Find your private dataset inside StyleGAN: leveraging public GANs for in-domain private image synthesis

Alex Bie

December 21, 2020

Abstract

Being able to generate a synthetic version of a dataset without compromising the privacy of contributing individuals is desirable, facillitating "plug-in privacy protection" that can be inserted right before the first step of an existing machine learning pipeline.

To obtain provable privacy guarantees via *differential privacy*, current approaches for images involve training a generative model on the private dataset with noisy gradients. These methods do not take advantage of the existence of public datasets arising from similar generative processes.

We present a two-step method that leverages public data for private data synthesis. First, we find latent vectors in the latent space of a public-data-trained GAN generator that reconstruct our private images. Then we fit a simple parametric distribution to these latent vectors while respecting differential privacy. The parameters of the GAN generator and private-data-adapted latent distribution are safe to release, and can be used to privately generate images.

We share preliminary, exploratory findings that suggest our method can be used to produce images that are both realistic (can be directly fed into an established image processing pipeline) as well as representative of the semantic characteristics of the private dataset.

1 Introduction

Differentially private data synthesis – the task of generating synthetic data that can substitute a real dataset, while providing differential privacy guarantees for contributing users – is a privacy solution that carries several advantages over the traditional curator/analyst online-querying paradigm:

1. **Reusability**: synthetic data can be reused and repurposed by numerous analyst. It does not need to be "thrown away" after some fixed number of interactions.

2. **Ease of use**: synthetic data directly integrates with existing pipelines and data from other sources. Data analysts can also directly look at the data, allowing them to develop intuitions and discover bugs.

However, a hardness result suggests that the task may be impossible in the worst case: for binary vector datasets, there is no poly-time algorithm that can approximately preserve even two-way marginal counts[1] [UV20]. Despite this, significant efforts have been made to find heuristics that generate data with good utility in practice for various downstream tasks. In the deployment of differential privacy for the 2020 US Census – where the advantages described above have been deemed crucial for the application – 90TB of random bits will be used to generate a set of synthetic records released for use in all future downstream applications [GL20].

[1] Under the one-way function existence conjecture that underpins cryptography.

Natural image synthesis We look toward the problem of natural image synthesis, which has seen successes in recent years through GPU-accelerated training of deep neural networks on large image datasets. The Generative Adversarial Network (GAN) framework [GPAM+14] lies at the core of the vast majority of approaches capable of generating high-quality and realistic images [KLA19, BDS18].

Several systems have been proposed to tackle the problem of image synthesis under differential privacy, most of which focus on privatizing GAN training [XLW+18, COF20]. These approaches however, have only demonstrated moderate success on small monochrome image datasets (MNIST and Fashion-MNIST) in the low privacy regime ($\varepsilon \approx 10$). GANs suffer from training instability, a problem that is exacerbated by the introduction of gradient noise for privacy. We note that these methods aim to model arbitrary image distributions from scratch. However, natural images arise from similar physical processes, obeying the same set of rules in terms of perspective, lighting, texture, etc. In the case that "in-domain" images are publically available, it makes sense to learn shared characteristics from public data, reserving our privacy budget for learning dataset-specific characterstics.

1.1 Our approach[2]

We decompose privacy-preserving image generation into two subtasks, where "semantic" characteristics are captured from private data, and the translation of these characteristics into realistic images is left to a generative model trained on public data. Drawing an analogy to computer graphics: we leave the learning of the "renderer" – with information on perspective, textures, and geometric building blocks common to both datasets – to public data, and spend our privacy budget on capturing the high-level "scenes" of our private dataset.

We propose the following two-step approach. Given a private dataset of images $X_{\mathrm{p}} = \{x_1^{\mathrm{p}}, ..., x_n^{\mathrm{p}}\}$ and a GAN generator $G : \mathcal{W} \to \mathcal{X}$ trained on a set of public images from $\mathcal{X}$:

1. We find latent vectors $W_{\mathrm{p}} = \{w_1^{\mathrm{p}}, ..., w_n^{\mathrm{p}}\}$ with each w_j^{p} corresponding to x_j^{p}.
2. We fit a parametric distribution (e.g. multivariate Gaussian) to W_{p} in a differentially private manner, obtaining sanitized parameters $\widetilde{\theta}_{W_{\mathrm{p}}}$.

Then, with access to the public GAN generator G and sanitized parameters $\widetilde{\theta}_{W_{\mathrm{p}}}$, we can sample new images. We draw $\widetilde{w} \sim \widetilde{\theta}_{W_{\mathrm{p}}}$, and run it through G to obtain our sample $\widetilde{x} = G(\widetilde{w})$.

