = c \cdot \texttt{sign}(g_n(x)_i) \cdot \mathbf{1}\{|g_n(x)_i| \geq c\} + g_n(x)_i \cdot \mathbf{1}\{|g_n(x)_i| < c\}.$$

The main idea is to prove that one of these yields the main term, which would correspond to $-\gamma\|\nabla f(x)\|^2$ for the usual gradient descent, and $-c\gamma\|\nabla f(x)\|_1$ for the signed gradient descent. With that in mind, let us for each $i \in [d]$ define $A_i = \{n \in [N] \colon |g_n(x)_i| \geq c\}$ and $B_i = \{n \in [N] \colon |g_n(x)_i| < c\}$, with $A_i \cap B_i = \emptyset$ and $A_i \cup B_i = [N]$, for all $i \in [d]$. Then

$$-\frac{\gamma}{N}\sum_{n\in[N]} I_n^{(1)} = -\frac{\gamma c}{N}\sum_{i\in[d]}\sum_{n\in A_i}\nabla f(x)_i \cdot \texttt{sign}(g_n(x)_i) - \frac{\gamma}{N}\sum_{i\in[d]}\sum_{n\in B_i}\nabla f(x)_i \cdot g_n(x)_i.$$

To explore the above mentioned dichotomy, we now rewrite the quantity we are trying to estimate in two ways:

$$-\frac{\gamma}{N}\sum_{n\in[N]} I_n^{(1)} = -\gamma\|\nabla f(x)\|^2 + \underbrace{\frac{\gamma}{N}\sum_{i\in[d]}\sum_{n\in A_i}|\nabla f(x)_i|^2}_{\texttt{err}(GD)} + \underbrace{\frac{\gamma}{N}\sum_{i\in[d]}\sum_{n\in A_i}\nabla f(x)_i\,(g_n(x)_i - \nabla f(x)_i - c\cdot\texttt{sign}(g_n(x)_i))}_{\texttt{err}(GD_2)},$$

$$-\frac{\gamma}{N}\sum_{n\in[N]} I_n^{(1)} = -c\gamma\|\nabla f(x)\|_1 + \underbrace{\frac{\gamma}{N}\sum_{i\in[d]}\sum_{n\in B_i}\nabla f(x)_i\,(c\cdot\texttt{sign}(g_n(x)_i) - g_n(x)_i)}_{\texttt{err}(signGD)} + \underbrace{\frac{\gamma c}{N}\sum_{i\in[d]}\sum_{n\in[N]}\nabla f(x)_i\,(\texttt{sign}(\nabla f(x)_i) - \texttt{sign}(g_n(x)_i))}_{\texttt{err}(signGD_2)}.$$

We start by bounding $\texttt{err}(GD_2)$ and $\texttt{err}(singGD_2)$. For $\texttt{err}(GD_2)$ we have

$$\texttt{err}(GD_2) \leq \frac{\gamma}{N}\sum_{i\in[d]}\sum_{n\in A_i}|\nabla f(x)_i||g_n(x)_i - \nabla f(x)_i|$$

$$\leq \frac{\gamma}{N}\sum_{i\in[d]}\sum_{n\in A_i}|g_n(x)_i - \nabla f(x)_i|^2 + \frac{\gamma}{N}\sum_{i\in[d]}\sum_{n\in A_i}|g_n(x)_i||g_n(x)_i - \nabla f(x)_i|$$

{Cauchy-Schwarz inequality} $$\leq \frac{\gamma}{N}\sum_{i\in[d]}\sum_{n\in A_i}|g_n(x)_i - \nabla f(x)_i|^2 + \frac{\gamma}{N}\sqrt{\sum_{i\in[d]}\sum_{n\in A_i}|g_n(x)_i|^2}\sqrt{\sum_{i\in[d]}\sum_{n\in A_i}|g_n(x)_i - \nabla f(x)_i|^2}$$

$$\left\{\sum_{n\in[N]}\|g_n(x)\|^2 \leq M^2N,\ \sum_{n\in[N]}\|g_n(x)_i - \nabla f(x)_i\|^2 \leq \sigma_i^2 N\right\} \qquad \leq \frac{\gamma}{N}\sum_{i\in[d]}\sigma_i^2 N + \frac{\gamma}{N}\sqrt{M^2N}\sqrt{\sum_{i\in[d]}\sigma_i^2 N},$$

