Private GANs, Revisited

Anonymous authors
Paper under double-blind review

Abstract

We show that the canonical approach for training differentially private GANs – updating the discriminator with differentially private stochastic gradient descent (DPSGD) – can yield significantly improved results after modifications to training. Existing instantiations of this approach neglect to consider how adding noise *only to discriminator updates* disrupts the careful balance between the generator and discriminator necessary for successful GAN training. We show that a simple fix – taking more discriminator steps between generator steps – restores parity and improves results. Additionally, with the goal of restoring parity between the generator and discriminator, we experiment with other modifications to improve discriminator training and see further improvements in generation quality. Our results demonstrate that on standard benchmarks, DPSGD outperforms all alternative GAN privatization schemes.

1 Introduction

Differential privacy (DP) (Dwork et al., 2006b) has emerged as a compelling approach for training machine learning models on sensitive data. However, incorporating DP requires significant changes to the training process. Notably, it prevents the modeller from working directly with sensitive data, complicating debugging and exploration. Furthermore, upon exhausting their allocated privacy budget, the modeller is restricted from interacting with sensitive data. One approach to alleviate these issues is by producing *differentially private synthetic data*, which can be plugged directly into existing machine learning pipelines, without further concern for privacy.

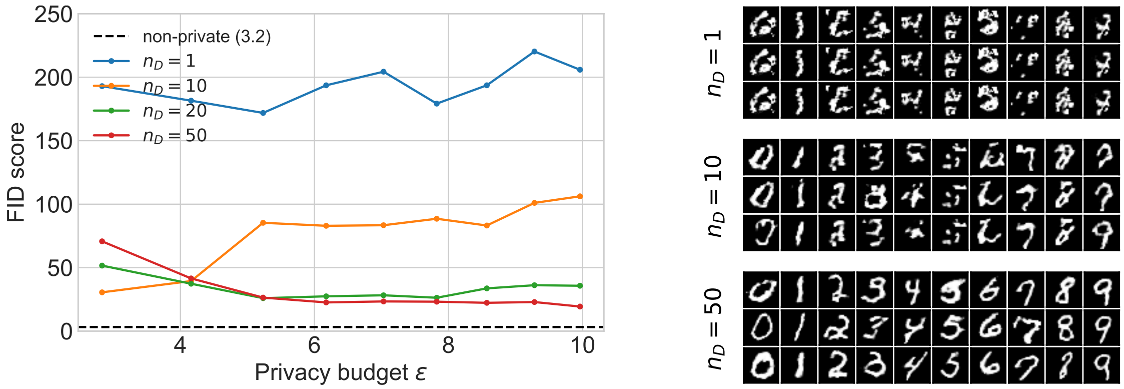
Towards generating high-dimensional, complex data (such as images), a line of work has examined privatizing generative adversarial networks (GANs) (Goodfellow et al., 2014) to produce DP synthetic data. Initial efforts proposed to use differentially private stochastic gradient descent (DPSGD) (Abadi et al., 2016) as a drop-in replacement for SGD to update the GAN discriminator – an approach referred to as *DPGAN* (Xie et al., 2018; Beaulieu-Jones et al., 2019; Torkzadehmahani et al., 2019). Follow-up work (Jordon et al., 2019; Long et al., 2021; Chen et al., 2020; Wang et al., 2021) departs from this approach, proposing alternative privatization schemes for GANs, and reports significant improvements over the DPGAN baseline.

However, even the best of these GAN-based schemes leave much to be desired, as they are associated with significant drops in utility (Table 1). Other methods for generating DP synthetic data diverge from GAN-based architectures, yielding improvements to utility in most cases (Table 2). This raises the question of whether GANs are suitable for DP training, or if bespoke architectures are required for DP synthetic data generation.

Our contributions. We show that DPGANs give far better utility than previously demonstrated, and compete with or outperform almost all other methods for DP synthetic data.[1] Previously demonstrated deficiencies of DPGANs should not be attributed to inherent limitations of the framework, but rather, training issues. Specifically, we propose that the *asymmetric noise addition* in DPGANs (adding noise to discriminator updates only) weakens the discriminator relative to the generator, disrupting the careful balance necessary for successful GAN training. We propose that taking more discriminator steps between generator

---

[1] A notable exception is diffusion models, discussed further in Section 2.



(a) MNIST FID over a training run

(b) Corresponding images at $(10, 10^{-5})$-DP

Figure 1: DPGAN results on MNIST synthesis at $(10, 10^{-5})$-DP. **(a)** We find that increasing $n_{\mathcal{D}}$, the number of discriminator steps taken between generator steps, significantly improves image synthesis. Increasing $n_{\mathcal{D}} = 1$ to $n_{\mathcal{D}} = 50$ improves FID from 205.9 → 19.4. **(b)** Corresponding synthesized images (each are trained with the same privacy budget). We observe that large $n_{\mathcal{D}}$ improves visual quality, and low $n_{\mathcal{D}}$ leads to mode collapse.

		MNIST		FashionMNIST		CelebA-Gender	
Privacy Level	Method	FID	Acc.(%)	FID	Acc.(%)	FID	Acc.(%)
$\varepsilon = \infty$	Real Data	1.0	99.2	1.5	92.5	1.1	96.6
	GAN	3.2	96.8	15.9	80.4	31.5	91.6
$\varepsilon = 10$	Best Private GAN	61.34	80.92	131.34	70.61	-	70.72
	DPGAN	179.16	80.11	243.80	60.98	-	54.09
$\varepsilon = 9.32$	Our DPGAN	13.0	95.0	56.8	74.8	157.6	86.9

Table 1: A summary of our results, compared to results reported in previous work on private GANs. *Acc.(%)* refers to downstream classification accuracy of CNN models trained with generated data. The middle two rows are a composite of the best results reported in the literature for DPGANs and other GAN privatization schemes (see Tables 2 and 3 for correspondences). Here we use Gopi et al. (2021) privacy accounting for our results. We find significant improvement over all previous GAN-based methods for DP synthetic data.

updates addresses the imbalance introduced by noise. With this change, DPGANs improve significantly (see Figure 1 and Table 1). Furthermore, we show this perspective on DPGAN training ("restoring parity to a discriminator weakened by DP noise") can be applied to improve training. We make other modifications to discriminator training – larger batch sizes and adaptive discriminator step frequency – to improve discriminator training and further improve upon the aforementioned results. In summary, we make the following contributions:

- We find that taking more discriminator steps between generator steps significantly improves DPGANs. Contrary to the previous results in the literature, DPGANs do compete with alternative GAN privatization schemes.
- We present empirical findings towards understanding why more frequent discriminator steps help. We propose an explanation based on *asymmetric noise addition* for why vanilla DPGANs do not perform well, and why taking more frequent discriminator steps helps.
- We employ our explanation as a principle for designing better private GAN training recipes, and indeed are able to improve over the aforementioned results.

2 Related work

Private GANs. The baseline DPGAN that employs a DPSGD-trained discriminator was introduced in Xie et al. (2018), and studied in follow-up work of Torkzadehmahani et al. (2019); Beaulieu-Jones et al. (2019). Despite significant interest in the approach (≈ 300 citations at time of writing), we were unable to find studies that explore the modifications we perform or uncover similar principles for improving training. As a consequence, subsequent work has departed from this approach, examining alternative privatization schemes for GANs (Jordon et al., 2019; Long et al., 2021; Chen et al., 2020; Wang et al., 2021). Broadly speaking, these approaches employ subsample-and-aggregate (Nissim et al., 2007) via the PATE approach (Papernot et al., 2017), dividing the data into $\geq$ 1K disjoint partitions and training teacher discriminators separately on each one. Our work shows that these privatization schemes are outperformed by DPSGD.

Other DP generative models. Other generative modelling frameworks have been applied to generate DP synthetic data: VAEs (Chen et al., 2018), maximum mean discrepancy (Harder et al., 2021; Vinaroz et al., 2022; Harder et al., 2022), Sinkhorn divergences (Cao et al., 2021), normalizing flows (Waites & Cummings, 2021), and diffusion models (Dockhorn et al., 2022). In a different vein, Chen et al. (2022) avoids learning a generative model, and instead generates a coreset of examples (≈ 20 per class) for the purpose of training a classifier. These approaches fall into two camps: applications of DPSGD to existing, highly-performant generative models; or custom approaches designed specifically for privacy which fall short of GANs when evaluated at their non-private limits ($\varepsilon \to \infty$).