We remark that this approach is analogous to how differentially private learning is done for natural image classification, albeit inverted. Reasonable CIFAR-10 classification results [ACG+16] were obtained by transferring pretrained filters from (assumed public) CIFAR-100 classification and reserving the privacy budget to train only the fully-connected layers that follow, which operate on feature space.

Organization Sections 2 & 3 review details about GANs (2) and differential privacy (3). Section 4 describes our method, and Sections 5 & 6 offer exploratory experimental findings (5) and directions for fleshing out this project (6).

2 Latent space of GANs

The goal of a GAN is to learn how to sample from some unknown distribution $\mathcal{D}$ over $\mathcal{X}$, given samples from it. GANs model $\mathcal{D}$ with the following generative process, from which arises $\mathcal{D}_G$:

- $w \sim p(w)$, where $p(w)$ is an elementary distribution that can easily be sampled from (e.g. standard Gaussian);
- $x = G(w)$, where $G : \mathcal{W} \to \mathcal{X}$ is a deep neural network.

[2]From my own Google search, I couldn't find any papers using this "find-latents/fit-latents" approach, but I think the idea is fairly low-hanging + there are lots and lots of deep learning papers so it's possible that I'm missing a reference.

We can use samples from $\mathcal{D}_G$ and $\mathcal{D}$ to approximate some metric over distributions, and if everything is differentiable, we can optimize G with gradient descent to minimize that metric.

Well-trained GANs tend to learn mappings G such that linear paths P in our latent space $\mathcal{W}$ have ranges $G(P)$ that vary "linearly" in some semantic characteristic. For example, we can find a direction in $\mathcal{W}$ that continuously rotates the camera angle while keeping the scene mostly the same [HHLP20]. Fitting a smooth parametric distribution to latent vectors in the latent space of a well-trained GAN could feasibly lead to realistic samples sharing semantic characteristics. On the other hand, this is certainly not true in image space – a Gaussian is not expressive enough to the distribution of natural images to produce realistic samples.

2.1 StyleGAN

We use StyleGAN2 [KLA+20] as our public-data-trained GAN, due to its ability to: generate high-resolution realistic images; interpolate well between samples; and disentangle semantic characteristics. The authors make available various pretrained models for different datasets[3]. With $d = 512, \mathcal{W} = \mathcal{Z} = \mathbb{R}^d$, and $\mathcal{W}^+ = \mathbb{R}^{16\times d}$ the generative process of StyleGAN is as follows:

- $z \sim N(0, I_d)$;
- $w = M(x)$, where $M : \mathcal{Z} \to \mathcal{W}$ is called the *mapping network*, an 8-layer MLP;
- $w^+ = A(w)$, where $A : \mathcal{W} \to \mathcal{W}^+$ is an affine transformation.
- For $l = 1...8$, x_l is an intermediate $2^{l+1} \times 2^{l+1}$ resolution image obtained via $x_l = G_l(x_{l-1}, w_l^+)$ where each G_l refines the previous lower-resolution generated image using "style" information $w_l^+ = w^+[2l - 1 : 2l] \in 2^{2\times d}$. The 4×4 image x_0 is a constant, and $x = x_8$ is the final output image. The mapping $G : \mathcal{W} \to \mathcal{X}$ is the composition of these G_l and A.

There are 3 latent spaces we can use: $\mathcal{Z}, \mathcal{W}$, and $\mathcal{W}^+$. Previous work has shown that interpolations in $\mathcal{Z}$ are usually poor [KLA19]. We avoid using $\mathcal{W}^+$ since it is too high dimensional: searching is more expensive in $\mathcal{W}^+$, and since in many cases we have $n < 8192$, we run into problems when estimating estimating covariance matrices. Hence, we will use $\mathcal{W}$.

3 Differential Privacy

Roughly speaking, differential privacy guarantees that the output of an algorithm does not change significantly as a result of the arbitrary change of any single entry in the input database.

Definition 1. ((ε, δ)-DP [DMNS06]). Let X be the space of data entries. Considering n-entry databases as elements in X^n, we say a randomized algorithm $\mathcal{M} : X^n \to O$ is (ε, δ)**-differentially private** ((ϵ, δ)**-DP**), if for every pair of input databases $D_1, D_2 \in X^n$ differing in at most one entry, we have

$$\mathbb{P}[\mathcal{M}(D_1) \in S] \leq e^{\varepsilon}\, \mathbb{P}[\mathcal{M}(D_2) \in S] + \delta \qquad \text{for all measurable } S \subseteq O$$

Post-processing and composition properties of differential privacy make it easy to construct private algorithms out of simple building blocks.