Therefore, for $\texttt{err}(GD_2)$ we have

$$\texttt{err}(GD_2) \leq \gamma\|\sigma\|^2 + \gamma\|\sigma\|M. \tag{5}$$

We bound $\texttt{err}(signGD_2)$ in a similar vein as in [9]. Note that

$$\texttt{err}(signGD_2) = \frac{2\gamma c}{N}\sum_{i\in[d]}\sum_{n\in[N]}\nabla f(x)_i \cdot \mathbf{1}\left\{\texttt{sign}(\nabla f(x)_i) \neq \texttt{sign}(g_n(x)_i)\right\}$$

$$= \frac{2\gamma c}{N}\sum_{i\in[d]}|\nabla f(x)_i|\sum_{n\in[N]}\mathbf{1}\left\{|\nabla f(x)_i - g_n(x)_i| \geq |\nabla f(x)_i|\right\}.$$

Let $E_i := \{n \in [N] : |\nabla f(x)_i - g_n(x)_i| \geq |\nabla f(x)_i|\}$. Then $\texttt{err}(signGD_2) = \frac{2\gamma c}{N} \sum_{i\in[d]} |\nabla f(x)_i||E_i|$. Note that

$$\frac{|E_i|}{N} \leq \frac{1}{N} \sum_{n\in[N]} \frac{|\nabla f(x)_i - g_n(x)_i|}{|\nabla f(x)_i|} \leq \frac{1}{|\nabla f(x)_i|} \sqrt{\frac{1}{N} \sum_{n\in[N]} |\nabla f(x)_i - g_n(x)_i|^2} \leq \frac{\sigma_i}{|\nabla f(x)_i|},$$

using the Cauchy-Schwarz inequality. This yields

$$\texttt{err}(signGD_2) \leq 2\gamma c \|\sigma\|_1. \tag{6}$$

We now want to prove that either $\texttt{err}(GD) \leq (1-\alpha)\gamma\|\nabla f(x)\|^2$ or $\texttt{err}(signGD_2) \leq (1-\alpha)c\gamma\|\nabla f(x)\|_1$. For the sake of contradiction, suppose that

$$\texttt{err}(GD) > (1-\alpha)\gamma\|\nabla f(x)\|^2, \qquad \texttt{err}(signGD) > (1-\alpha)c\gamma\|\nabla f(x)\|_1. \tag{7}$$

It is easy to see that these conditions, for $\texttt{err}(signGD)$ imply

$$\frac{c\gamma}{N} \sum_{i\in[d]} |\nabla f(x)_i||B_i| \geq \texttt{err}(signGD) > (1-\alpha)c\gamma\|\nabla f(x)\|_1, \tag{8}$$

whereas for $\texttt{err}(GD)$ they imply

$$\frac{\gamma}{N} \sum_{i\in[d]} |\nabla f(x)_i|^2|A_i| \geq \texttt{err}(GD) > (1-\alpha)\gamma\|\nabla f(x)\|^2. \tag{9}$$

Let

$$A := \{i \in [d] : |A_i| \geq (1-\alpha)N\}, \qquad B := \{i \in [d] : |B_i| \geq (1-\alpha)N\},$$

noting that $A \cap B = \emptyset$ and $A \cup B \subseteq [d]$. Moreover, since each (A_i, B_i) is a partition of $[N]$, we have $|B_i| < \alpha N$, for all $i \in A$, and $|A_i| < \alpha N$ for all $i \in B$. Rewriting (8) yields

$$\begin{aligned}
(1-\alpha)c\gamma \sum_{i\in[d]} |\nabla f(x)_i| &< \frac{c\gamma}{N} \sum_{i\in A} |\nabla f(x)_i||B_i| + \frac{c\gamma}{N} \sum_{i\in B} |\nabla f(x)_i||B_i| + \frac{c\gamma}{N} \sum_{i\notin A\cup B} |\nabla f(x)_i||B_i| \\
&< \alpha c\gamma \sum_{i\in A} |\nabla f(x)_i| + \frac{c\gamma}{N} \sum_{i\in B} |\nabla f(x)_i||B_i| + (1-\alpha)c\gamma \sum_{i\notin A\cup B} |\nabla f(x)_i| \\
\iff \qquad (1-2\alpha) \sum_{i\in A} |\nabla f(x)_i| &< \sum_{i\in B} |\nabla f(x)_i| \left(\frac{|B_i|}{N} - (1-\alpha)\right) \leq \alpha \sum_{i\in B} |\nabla f(x)_i|.
\end{aligned}$$