Concurrent work on DP diffusion models. Simultaneous and independent work by Dockhorn et al. (2022) is the first to investigate DP training of diffusion models. They achieve impressive state-of-the-art results for generating DP synthetic data in a variety of settings, in particular, outperforming our results for DPGANs reported in this paper. We consider our results to still be of significant interest to the community, as we challenge the conventional wisdom regarding deficiencies of DPGANs, showing that they give much better utility than previously thought. Indeed, GANs are still one of the most popular and well-studied generative models, and consequently, there are many cases where one would prefer a GAN over an alternative approach. By revisiting several of the design choices in DPGANs, we give guidance on how to seamlessly introduce differential privacy into such pipelines. Furthermore, both our work and the work of Dockhorn et al. (2022) are aligned in supporting a broader message: training conventional machine learning architectures with DPSGD frequently achieves state-of-the-art results under differential privacy. Indeed, both our results and theirs outperform almost all custom methods designed for DP synthetic data. This reaffirms a similar message recently demonstrated in other private ML settings, including image classification (De et al., 2022) and NLP (Li et al., 2022; Yu et al., 2022).

DP tabular data synthesis. Our investigation focuses on image datasets, while many important applications of private data generation involve tabular data. In these settings, marginal-based approaches (Hardt et al., 2012; Zhang et al., 2017; McKenna et al., 2019) perform the best. While Tao et al. (2021) find that private GAN-based approaches fail to preserve even basic statistics in these settings, we believe that our techniques may yield similar improvements.

3 Preliminaries

Our goal is to train a generative model on sensitive data that is safe to release, i.e., it does not leak the secrets of individuals in the training dataset. We do this by ensuring the training algorithm $\mathcal{A}$ – which takes as input the sensitive dataset $D \in \mathcal{U}$ and returns the parameters of a trained (generative) model $\theta \in \Theta$ – satisfies differential privacy.

Definition 1 (Differential Privacy, Dwork et al. 2006b)**.** A randomized algorithm $\mathcal{A} : \mathcal{U} \to \Theta$ is (ε, δ)-*differentially private* if for every pair of neighbouring datasets $D, D' \in \mathcal{U}$, we have

$$\mathbb{P}\{\mathcal{A}(D) \in S\} \leq \exp(\varepsilon) \cdot \mathbb{P}\{\mathcal{A}(D') \in S\} + \delta \qquad \text{for all } S \subseteq \Theta.$$

Algorithm 1 TrainDPGAN($D; \cdot$)

```
Input: Labelled dataset $D = \{(x_j, y_j)\}_{j=1}^{n}$. Discriminator $\mathcal{D}$ and generator $\mathcal{G}$ initializations $\phi_0$ and $\theta_0$. Optimizers OptD, OptG. Privacy parameter $\delta$. Hyperparameters: $n_{\mathcal{D}}$ ($\mathcal{D}$ steps per $\mathcal{G}$ step), $T$ (total number of $\mathcal{D}$ steps), $B$ (expected batch size), $C$ (clipping norm), $\sigma$ (noise multiplier).
$q \leftarrow B/|D|$ and $t, k \leftarrow 0$    ▷ Calculate sampling rate $q$, initialize counters.
while $t < T$ do    ▷ Update $\mathcal{D}$ with DPSGD.
    $S_t \sim \text{PoissonSample}(D, q)$    ▷ Sample a real batch $S_t$ by including each $(x, y) \in D$ w.p. $q$.
    $\widetilde{S}_t \sim \mathcal{G}(\cdot; \theta_k)^B$    ▷ Sample fake batch $\widetilde{S}_t$.
    $g_{\phi_t} \leftarrow \sum_{(x,y)\in S_t} \text{clip}\left(\nabla_{\phi_t}(-\log(\mathcal{D}(x, y; \phi_t))); C\right)$
        $+ \sum_{(\widetilde{x},\widetilde{y})\in \widetilde{S}_t} \text{clip}\left(\nabla_{\phi_t}(-\log(1 - \mathcal{D}(\widetilde{x}, \widetilde{y}; \phi_t))); C\right)$    ▷ Clip per-example gradients.
    $\widehat{g}_{\phi_t} \leftarrow \frac{1}{2B}(g_{\phi_t} + z_t)$, where $z_t \sim \mathcal{N}(0, C^2\sigma^2 I)$    ▷ Add Gaussian noise.
    $\phi_{t+1} \leftarrow \texttt{OptD}(\phi_t, \widehat{g}_{\theta_t})$
    $t \leftarrow t + 1$
    if $n_{\mathcal{D}}$ divides $t$ then    ▷ Perform $\mathcal{G}$ update every $n_{\mathcal{D}}$ steps.
        $\widetilde{S}'_t \sim \mathcal{G}(\cdot; \theta_k)^B$
        $g_{\theta_k} \leftarrow \frac{1}{B} \sum_{(\widetilde{x},\widetilde{y})\in \widetilde{S}'_t} \nabla_{\theta_k}(-\log(\mathcal{D}(\widetilde{x}, \widetilde{y}; \phi_t)))$
        $\theta_{k+1} \leftarrow \texttt{OptG}(\theta_k, g_{\theta_k})$
        $k \leftarrow k + 1$
    end if
end while
$\varepsilon \leftarrow \text{PrivacyAccountant}(T, \sigma, q, \delta)$    ▷ Compute privacy budget spent.
Output: Final $\mathcal{G}$ parameters $\theta_k$. $(\varepsilon, \delta)$-DP guarantee.
```

In this work, we adopt the add/remove definition of DP, and say two datasets D and D' are neighbouring if they differ in at most one entry, that is, $D = D' \cup \{x\}$ or $D' = D \cup \{x\}$.

We highlight one convenient property of DP, known as *closure under post-processing*. This says that interacting with a privatized model (e.g., using it to compute gradients on non-sensitive data, generate samples) does not lead to any further privacy violation.

Proposition 2 (Post-processing)**.** Let $\mathcal{A} : \mathcal{U} \to \Theta$ be a randomized algorithm that is (ε, δ)-DP, and $f : \Theta \to \mathcal{Y}$ be an arbitrarily randomized mapping. Then $f \circ \mathcal{A} : \mathcal{U} \to \mathcal{Y}$ is (ε, δ)-DP.

DPSGD. A gradient-based learning algorithm can be privatized by employing *differentially private stochastic gradient descent* (DPSGD) (Song et al., 2013; Bassily et al., 2014; Abadi et al., 2016) as a drop-in replacement for SGD. DPSGD involves clipping per-example gradients and adding Gaussian noise to their sum, which effectively bounds and masks the contribution of any individual point to the final model parameters. Privacy analysis of DPSGD follows from several classic tools in the DP toolbox: Gaussian mechanism, privacy amplification by subsampling, and composition (Dwork et al., 2006a; Dwork & Roth, 2014; Abadi et al., 2016; Wang et al., 2019). In our work, we use two different privacy accounting methods for DPSGD: (a) the classical approach of Mironov et al. (2019), implemented in Opacus (Yousefpour et al., 2021), and (b) the recent exact privacy accounting of Gopi et al. (2021). By default, we use the former technique for a closer direct comparison with prior works (though we note that some prior works use even looser accounting techniques). However, the latter technique gives tighter bounds on the true privacy loss, and for all practical purposes, is the preferred method of privacy accounting. We use Gopi et al. (2021) accounting only where indicated in Tables 1, 2, and 3.

DPGANs. Algorithm 1 details the training algorithm for DPGANs, which is effectively an instantiation of DPSGD. Note that only gradients for the discriminator $\mathcal{D}$ must be privatized (via clipping and noise), and not those for the generator $\mathcal{G}$. This is a consequence of post-processing (Proposition 2) – the generator only interacts with the sensitive dataset indirectly via discriminator parameters, and therefore does not need further privatization.

4 Frequent discriminator steps improves private GANs

In this section, we discuss our main finding: the number of discriminator steps taken between each generator step ($n_{\mathcal{D}}$ from Algorithm 1) plays a significant role in the success of private GAN training. For a fixed setting of DPSGD hyperparameters, there is an optimal range of values for $n_{\mathcal{D}}$ that maximizes generation quality, in terms of both visual quality and utility for downstream classifier training. This value can be quite large ($n_{\mathcal{D}} \approx 100$ in some cases).

4.1 Experimental details

Setup. We focus on labelled generation of MNIST (LeCun et al., 1998) and FashionMNIST (Xiao et al., 2017), both of which are comprised of 60K 28×28 grayscale images divided into 10 classes. To build a strong baseline, we begin from an open source PyTorch (Paszke et al., 2019) implementation[2] of DCGAN (Radford et al., 2016) that performs well non-privately, and copy their training recipe. We then adapt their architecture to our purposes: removing BatchNorm layers (which are not compatible with DPSGD) and adding label embedding layers to enable labelled generation. Training this configuration non-privately yields labelled generation that achieves FID scores of 3.2 on MNIST and 15.9 on FashionMNIST. Finally, we note that these models are not small: $\mathcal{D}$ and $\mathcal{G}$ have 1.72M and 2.27M trainable parameters respectively. For further details, please see Appendix B.1.