Theorem 1. (Properties of (ϵ, δ)-DP [DR14]). Let $\mathcal{M} : X^n \to O$ be (ϵ, δ)-DP.

1. (Post-processing). If $\mathcal{A} : O \to O'$ is any deterministic or randomized mapping then $\mathcal{A} \circ \mathcal{M} : X^n \to O'$ is also (ϵ, δ)-differentially private.
2. (Naive composition). If $\mathcal{M}' : O \times X^n \to O'$ has that $M'(o, \cdot)$ is (ε', δ')-DP for any $o \in O$, then $\mathcal{M}' \circ (\mathcal{M}, I_{X^n}) : X^n \to O'$ is $(\varepsilon + \varepsilon', \delta + \delta')$-DP.

[3] Code: https://github.com/NVlabs/stylegan2

The Gaussian mechanism is our choice of building block that realizes (ε, δ)-DP. To privatize the computation of some function of our private data $f : X^n \to \mathbb{R}^d$, the Gaussian mechanism adds specifically calibrated independent Gaussian noise to all d dimensions of the output.

Theorem 2. (**Gaussian mechanism** [DR14]). Let $f : X^n \to \mathbb{R}^d$, and let

$$\Delta_2(f) = \max_{D,D':\|D-D'\|_1 \leq 1} \|f(D) - f(D')\|_2$$

denote the ℓ_2**-sensitivity** of f. For $0 < \varepsilon < 1$, $c \geq 2 \ln \frac{1.25}{\delta}$, we have $\mathcal{M}(D) = f(D) + \mathcal{N}(0, \frac{c\Delta_2(f)}{\varepsilon} I_d)$ is (ε, δ)-DP.

Since most of the experiments we conduct are in the low privacy regime ($\varepsilon > 1$), we resort instead to the analytic Gaussian mechanism[4] [BW18] to compute a c satisfying (ε, δ)-DP for our target ε.

4 Method

We provide more detail on the two-step approach of: first finding latent vectors that reconstruct our private images; then privately fitting a distribution on those latents (as described in Section 1.1).

4.1 Finding latent vectors

The approach to find the StyleGAN latent vectors corresponding to our images comes from previous work on the problem [AQW20, KLA+20]. We consider a few variations that can be described under the following steps of:

1. **Initialization.** For an image x, we initialize with a latent vector $w_0 \in \mathcal{W}$. Our choice of w_0 can be:
   (a) $w_{\text{avg}} = \frac{1}{m} \sum_{i=1}^m w_i$ the average of m samples $\{w_i\}_{i=1}^m$ obtained by sampling $z_i \sim \mathcal{N}(0, I_d)$ and running them through StyleGAN's mapping network.
   (b) $w_{\text{FF}} = F_\phi(x)$, the result of mapping x through an inversion network $F_\phi : \mathcal{X} \to \mathcal{W}$ obtained by training on pairs $\{(w_i, x_i)\}_{i=1}^m$ obtained by sampling from the GAN. F_ϕ is trained with minibatch SGD to optimize an MSE regression loss

$$\phi^* = \min_\phi \frac{1}{m} \sum_{i=1}^m \|w_i - F_\phi(x_i)\| \tag{1}$$

2. **Optimization.** Given $X_{\text{p}} = \{x_1^{\text{p}}, ..., x_n^{\text{p}}\}$, for each x_j^{p} we aim to solve the optimization problem

$$w_j^{\text{p}} = \min_w \text{d}(G(w), x_j^{\text{p}}) + \lambda \|w\|_2^2 \tag{2}$$

   where $\text{d}(\cdot, \cdot)$ is a differentiable image similarity metric and λ is hyperparameter that controls how much we penalize solutions with large norm (which we motivate in Section x). We optimize the objective through gradient descent on w, starting from the initialization w_0. We use the LPIPS metric [ZIE+18] computed using neural-network-extracted features, which is much better than the blurry results we get from image-space MSE.