It is easy to see that this implies

$$d\alpha \max_{i\in B} |\nabla f(x)_i| \geq (1-2\alpha) \max_{i\in A} |\nabla f(x)_i|. \tag{10}$$

Rewriting (9) in the similar vein yields

$$(1-2\alpha) \sum_{i\in B} |\nabla f(x)_i|^2 < \alpha \sum_{i\in A} |\nabla f(x)_i|^2,$$

which together with 10 implies

$$d\alpha \max_{i\in A} |\nabla f(x)_i|^2 > (1-2\alpha) \max_{i\in B} |\nabla f(x)_i|^2 > \frac{(1-2\alpha)^3}{(d\alpha)^2} \max_{i\in A} |\nabla f(x)_i|^2,$$

which holds only if $\alpha > \frac{1}{d+2}$, contradicting our assumption. Therefore, either $\texttt{err}(GD) \leq (1-\alpha)\gamma\|\nabla f(x)\|^2$ or $\texttt{err}(signGD_2) \leq (1-\alpha)c\gamma\|\nabla f(x)\|_1$, which proves Claim 1.

Continuing the proof of Theorem 6, we now bound $\frac{\gamma}{N} \sum_{n\in[N]} I_n^{(2)}$ by

$$\begin{aligned}
\frac{\gamma}{N} \sum_{n\in[N]} I_n^{(2)} &= \frac{\gamma}{N} \sum_{n\in[N]} \langle \nabla f(x_t), \texttt{clip}(F_n'(x_t), c) - \texttt{clip}(\texttt{top-k}(F_n'(x_t)), c)\rangle \\
\{\text{Cauchy-Schwarz inequality}\} \qquad &\leq \gamma\|\nabla f(x_t)\| \frac{1}{N} \sum_{n\in[N]} \|\texttt{clip}(F_n'(x_t), c) - \texttt{clip}(\texttt{top-k}(F_n'(x_t)), c)\|.
\end{aligned}$$

We will now bound the RHS in two different ways. First, by expanding different cases per coordinate, it is easy to see that

$$\|\texttt{clip}(F_n'(x_t), c) - \texttt{clip}(\texttt{top-k}(F_n'(x_t)), c)\| \leq \|F_n'(x_t) - \texttt{top-k}(F_n'(x_t))\| \leq \tau_k\|F_n'(x_t)\|,$$

by assumption. Therefore,

$$\frac{\gamma}{N} \sum_{n\in[N]} I_n^{(2)} \leq \gamma\|\nabla f(x_t)\|\tau_k \frac{1}{N} \sum_{n\in[N]} \|F_n'(x_t)\| \leq \gamma\tau_k M^2,$$

by the Cauchy-Schwarz inequality and the assumption $\frac{1}{N} \sum_{n\in[N]} \|F_n'(x_t)\|^2 \leq M^2$.

On the other hand, since $\|\texttt{clip}(F'_n(x_t), c) - \texttt{clip}(\texttt{top-k}(F'_n(x_t)), c)\| \leq c(d-k)$, we easily get

$$\frac{\gamma}{N} \sum_{n\in[N]} I_n^{(2)} \leq \gamma M \min\{\tau_k M, c(d-k)\}.$$

Combining the bounds with respect to $I_n^{(1)}$ and $I_n^{(2)}$ and adding easy analysis of different cases for scaling by c yields

$$-\frac{\gamma}{N} \sum_{n\in[N]} I_n(x_t) \leq -\frac{\gamma \min\{c,1\}}{d+2} \min\{\|\nabla f(x_t)\|^2, \|\nabla f(x_t)\|_1\} + \gamma \max\{\|\sigma\|^2 + \|\sigma\| M, 2\|\sigma\|_1\} + \gamma \min\{\tau_k M^2, c(d-k)M\}. \tag{11}$$