Privacy implementation. To privatize training, we use Opacus (Yousefpour et al., 2021) which implements per-example gradient computation. As discussed before, we use the Rényi differential privacy (RDP) accounting of Mironov et al. (2019) (except in a few noted instances, where we instead use the tighter Gopi et al. (2021) accounting). For our baseline setting, we use the following DPSGD hyperparameters: we keep the non-private (expected) batch size $B = 128$, and use a noise scale $\sigma = 1$ and clipping norm $C = 1$. Under these settings, we have the budget for $T = 450\text{K}$ discriminator steps when targeting $(10, 10^{-5})$-DP.

Evaluation. We evaluate our generative models by examining the *visual quality* and *utility for downstream tasks* of generated images. Following prior work, we measure visual quality by computing the Fréchet Inception Distance (FID) (Heusel et al., 2017) between 60K generated images and entire test set.[3] To measure downstream task utility, we again follow prior work, and train a CNN classifier on 60K generated image-label pairs and report its accuracy on the real test set.

4.2 Results

More frequent discriminator steps improves generation. We plot in Figures 1a and 2 the evolution of FID and downstream accuracy during DPGAN training for both MNIST and FashionMNIST, under varying discriminator update frequencies $n_{\mathcal{D}}$. The effect of this parameter has outsized impact on the final results. For MNIST, $n_{\mathcal{D}} = 50$ yields the best results; on FashionMNIST, the best FID is obtained at $n_{\mathcal{D}} = 200$ and the best accuracy at $n_{\mathcal{D}} = 100$.

We emphasize that increasing the *frequency* of discriminator steps, relative to generator steps, does not affect the privacy cost of Algorithm 1. For any setting of $n_{\mathcal{D}}$, we perform the same number of noisy gradient queries on real data – what changes is the total number of generator steps taken over the course of training, which is reduced by a factor of $n_{\mathcal{D}}$.

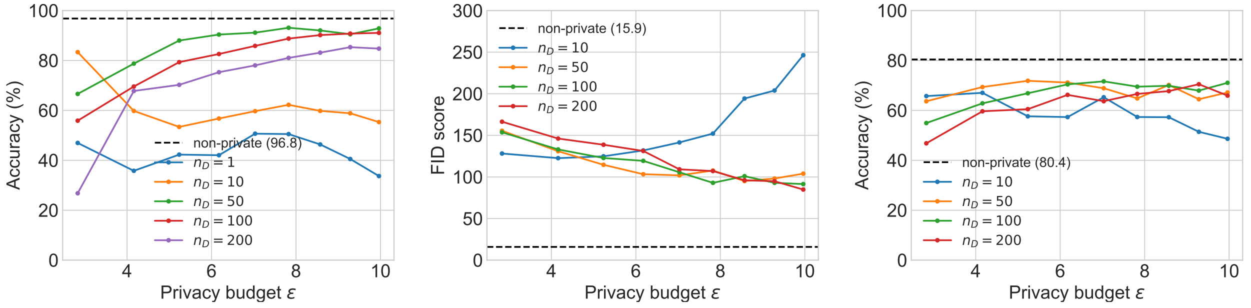
Private GANs are on a path to mode collapse. For the MNIST results in Figures 1a and 2a, we observe that at low discriminator update frequencies ($n_{\mathcal{D}} = 10$), the best FID and accuracy scores occur early in training, *well before the privacy budget we are targeting is exhausted.*[4] In fact, at 50K discriminator steps ($\varepsilon \approx 2.85$), $n_{\mathcal{D}} = 10$ has better FID (30.6) and accuracy (83.3%) than other settings of $n_{\mathcal{D}}$. However,

---

[2] Courtesy of Hyeonwoo Kang (`https://github.com/znxlwm`). Code available at this link.

[3] We use an open source PyTorch implementation to compute FID: `https://github.com/mseitzer/pytorch-fid`.

[4] This observation has been reported in Neunhoeffer et al. (2021), serving as motivation for their remedy of taking a mixture of intermediate models encountered in training. We are not aware of any mentions of this aspect of DPGAN training in papers reporting DPGAN baselines for labelled image synthesis.



(a) MNIST accuracy

(b) FashionMNIST FID

(c) FashionMNIST accuracy

Figure 2: DPGAN results over training runs using different discriminator update frequencies $n_{\mathcal{D}}$, targeting $(10, 10^{-5})$-DP. Each plotted line indicates the utility of a model over a single training run, as the privacy budget is expended. **(a)** As a measure of synthetic data utility, we plot the test set accuracy of a CNN trained on generated data only. Accuracy mirrors the FID scores from Figure 1a. Going from $n_{\mathcal{D}} = 1$ to $n_{\mathcal{D}} = 50$ improves accuracy from 33.7% → 92.9%. Further $n_{\mathcal{D}}$ increases hurt accuracy. **(b) and (c)** We obtain similar results for FashionMNIST. Note that the optimal $n_{\mathcal{D}}$ is higher (around $n_{\mathcal{D}} \approx 100$). At $n_{\mathcal{D}} = 100$, we obtain an FID of 91.5 and accuracy of 71.1%.

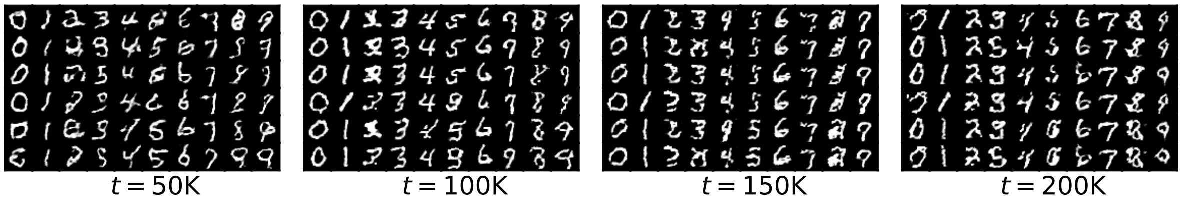


Figure 3: Evolution of samples drawn during training with $n_{\mathcal{D}} = 10$, when targeting $(10, 10^{-5})$-DP. This setting reports its best FID and downstream accuracy at $t = 50$K iterations ($\varepsilon \approx 2.85$). As training progresses beyond this point, we observe mode collapse for several classes (e.g., the 6's and 7's, particularly at $t = 150$K), co-occuring with the deterioration in evaluation metrics (these samples correspond to the first 4 data points in the $n_{\mathcal{D}} = 10$ line in Figures 1a and 2a).

these results deteriorate with continued training. In Figure 3, we plot the evolution of generated images for this $n_{\mathcal{D}} = 10$ run over the course of training and observe qualitative evidence of mode collapse, co-occurring with the deterioration in FID and accuracy observed in the first 4 data points of the $n_{\mathcal{D}} = 10$ run in Figures 1a and 2a.

An optimal discriminator update frequency. These results suggest that *fixing other DPSGD hyperparameters, there is an optimal setting for the discriminator step frequency* $n_{\mathcal{D}}$ that strikes a balance between: (a) being too low, causing the generation quality to peak early in training and then undergo mode collapse; resulting in all subsequent training to consume additional privacy budget *without improving the model*; and (b) being too high, preventing the generator from taking enough steps to converge before the privacy budget is exhausted (an example of this is the $n_{\mathcal{D}} = 200$ run in Figure 2a). Striking this balance results in the most effective utilization of privacy budget towards improving the generator.

5 Why does taking more steps help?

In this section, we present empirical findings towards understanding why more frequent discriminator steps improves DPGAN training. We propose an explanation that is conistent with our findings.