[4] Code: https://github.com/BorjaBalle/analytic-gaussian-mechanism

4.2 Fitting the distribution of latent vectors

We fit a d-dimensional multivariate Gaussian to W_{p}, where d is the dimensionality of $\mathcal{W}$. The (non-private) MLE parameter estimates are

$$\widehat{\mu}_{W_{\text{p}}} = \frac{1}{n}\sum_{j=1}^{n} w_j^{\text{p}} \quad \text{and} \quad \widehat{\Sigma}_{W_{\text{p}}} = \frac{1}{n}\sum_{j=1}^{n}(w_j^{\text{p}} - \widehat{\mu}_{W_{\text{p}}})(w_j^{\text{p}} - \widehat{\mu}_{W_{\text{p}}})^T \tag{3}$$

To enforce differential privacy, we first clip the w_j^{p} to $\|\cdot\|_2 \leq M$ to form $\overline{W_{\text{p}}} = \{\overline{w_1^{\text{p}}}, ..., \overline{w_n^{\text{p}}}\}$ where M is chosen to keep most points unchanged[5], and then apply the Gaussian mechanism to obtain $\widetilde{\mu}_{W_{\text{p}}}$ and $\widetilde{\Sigma}_{W_{\text{p}}}$. For the ℓ_2-sensitivity of the mean, we get $\Delta_2(\widehat{\mu}) = 2M/n$. For the covariance, we apply Gaussian mechanism to computation of $\frac{1}{n}\sum_{j=1}^{n} \overline{w_j^{\text{p}}}(\overline{w_j^{\text{p}}})^T$, which has ℓ_2-sensitivity of $2M^2/n$ by a matrix CS inequality. We arrive at DP estimates

$$\widetilde{\mu}_{W_{\text{p}}} = \frac{1}{n}\sum_{j=1}^{n} \overline{w_j^{\text{p}}} + \mathcal{N}(0, \frac{2Mc}{n\varepsilon_\mu} I_d) \quad \text{and} \quad \widetilde{\Sigma}_{W_{\text{p}}} = \frac{1}{n}\sum_{j=1}^{n} \overline{w_j^{\text{p}}}(\overline{w_j^{\text{p}}})^T + \mathcal{N}(0, \frac{2M^2c}{n\varepsilon_\Sigma} I_{d\times d}) - \widetilde{\mu}_{W_{\text{p}}}(\widetilde{\mu}_{W_{\text{p}}})^T \tag{4}$$

where the $c(\varepsilon, \delta)$ is obtained from code for the analytical Gaussian mechanism [BW18].

We apply some post-processing steps to $\widetilde{\Sigma}_{W_{\text{p}}}$. Since the covariance matrix should be symmetric, we take the $\frac{d(d+1)}{2}$ upper triangular entries and reflect to get the lower half. Furthermore, to sample with $\widetilde{\Sigma}_{W_{\text{p}}}$, we require it to be positive definite, which is not always the case. We apply a hacky fix by replacing negative eigenvalues in its $\widetilde{\Sigma}_{W_{\text{p}}}$'s eigendecomposition (its now symmetric) with a small positive value. In scenarios where the noise added is relatively small, negative eigenvalues usually have very small magnitude.

5 Experimental findings

5.1 Datasets

Public data For our public-data-trained GAN, we use the "StyleGAN2-Car-F" checkpoint made available by the authors of [KLA+20], which is trained on ∼900k, 384 × 512 car images from the LSUN dataset [YZS+15].

Private data For our private datasets, we use the Stanford Cars [KSDFF13] and VMMRdb [TFN17] datasets. **Stanford Cars** consists of ∼9k car images and bounding boxes of various models. We use the bounding boxes to crop and scale images to match StyleGAN's dimensions. **VMMRdb** consists of ∼300k car images with make/model/year labels. For preprocessing, we predict car bounding boxes using MMDetection, an off the shelf object recognition library [CWP+19], to scale and crop the images.

5.2 Finding latent vectors

Inversion network We use the EfficientNet B1 architecture [TL19], pretrained on ImageNet, as the base for our image→latent inversion map F_ϕ. We replace the final prediction layer with a fully-connected layer to output $w \in \mathbb{R}^{512}$, and train the whole network following Equation 1. We generate new batches of training data $\{(x_i, w_i)\}_{i=1}^{B}$ with StyleGAN on the fly and periodically validate. We stop training when validation accuracy does not improve for k consecutive runs.

[5]To ensure DP, we need to pick the clipping norm M beforehand which could lead to excess noise. For our experiments, we cheat and pick M to be the 99th norm percentile of W_{p}. This however, can be made private by allocating a small portion of the budget to first building a DP-histogram to compute this quantity (as seen in Problem Set 1). We will make this fix in later versions

(a) StyleGAN generated images

(b) Stanford Cars

(c) VMMRdb

Figure 1: Preprocessed real images for our various datasets.