For $J_n(x_t)$ note that

$$\begin{aligned}
\frac{L\gamma^2}{N} \sum_{n\in[N]} J_n(x_t) &= \frac{L\gamma^2}{N} \sum_{n\in[N]} \mathbb{E}_{\xi_t} \left\|Q(\texttt{clip}(\texttt{top-k}(F'_n(x_t)), c), \xi_t)\right\|^2 \\
\left\{\|a+b\|^2 \leq 2\|a\|^2 + 2\|b\|^2\right\} \qquad &\leq \frac{2L\gamma^2}{N} \sum_{n\in[N]} \mathbb{E}_{\xi_t} \left\|Q(\texttt{clip}(\texttt{top-k}(F'_n(x_t)), c), \xi_t) - \texttt{clip}(\texttt{top-k}(F'_n(x_t)), c)\right\|^2 \\
&\quad + \frac{2L\gamma^2}{N} \sum_{n\in[N]} \|\texttt{clip}(\texttt{top-k}(F'_n(x_t)), c)\|^2 \\
\left\{\mathbb{E}_\xi\left[\|Q(x,\xi) - x\|^2\right] \leq \tilde{\sigma}^2\right\} \qquad &\leq \frac{2L\gamma^2}{N} \sum_{n\in[N]} \left(\tilde{\sigma}^2 + \|\texttt{clip}(\texttt{top-k}(F'_n(x_t)), c)\|^2\right) \\
\left\{\frac{1}{N} \sum_{n\in[N]} \left\|F'_n(x)\right\|^2 \leq M^2\right\} \qquad &\leq 2L\gamma^2(\tilde{\sigma}^2 + \min\{c^2, M^2\}).
\end{aligned}$$

Combining bounds on $I_n(x_t)$ and $J_n(x_t)$ yields

$$\begin{aligned}
\frac{\gamma \min\{c,1\}}{d+2} \min\{\|\nabla f(x_t)\|^2, \|\nabla f(x_t)\|_1\} &\leq f(x_t) - \mathbb{E}_{\mathcal{N}} \mathbb{E}_{\xi_t} f(x_{t+1}) \\
&\quad + \gamma \max\{\|\sigma\|^2 + \|\sigma\| M, 2\|\sigma\|_1\} + \gamma \min\{\tau_k M^2, c(d-k)M\} + 2L\gamma^2(\tilde{\sigma}^2 + \min\{c^2, M^2\}) + L\gamma^2 Ak.
\end{aligned}$$

Summing over all $t \in [T]$ yields

$$\begin{aligned}
\frac{\gamma \min\{c,1\}}{d+2} \sum_{t\in[T]} \min\{\mathbb{E}\|\nabla f(x_t)\|^2, \mathbb{E}\|\nabla f(x_t)\|_1\} &\leq f(x_0) - f(x^*) \\
&\quad + T\gamma\big(\min\{\tau_k M^2, c(d-k)M\} + L\gamma Ak + \max\{\|\sigma\|^2 + \|\sigma\| M, 2\|\sigma\|_1\} + 2L\gamma(\tilde{\sigma}^2 + \min\{c^2, M^2\})\big),
\end{aligned}$$

which after dividing with $T\gamma$ finishes the proof of the first part.

For the moreover part, in which no quantization is performed, *i.e.*, $Q(x,\xi) = x$, for all x (point-wise, not just on average), the calculation for $J_n(x_t)$ becomes

$$\frac{L\gamma^2}{N} \sum_{n\in[N]} J_n(x_t) = \frac{L\gamma^2}{N} \sum_{n\in[N]} \left\|\texttt{clip}(\texttt{top-k}(F'_n(x_t)), c)\right\|^2 \leq L\gamma^2 \min\{c^2, M^2\}.$$

Continuing the proof as above (summing over all $t \in [T]$ and dividing by $T\gamma$) finishes the proof of the second part.