How does DP affect GAN training? Figure 4 compares the accuracy of the GAN discriminator on held-out real and fake examples immediately before each generator step, between private and non-private training with different settings of $n_{\mathcal{D}}$. We observe that non-privately at $n_{\mathcal{D}} = 1$, discriminator accuracy stabilizes at around 60%. Naively introducing DP ($n_{\mathcal{D}} = 1$) leads to a qualitative difference: DP causes

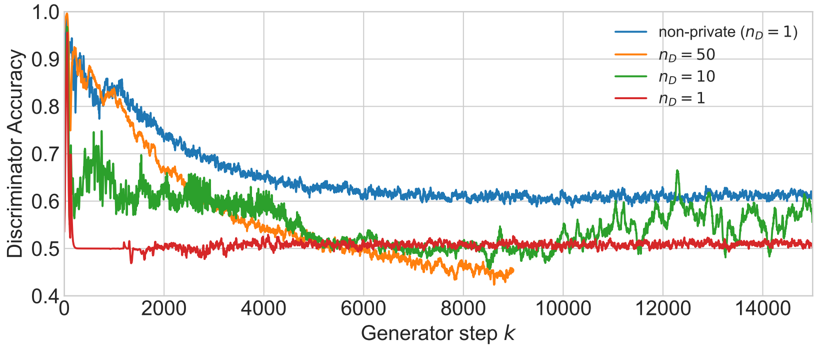


Figure 4: Exponential moving average ($\beta = 0.95$) of GAN discriminator accuracy on mini-batches, immediately before each generator step. While non-privately the discriminator maintains a 60% accuracy, the private discriminator with $n_{\mathcal{D}} = 1$ is effectively a random guess. Increasing the number of discriminator steps recovers the discriminator's advantage early on, leading to generator improvement. As the generator improves, the discriminator's task is made more difficult, driving down accuracy.

discriminator accuracy to drop to 50% (i.e., comparable accuracy to randomly guessing) immediately at the start of training, to never recover.[5]

For other settings of $n_{\mathcal{D}}$, we make following observations: (1) larger $n_{\mathcal{D}}$ corresponds to higher discriminator accuracy in early training; (2) in a training run, discriminator accuracy decreases throughout as the generator improves; (3) after discriminator accuracy falls below a certain threshold, the generator degrades or sees limited improvement.[6] Based on these observations, we propose the following explanation for why more frequent discriminator steps help:

- Generator improvement occurs when the discriminator is effective at distinguishing between real and fake data.
- The *asymmetric noise addition* introduced by DP to the discriminator makes such a task difficult, resulting in limited generator improvement.
- Allowing the discriminator to train longer on a fixed generator improves its accuracy, recovering the non-private case where the generator and discriminator are balanced.

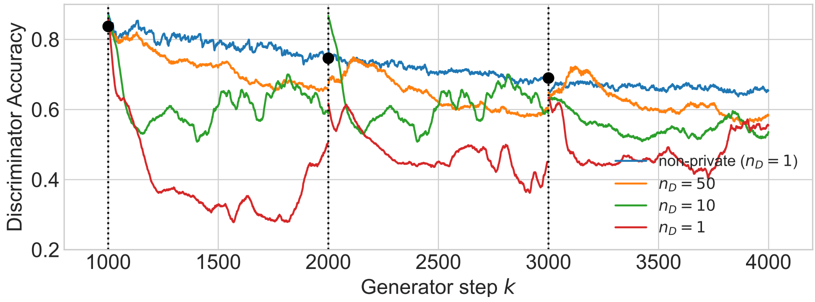
Checkpoint restarting experiment. We perform a checkpoint restarting experiment to examine this explanation in a more controlled setting. We train a non-private GAN for 3K generator steps, and save checkpoints of $\mathcal{D}$ and $\mathcal{G}$ (and their respective optimizers) at 1K, 2K, and 3K generator steps. We restart training from each of these checkpoints for 1K generator steps under different $n_{\mathcal{D}}$ and privacy settings. We plot the progression of discriminator accuracy, FID, and downstream classification accuracy. Results are pictured in Figure 5. Broadly, our results corroborate the observations that discriminator accuracy improves with larger $n_{\mathcal{D}}$ and decreases with better generators, and that generator improvement occurs when the discriminator has sufficiently high accuracy.

Does reducing noise accomplish the same thing? In light of the above explanation, we ask if reducing the noise level σ can offer the same improvement as taking more steps, as reducing σ should also improve discriminator accuracy before a generator step. To test this: starting from our setting in Section 4, fixing $n_{\mathcal{D}} = 1$, and targeting MNIST at $\varepsilon = 10$, we search over a grid of noise levels σ (the lowest of which, $\sigma = 0.4$, admits a budget of only $T = 360$ discriminator steps). Results are pictured in Figure 6. We obtain a best FID of 127.1 and best accuracy of 57.5% at noise level $\sigma = 0.45$. Hence we can conclude that in

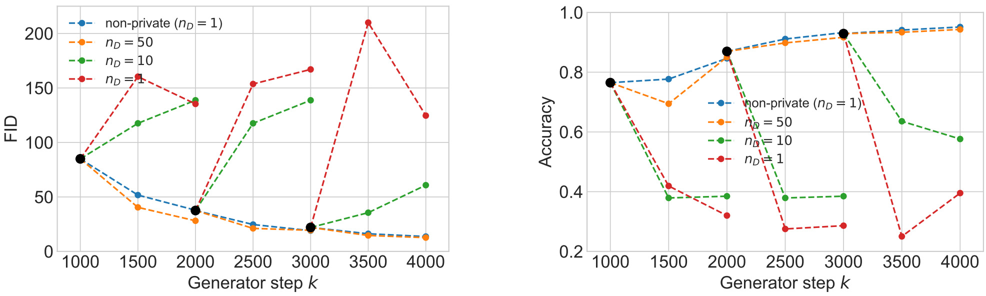
---

[5] Our plot only shows the first 15K generator steps, but we remark that this persists until the end of training (450K steps).

[6] For $n_{\mathcal{D}} = 10$, accuracy falls below 50% after 5K $\mathcal{G}$ steps (= 50K $\mathcal{D}$ steps), which corresponds to the first point in the $n_{\mathcal{D}} = 10$ line in Figures 1a and 2a. For $n_{\mathcal{D}} = 50$, accuracy falls below 50% after 5K $\mathcal{G}$ steps (= 250K $\mathcal{D}$ steps), which corresponds to the 5th point in the $n_{\mathcal{D}} = 50$ line in Figures 1a and 2a.



(a) Exponential moving average ($\beta = 0.95$) of discriminator accuracy on mini-batches, after checkpoint restarts



(b) FID after checkpoint restarts

(c) Accuracy after checkpoint restarts

Figure 5: We restart training under various privacy and $n_{\mathcal{D}}$ settings at 3 checkpoints taken at 1K, 2K, and 3K generator steps into non-private training. We plot the progression of discriminator accuracy, FID, and downstream classification accuracy. The black dots correspond to the initial values of a checkpoint. We observe that low $n_{\mathcal{D}}$ settings do not achieve comparable discriminator accuracy to non-private training (a), and results in degradation of utility ((b) and (c)). Discriminator accuracy for $n_{\mathcal{D}} = 50$ tracks non-private training, and we observe utility improvement throughout training like in the non-private setting.

this experimental setting, incorporating discriminator update frequency in our design space allows for more effective use of privacy budget for improving generation quality.

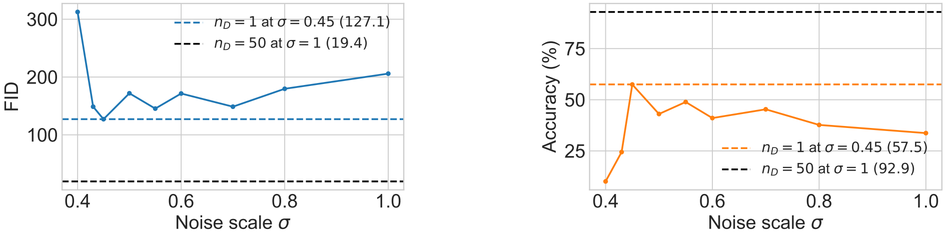
Does taking more discriminator steps always help? As we discuss in more detail in Section 6.1, when we are able to find other means to improve the discriminator beyond taking more steps, tuning discriminator update frequency may not yield improvements. To illustrate with an extreme case, consider eliminating the privacy constraint. In non-private GAN training, taking more steps is known to be unnecessary. We corroborate this result: we run our non-private baseline from Section 4 with the same number of generator steps, but opt to take 10 discriminator steps between each generator step instead of 1. FID worsens from 3.2 → 8.3, and accuracy worsens from 96.8% → 91.3%.

6 Better generators via better discriminators

Our proposed explanation in Section 5 provides a concrete suggestion for improving GAN training: effectively use our privacy budget to maximize the number of generator steps taken when the discriminator has sufficiently high accuracy. We experiment with modifications to the private GAN training recipe towards these ends, which translate to improved generation.

6.1 Larger batch sizes

Several recent works have demonstrated that for classification tasks, DPSGD achieves higher accuracy with larger batch sizes, after tuning the noise scale σ accordingly (Tramèr & Boneh, 2021; Anil et al., 2022; De et al., 2022). GAN training is typically conducted with small batch sizes (for example, DCGAN uses



(a) Varying σ only vs. FID at $n_{\mathcal{D}} = 1$

(b) Varying σ only vs. accuracy at $n_{\mathcal{D}} = 1$

Figure 6: On MNIST, we fix $n_{\mathcal{D}} = 1$ and report results for various settings of the DPSGD noise scale σ, where the number of iterations T is chosen for each σ to target $(10, 10^{-5})$-DP. The gap between the dashed lines represent the advancement of the utility frontier by incorporating the choice of $n_{\mathcal{D}}$ into our design space.

$B = 128$, which we adopt; StyleGAN uses $B = 32$). Therefore it is interesting to see if large batch sizes indeed improve private GAN training. We corroborate that larger batch sizes do not significantly improve our non-private MNIST baseline from Section 4: when we go up to $B = 2048$ from $B = 128$, FID stays at 3.2 and accuracy improves from 96.8% → 97.5%.

Results. We scale up batch sizes, considering $B \in \{64, 128, 512, 2048\}$, and search for the optimal noise scale σ and $n_{\mathcal{D}}$ (details in Appendix B.2). We target both $\varepsilon = 1$ and $\varepsilon = 10$. We report the best results from our hyperparameter search in in Table 2. We find that larger batch sizes leads to improvements: for both $\varepsilon = 1$ and $\varepsilon = 10$, the best results are achieved at $B = 512$ and $B = 2048$ respectively. We also note that for large batch sizes, the optimal number of generator steps can be quite small. For $B = 2048$, $\sigma = 4.0$, targeting MNIST at $\varepsilon = 10$, $n_{\mathcal{D}} = 5$ is the optimal discriminator update frequency, and improves over our best $B = 128$ setting employing $n_{\mathcal{D}} = 50$.

6.2 Adaptive discriminator step frequency

Our observations from Section 4 and 5 motivate us to consider *adaptive* discriminator step frequencies. As pictured in Figures 4 and 5a, discriminator accuracy drops during training as the generator improves. In this scenario, we want to take more steps to improve the discriminator, in order to further improve the generator. However, using a large discriminator update frequency right from the beginning of training is wasteful – as evidenced by the fact that low $n_{\mathcal{D}}$ achieves the best FID and accuracy early in training. Hence we propose to start at a low discriminator update frequency ($n_{\mathcal{D}} = 1$), and ramp up when our discriminator is performing poorly. Accuracy on real data must be released with DP. While this is feasible, it introduces the additional problem of having to find the right split of privacy budget for the best performance. We observe that discriminator accuracy is related to discriminator accuracy on fake samples only (which are free to evaluate on, by post-processing). Hence we use it as a proxy to assess discriminator performance.

We propose an *adaptive step frequency*, parameterized by β and d. β is the decay parameter used to compute the exponential moving average (EMA) of discriminator accuracy on fake batches before each generator update. d is the accuracy floor that upon reaching, we move to the next update frequency $n_{\mathcal{D}} \in \{1, 2, 5, 10, 20, 50, 100, 200, 500, 1000, ...\}$. Additionally, we promise a grace period of $2/(1-\beta)$ generator steps before moving on to the next update frequency – motivated by the fact that β-EMA's value is primarily determined by its last $2/(1-\beta)$ observations.

We use $\beta = 0.99$ in all settings, and try $d = 0.6$ and $d = 0.7$, finding that 0.7 works better for large batches. The additional benefit of the adaptive step frequency is that it means we do not have to search for the optimal update frequency. Although the adaptive step frequency introduces the extra hyperparameter of the threshold d, we found that these two settings ($d = 0.6$ and $d = 0.7$) were sufficient to improve over results of a much more extensive hyperparameter search over $n_{\mathcal{D}}$ (whose optimal value varied significantly based on the noise level σ and expected batch size B).

			MNIST		FashionMNIST	
Privacy Level	Method	Reported In	FID	Acc.(%)	FID	Acc.(%)
$\varepsilon = \infty$	Real data	(This work)	1.0	99.2	1.5	92.5
	GAN		3.2	96.8	15.9	80.4
$\varepsilon = 10$	DP-MERF	Cao et al. (2021)	116.3	82.1	132.6	75.5
	DP-Sinkhorn	Cao et al. (2021)	48.4	83.2	128.3	75.1
	PSG[7]	Chen et al. (2022)	-	95.6	-	77.7
	DPDM	Dockhorn et al. (2022)	5.01	97.3	18.6	84.9
	DPGAN[8]	Chen et al. (2020)	179.16	63	243.80	50
		Long et al. (2021)	304.86	80.11	433.38	60.98
	GS-WGAN	Chen et al. (2020)	61.34	80	131.34	65
	PATE-GAN	Long et al. (2021)	253.55	66.67	229.25	62.18
	G-PATE	Long et al. (2021)	150.62	80.92	171.90	69.34
	DataLens	Wang et al. (2021)	173.50	80.66	167.68	70.61
$\varepsilon = 9.32$*	Our DPGAN	(This work)	19.4	92.9	91.5	71.1
	+ large batches		13.2	94.3	66.7	72.1
	+ adaptive $n_{\mathcal{D}}$		13.0	95.0	56.8	74.8
$\varepsilon = 1$	DP-MERF[9]	Vinaroz et al. (2022)	-	80.7	-	73.9
	DP-HP	Vinaroz et al. (2022)	-	81.5	-	72.3
	PSG	Chen et al. (2022)	-	80.9	-	70.2
	DPDM	Dockhorn et al. (2022)	23.4	95.3	37.8	79.4
	DPGAN	Long et al. (2021)	470.20	40.36	472.03	10.53
	GS-WGAN	Long et al. (2021)	489.75	14.32	587.31	16.61
	PATE-GAN	Long et al. (2021)	231.54	41.68	253.19	42.22
	G-PATE	Long et al. (2021)	153.38	58.80	214.78	58.12
	DataLens	Wang et al. (2021)	186.06	71.23	194.98	64.78
$\varepsilon = 0.912$*	Our DPGAN	(This work)	91.7	77.4	151.9	65.0
	+ large batches		66.1	73.7	153.2	66.6
	+ adaptive $n_{\mathcal{D}}$		56.2	80.1	121.8	68.0

Table 2: We gather previously reported results in the literature on the performance of various methods for labelled generation of MNIST and FashionMNIST, compared with our results. Note that *Reported In* refers to the source of the numerical result, not the originator of the approach. For downstream accuracy, we report the best accuracy among classifiers they use, and compare against our CNN classifier accuracy. **(*)** For our results, we target $\varepsilon = 10/\varepsilon = 1$ with Opacus accounting and additionally report ε using the improved privacy accounting of Gopi et al. (2021).

6.3 Comparison with previous results in the literature

6.3.1 MNIST and FashionMNIST

Table 2 summarizes our best experimental settings for MNIST and FashionMNIST, and situates them in the context of previously reported results for the task. We also present a visual comparison in Figure 7. We provide some examples of generated images in Figures 9 and 10 for $\varepsilon = 10$, and Figures 11 and 12 for $\varepsilon = 1$.

Plain DPSGD beats all alternative GAN privatization schemes. Our baseline DPGAN from Section 4, with the appropriate choice of $n_{\mathcal{D}}$ (and without the modifications described in this section yet), outperforms all other GAN-based approaches proposed in the literature (GS-WGAN, PATE-GAN, G-PATE, and DataLens) *uniformly* across both metrics, both datasets, and both privacy levels.

---

[7] Since PSG produces a coreset of only 200 examples (20 per class), the covariance of its InceptionNet-extracted features is singular, and therefore it is not possible to compute an FID score.

[8] We group per-class unconditional GANs together with conditional GANs under the DPGAN umbrella.

[9] Results from Vinaroz et al. (2022) are presented graphically in the paper. Exact numbers can be found in their code.

Privacy	Method	Reported In	FID	Acc.(%)
$\varepsilon = \infty$	Real data	(This work)	1.1	96.6
	GAN		31.5	91.6
$\varepsilon = 10$	DP-MERF	Cao et al. (2021)	274.0	65
	DP-Sinkhorn	Cao et al. (2021)	189.5	76.3
	DPDM	Dockhorn et al. (2022)	21.1	-
	DPGAN	Long et al. (2021)	-	54.09
	GS-WGAN	Long et al. (2021)	-	63.26
	PATE-GAN	Long et al. (2021)	-	58.70
	G-PATE	Long et al. (2021)	-	70.72
$\varepsilon = 9.39$*	Our DPGAN	(This work)	157.6	86.9
$\varepsilon = 10$	DPGAN	Long et al. (2021)	485.41	52.11
	GS-WGAN	Long et al. (2021)	432.58	61.36
	PATE-GAN	Long et al. (2021)	424.60	65.35
	G-PATE	Long et al. (2021)	305.92	68.97
	DataLens	Wang et al. (2021)	320.84	72.87

Table 3: **Top section of the table:** Comparison to state-of-the-art results on 32×32 CelebA-Gender, targeting $(\varepsilon, 10^{-6})$-DP (except for the results reported in Long et al. (2021) which target a weaker $(\varepsilon, 10^{-5})$-DP). **(*)** For our results, we target $\varepsilon = 10$ with Opacus accounting and additionally report ε using the improved privacy accounting of Gopi et al. (2021). DPDM reports a much better FID score that our DPGAN (which itself, is an improvement over previous results). Our DPGAN achieves the best reported accuracy score. **Bottom section of the table:** Results for GAN-based approaches reported in Long et al. (2021) and Wang et al. (2021), which are not directly comparable because they target $(10, 10^{-5})$-DP and use 64×64 CelebA-Gender.

Large batch sizes and adaptive discriminator step frequency improve GAN training. Broadly speaking, across both privacy levels and both datasets, we see an improvement from taking larger batch sizes, and then another with the adaptive step frequency.

Comparison with state-of-the-art. With the exception of DPDM, our best DPGANs are competitive with state-of-the-art approaches for DP synthetic data, especially in terms of FID scores.

6.3.2 CelebA-Gender

We also report results on generating 32×32 CelebA, conditioned on gender at $(10, 10^{-6})$-DP. For these experiments, we used slightly larger models (2.64M and 3.16M parameters for $\mathcal{D}$ and $\mathcal{G}$ respectively), and employed large batches ($B = 2048$) and adaptive discriminator step frequency with threshold $d = 0.6$. Results are summarized in Table 3 and visualized in Figure 8. For more example generations, see Figure 13.

7 Conclusion

We revisit differentially private GANs and show that, with appropriate tuning of the training procedure, they can perform dramatically better than previously thought. Some crucial modifications include increasing the number of discriminator steps, increasing the batch size, and introducing adaptive discriminator step frequency. We explore the hypothesis that the previous deficiencies of DPGANs were due to poor classification accuracy of the discriminator. More broadly, our work supports the recurring finding that carefully-tuned DPSGD on conventional architectures can yield strong results for differentially private machine learning.

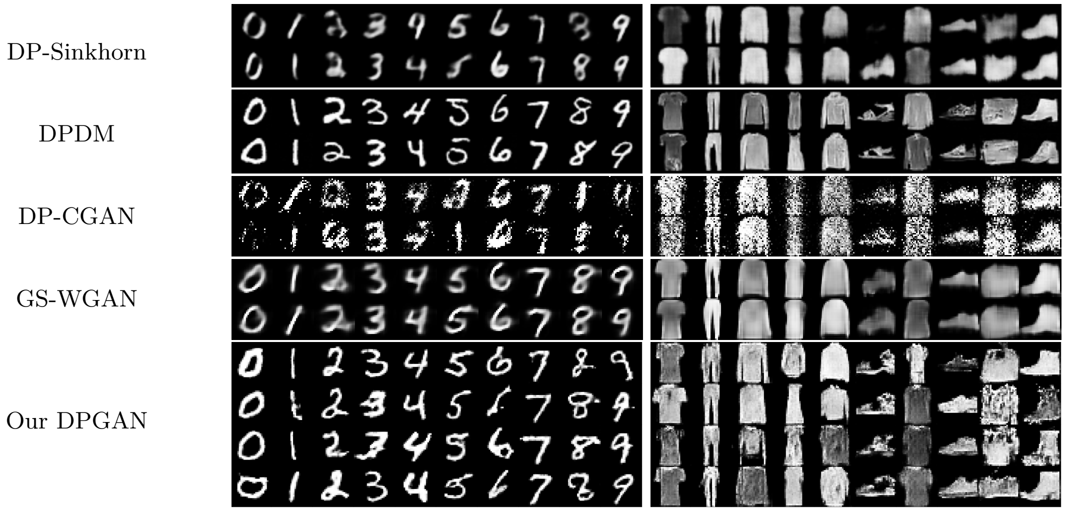


Figure 7: MNIST and FashionMNIST results at $(10, 10^{-5})$-DP for different methods. Images of other methods are from Cao et al. (2021) and Dockhorn et al. (2022).

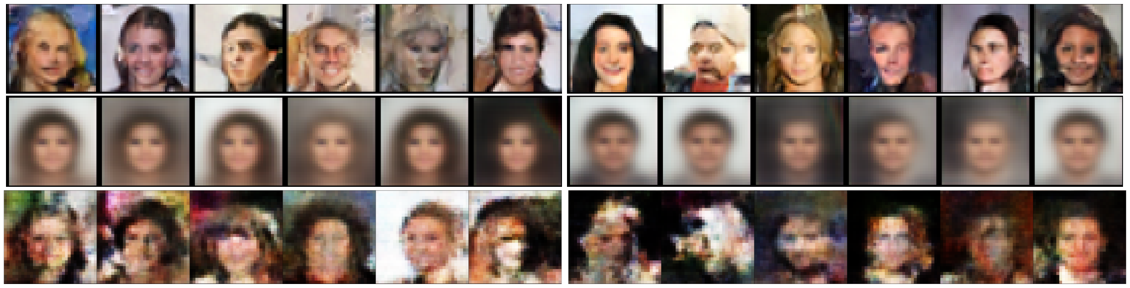


Figure 8: 32×32 CelebA-Gender at $(10, 10^{-6})$-DP. **From top to bottom:** DPDM (unconditional generation), DP-Sinkhorn, and our DPGAN. Images of other methods are from Cao et al. (2021) and Dockhorn et al. (2022).

References

Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *CCS'16: 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016.

Rohan Anil, Badih Ghazi, Vineet Gupta, Ravi Kumar, and Pasin Manurangsi. Large-scale differentially private BERT. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, 2022.

Raef Bassily, Adam Smith, and Abhradeep Thakurta. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science (FOCS'14)*, 2014.

Brett K. Beaulieu-Jones, Zhiwei Steven Wu, Chris Williams, Ran Lee, Sanjeev P. Bhavnani, James Brian Byrd, and Casey S. Greene. Privacy-preserving generative deep neural networks support clinical data sharing. *Circulation: Cardiovascular Quality and Outcomes*, 12(7), 2019.

Tianshi Cao, Alex Bie, Arash Vahdat, Sanja Fidler, and Karsten Kreis. Don't generate me: Training differentially private generative models with Sinkhorn divergence. In *Advances in Neural Information Processing Systems 34 (NeurIPS'21)*, 2021.

Dingfan Chen, Tribhuvanesh Orekondy, and Mario Fritz. GS-WGAN: A gradient-sanitized approach for learning differentially private generators. In *Advances in Neural Information Processing Systems 33 (NeurIPS'20)*, 2020.

Dingfan Chen, Raouf Kerkouche, and Mario Fritz. Private set generation with discriminative information. In *Advances in Neural Information Processing Systems 35 (NeurIPS'22)*, 2022.

Qingrong Chen, Chong Xiang, Minhui Xue, Bo Li, Nikita Borisov, Dali Kaafar, and Haojin Zhu. Differentially private data generative models. *CoRR*, abs/1812.02274, 2018.

Soumith Chintala, Emily Denton, Martin Arjovsky, and Michael Mathieu. How to train a GAN? Tips and tricks to make GANs work. *GitHub*, 2016.

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

Tim Dockhorn, Tianshi Cao, Arash Vahdat, and Karsten Kreis. Differentially private diffusion models. *CoRR*, abs/2210.09929, 2022.

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

Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *25th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT'06)*, 2006a.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography*, 2006b.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27 (NIPS'14)*, 2014.

Sivakanth Gopi, Yin Tat Lee, and Lukas Wutschitz. Numerical composition of differential privacy. In *Advances in Neural Information Processing Systems 34*, NeurIPS '21. Curran Associates, Inc., 2021.

Frederik Harder, Kamil Adamczewski, and Mijung Park. DP-MERF: Differentially private mean embeddings with random features for practical privacy-preserving data generation. In *24th International Conference on Artificial Intelligence and Statistics (AISTATS'21)*, 2021.

Frederik Harder, Milad Jalali Asadabadi, Danica J. Sutherland, and Mijung Park. Differentially private data generation needs better features. *CoRR*, abs/2205.12900, 2022.

Moritz Hardt, Katrina Ligett, and Frank Mcsherry. A simple and practical algorithm for differentially private data release. In *Advances in Neural Information Processing Systems 25 (NIPS'12)*, 2012.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems 30 (NIPS'17)*, 2017.

James Jordon, Jinsung Yoon, and Mihaela van der Schaar. PATE-GAN: Generating synthetic data with differential privacy guarantees. In *7th International Conference on Learning Representations (ICLR'19)*, 2019.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations (ICLR'15)*, 2015.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324, 1998.

Xuechen Li, Florian Tramèr, Percy Liang, and Tatsunori Hashimoto. Large language models can be strong differentially private learners. In *10th International Conference on Learning Representations (ICLR'22)*, 2022.

Yunhui Long, Boxin Wang, Zhuolin Yang, Bhavya Kailkhura, Aston Zhang, Carl Gunter, and Bo Li. G-PATE: Scalable differentially private data generator via private aggregation of teacher discriminators. In *Advances in Neural Information Processing Systems 34 (NeurIPS'21)*, 2021.

Ryan McKenna, Daniel Sheldon, and Gerome Miklau. Graphical-model based estimation and inference for differential privacy. In *Proceedings of the 36th International Conference on Machine Learning (ICML'19)*, 2019.

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

Marcel Neunhoeffer, Steven Wu, and Cynthia Dwork. Private post-GAN boosting. In *9th International Conference on Learning Representations (ICLR'19)*, 2021.

Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the 39th Annual ACM Symposium on the Theory of Computing*, STOC '07, pp. 75–84, New York, NY, USA, 2007. ACM.

Nicolas Papernot, Martín Abadi, Úlfar Erlingsson, Ian J. Goodfellow, and Kunal Talwar. Semi-supervised knowledge transfer for deep learning from private training data. In *5th International Conference on Learning Representations (ICLR'2017)*, 2017.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32 (NeurIPS'19)*, 2019.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *4th International Conference on Learning Representations (ICLR'16)*, 2016.

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

Yuchao Tao, Ryan McKenna, Michael Hay, Ashwin Machanavajjhala, and Gerome Miklau. Benchmarking differentially private synthetic data generation algorithms. *CoRR*, abs/2112.09238, 2021.

Reihaneh Torkzadehmahani, Peter Kairouz, and Benedict Paten. DP-CGAN: Differentially private synthetic data and label generation. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops, (CVPR Workshops'19)*, 2019.

Florian Tramèr and Dan Boneh. Differentially private learning needs better features (or much more data). In *9th International Conference on Learning Representations (ICLR'21)*, 2021.

Margarita Vinaroz, Mohammad-Amin Charusaie, Frederik Harder, Kamil Adamczewski, and Mi Jung Park. Hermite polynomial features for private data generation. In *Proceedings of the 39th International Conference on Machine Learning (ICML'22)*, 2022.

Chris Waites and Rachel Cummings. Differentially private normalizing flows for privacy-preserving density estimation. *CoRR*, abs/2103.14068, 2021.

Boxin Wang, Fan Wu, Yunhui Long, Luka Rimanic, Ce Zhang, and Bo Li. DataLens: Scalable privacy preserving training via gradient compression and aggregation. In *CCS'21: 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021.

Yu-Xiang Wang, Borja Balle, and Shiva Prasad Kasiviswanathan. Subsampled Rényi differential privacy and analytical moments accountant. In *22nd International Conference on Artificial Intelligence and Statistics (AISTATS'19)*, 2019.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. *CoRR*, abs/1708.07747, 2017.

Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially private generative adversarial network. *CoRR*, abs/1802.06739, 2018.

Ashkan Yousefpour, Igor Shilov, Alexandre Sablayrolles, Davide Testuggine, Karthik Prasad, Mani Malek, John Nguyen, Sayan Gosh, Akash Bharadwaj, Jessica Zhao, Graham Cormode, and Ilya Mironov. Opacus: User-friendly differential privacy library in PyTorch. *CoRR*, abs/2109.12298, 2021.

Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A. Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, Sergey Yekhanin, and Huishuai Zhang. Differentially private fine-tuning of language models. In *10th International Conference on Learning Representations (ICLR'22)*, 2022.

Jun Zhang, Graham Cormode, Cecilia M. Procopiuc, Divesh Srivastava, and Xiaokui Xiao. PrivBayes: Private data release via bayesian networks. *ACM Trans. Database Syst.*, 42(4), 2017.

A Generated samples

We provide a few non-cherrypicked samples for MNIST and FashionMNIST at $\varepsilon = 10$ and $\varepsilon = 1$, as well as 32×32 CelebA-Gender at $\varepsilon = 10$.

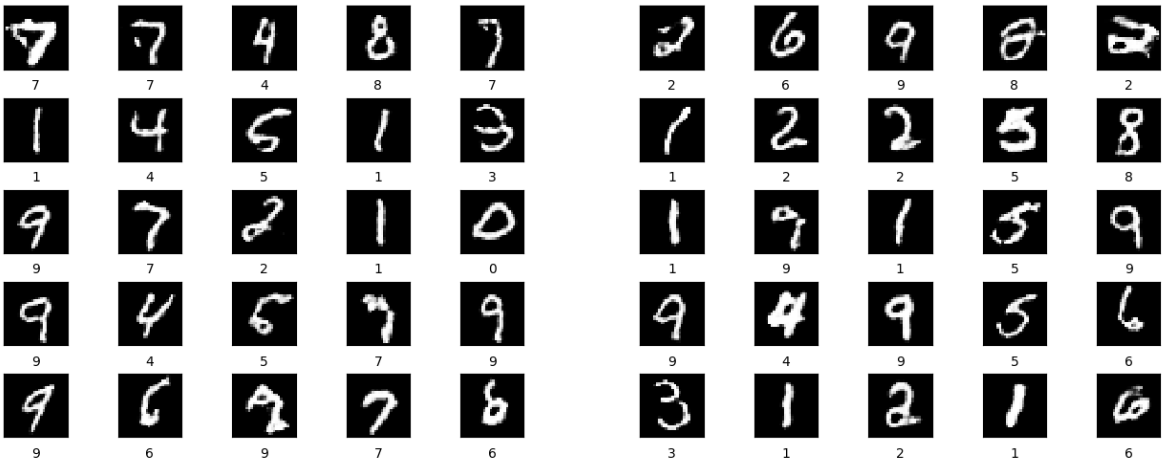


Figure 9: Some non-cherrypicked MNIST samples from our method, $\varepsilon = 10$.

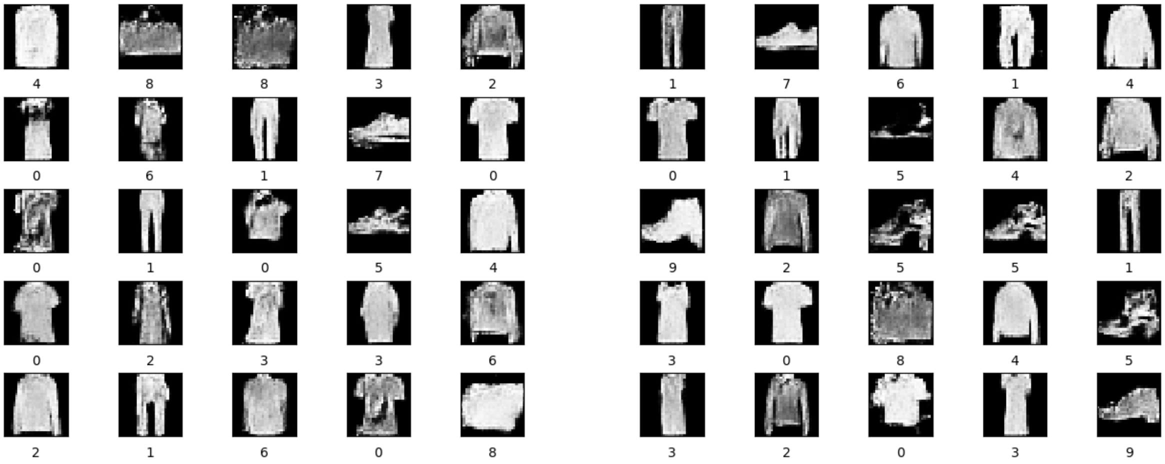


Figure 10: Some non-cherrypicked FashionMNIST samples from our method, $\varepsilon = 10$.

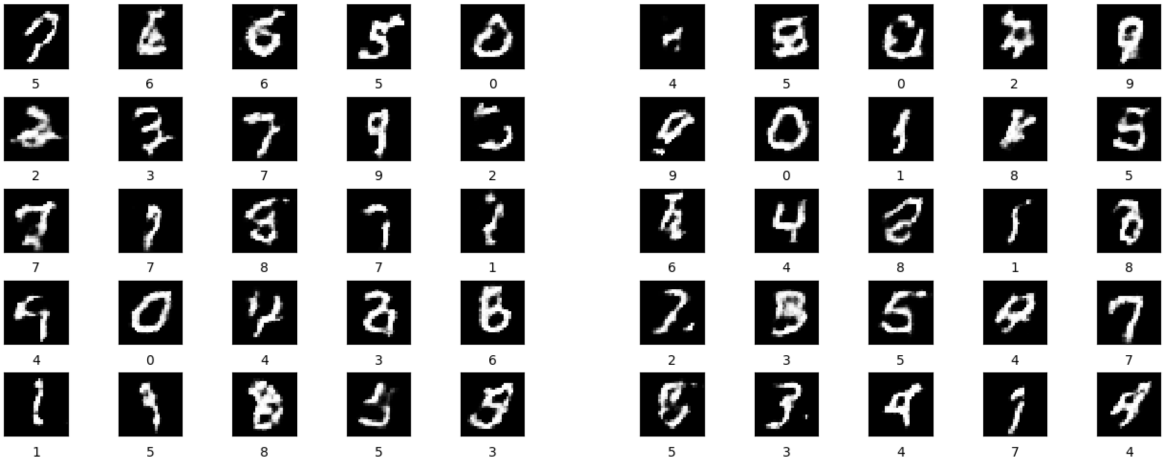


Figure 11: Some non-cherrypicked MNIST samples from our method, $\varepsilon = 1$.

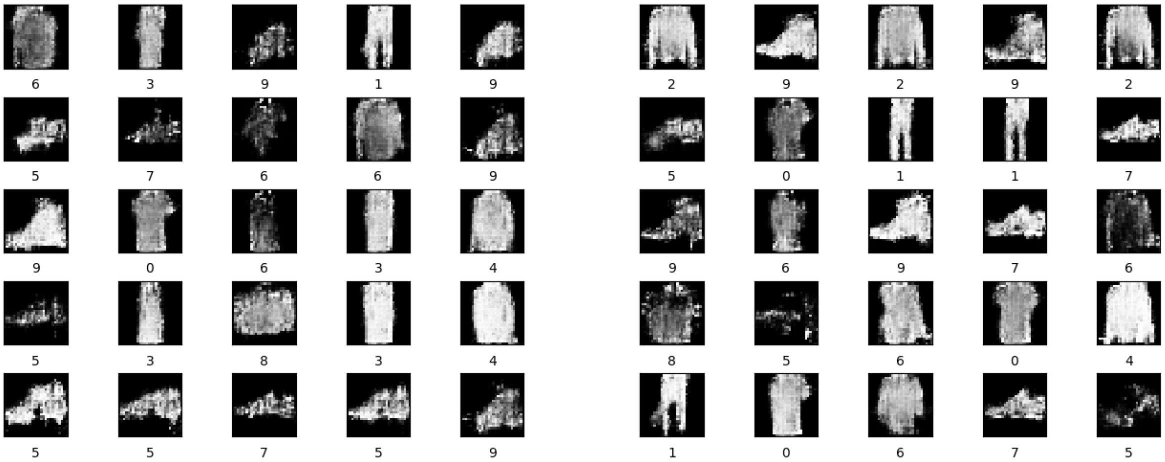


Figure 12: Some non-cherrypicked FashionMNIST samples from our method, $\varepsilon = 1$.

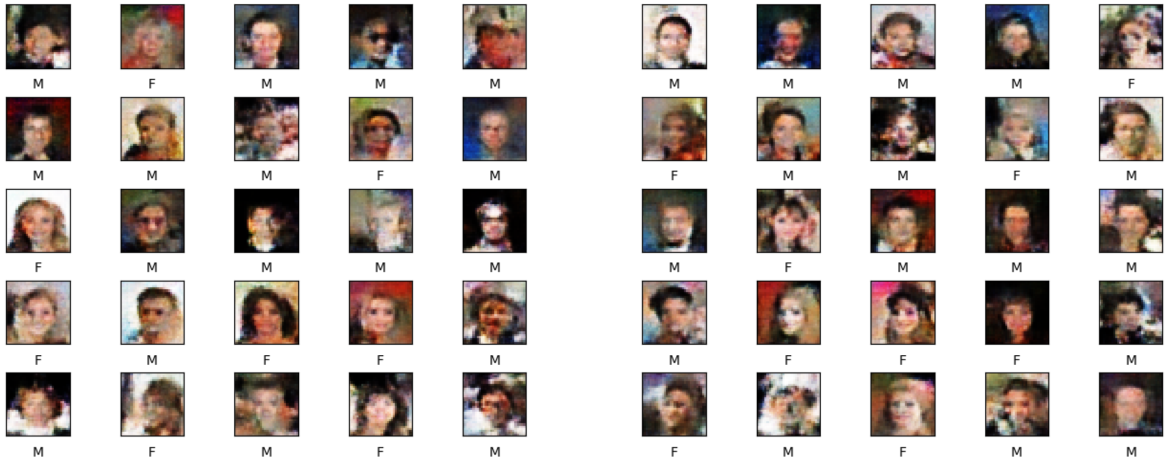


Figure 13: Some non-cherrypicked CelebA samples from our method, $\varepsilon = 10$.

B Implementation details

B.1 MNIST and FashionMNIST training recipe

For MNIST and FashionMNIST, we begin from an open source PyTorch implementation of DCGAN (Radford et al., 2016) (available at this link) that performs well non-privately, and copy their training recipe. This includes: batch size $B = 128$, the Adam optimizer (Kingma & Ba, 2015) with parameters ($\alpha = 0.0002, \beta_1 = 0.5, \beta_2 = 0.999$) for both $\mathcal{G}$ and $\mathcal{D}$, the non-saturating GAN loss (Goodfellow et al., 2014), and a 5-layer fully convolutional architecture with width parameter $d = 128$.

To adapt it to our purposes, we make three architectural modifications: in both $\mathcal{G}$ and $\mathcal{D}$ we (1) remove all BatchNorm layers (which are not compatible with DPSGD); (2) add label embedding layers to enable labelled generation; and (3) adjust convolutional/transpose convolutional stride lengths and kernel sizes as well as remove the last layer, in order to process $1 \times 28 \times 28$ images without having to resize. Finally, we remove their custom weight initialization, opting for PyTorch defaults.

Our baseline non-private GANs are trained for 45K steps. We train our non-private GANs with poisson sampling as well: for each step of discriminator training, we sample real examples by including each element of our dataset independently with probability B/n, where n is the size of our dataset. We then add B fake examples sampled from $\mathcal{G}$ to form our fake/real combined batch.

Clipping fake sample gradients. When training the discriminator privately with DPSGD, we draw B fake examples and compute clipped per-example gradients on the entire combined batch of real and fake examples (see Algorithm 1). This is the approach taken in the prior work of Torkzadehmahani et al. (2019). We remark that this is purely a design choice – it is not necessary to clip the gradients of the fake samples, nor to process them together in the same batch. So long as we preserve the sensitivity of gradient queries *with respect to the real data*, the same amount of noise will suffice for privacy.

B.2 Large batch size hyperparameter search

We scale up batch sizes, considering $B \in \{64, 128, 512, 2048\}$, and search for the optimal noise scale σ and $n_{\mathcal{D}}$. For $B = 128$ targeting $\varepsilon = 10$, we search over three noise scales, $\Sigma^{\varepsilon=10}_{B=128} = \{0.6, 1.0, 1.4\}$. We choose candidate noise scales for other batch sizes as follows: when considering a batch size $128n$, we search over $\Sigma^{\varepsilon 10}_{128n} := \{\sqrt{n} \cdot \sigma : \sigma \in \Sigma^{\varepsilon=10}_{B=128}\}$. We also target the high privacy ($\varepsilon = 1$) regime. For $\varepsilon = 1$, we multiply all noise scales by 5, $\Sigma^{\varepsilon=1}_{B} = \{5\sigma : \sigma \in \Sigma^{\varepsilon=10}_{B}\}$. For each setting of (B, σ), we search over a grid of $n_{\mathcal{D}} \in \{1, 2, 5, 10, 20, 50, 100, 200, 500\}$. Due to compute limitations, we omit some values that we are confident will fail (e.g., trying $n_{\mathcal{D}} = 1$ when mode collapse occurs for $n_{\mathcal{D}} = 5$).

C Additional discussion

GANhacks. Guidance in the non-private setting (tip 14 of Chintala et al. (2016)) prescribes to train the discriminator for more steps in the presence of noise (a regularization approach used in non-private GANs). This is the case for DP, and is our core strategy that yields the most significant gains in utility. We were not aware of this tip when we discovered this phenomenon, but it serves as validation of our finding. While Chintala et al. (2016) provides little elaboration, looking at further explorations of this principle in the non-private setting may offer guidance for improving DPGANs.