	$\|\widehat{\mu}_{W_\mathrm{p}} - \widetilde{\mu}_{W_\mathrm{p}}\|_2$	$\|\widehat{\sigma}_{W_\mathrm{p}} - \widetilde{\Sigma}_{W_\mathrm{p}}\|_2$
FF	0.354	6.43
OPT	0.826	33.8

Table 1: DP-fit error for Stanford Cars.

	$\|\widehat{\mu}_{W_\mathrm{p}} - \widetilde{\mu}_{W_\mathrm{p}}\|_2$	$\|\widehat{\sigma}_{W_\mathrm{p}} - \widetilde{\Sigma}_{W_\mathrm{p}}\|_2$
FF	0.0388	0.793
$\mathbf{OPT}_A$	0.0364	1.11
$\mathbf{OPT}_B$	0.0312	0.758

Table 2: DP-fit error for VMMRdb. Note that n is much larger so we can add less noise.

5.2.1 Regression vs. Optimization

Based on the setup in Section 4.1, we can describe various approaches for finding latent vectors, with each expressed as a 4-tuple $(w_0, \text{optimization steps}, \text{learning rate}, \lambda)$. We first compare results for **FF** $= (w_{\mathrm{FF}}, 0, -, -)$, which is just the output of F_ϕ, and **OPT** $= (w_{\mathrm{FF}}, 500, 0.1, 0)$, which is initializing at w_{FF} and taking 500 steps according to Equation 2. Qualitative results are pictured in Figure 2. The DP parameters used for **DP-fit** are $(\varepsilon_\mu, \delta_\mu) = (1, 5 \cdot 10^{-6}), (\varepsilon_\Sigma, \delta_\Sigma) = (10, 5 \cdot 10^{-6})$.
When comparing latent reconstructions, **OPT** is more successful at recovering the original image (see real images in Figure 1b), but samples from the Gaussian fit on **FF** latents are much better. **Over-optimized latents lead to poorer (non-private) fits of the private data**. Looking at the distribution of distances from w_{avg} for different groups of latents (Figure 3), we hypothesize that this is because most latents obtained from **OPT** are guaranteed to be in very low density regions of $\mathcal{W}$, so sampling around them will not yield realistic images. Table 1 compares the parameter error induced by the noise added for DP. **Over-optimized latents have large norms, requiring more noise and resulting in greater estimation error.**

5.2.2 Norm penalty

These findings motivate us to experiment with adding a norm penalty λ in our optimization objective (Equation 2). A chart of optimization trajectories for different configurations can be found in Appendix A. With the norm penalty, we are able to get good reconstructions while keeping $\|w^{\mathrm{p}}\|_2$ small. We pick

(a) **FF** (b) **OPT**

Figure 2: Two rows for each category. Comparison of two approaches for finding latent vectors for our private images, and random samples drawn from non-private and private Gaussian fits on those latent vectors.

two configurations $\mathbf{OPT}_A$ and $\mathbf{OPT}_B$ that show good tradeoffs between reconstruction quality and norm size, and use them to apply our method to VMMRdb. The DP parameters we use are $(\varepsilon_\mu, \delta_\mu) = (1, 5 \cdot 10^{-7}), (\varepsilon_\Sigma, \delta_\Sigma) = (10, 5 \cdot 10^{-7})$. Qualitative results and distance distributions can be found in Appendix B. Table 2 demonstrates the advantage of small norms on parameter estimation error. Furthermore, although the norm penalty does not force w into high density regions, it appears to be a useful proxy, and helps us get better non-private fits.

5.3 FID

For our results on VMMRdb, we follow [HRU+17] to compute FID, a measure of distance between sets of images. FID scores against the target distribution are commonly used to evaluate GANs. Results are in Table 3. Although it is strange that some of our private fits have better FID than our non-private fits, we do observe that our method is able to bias GAN generation towards characteristics observed in our private dataset.

	FF			$\mathbf{OPT}_A$			$\mathbf{OPT}_A$			**GAN**	
	LR	F	DP-F	LR	F	DP-F	LR	F	DP-F	A	B
FID	10.87	13.23	13.10	7.68	13.06	11.59	8.06	12.35	12.49	38.94	37.37

Table 3: FID scores against real VMMRdb images. For each of the latent finding configurations, we report FID against 10k real VMMRdb images with 10k of either: reconstructed latents (LR), samples from the non-private fit (F), and samples from the private fit (DP-F). We also compare to the baseline of random GAN samples that do not take into account the private data. Config A is sampling directly from the GAN, and Config B is sampling with the "truncation trick" ($\psi = 0.5$) [AQW20], a technique used to get better samples from GANs.

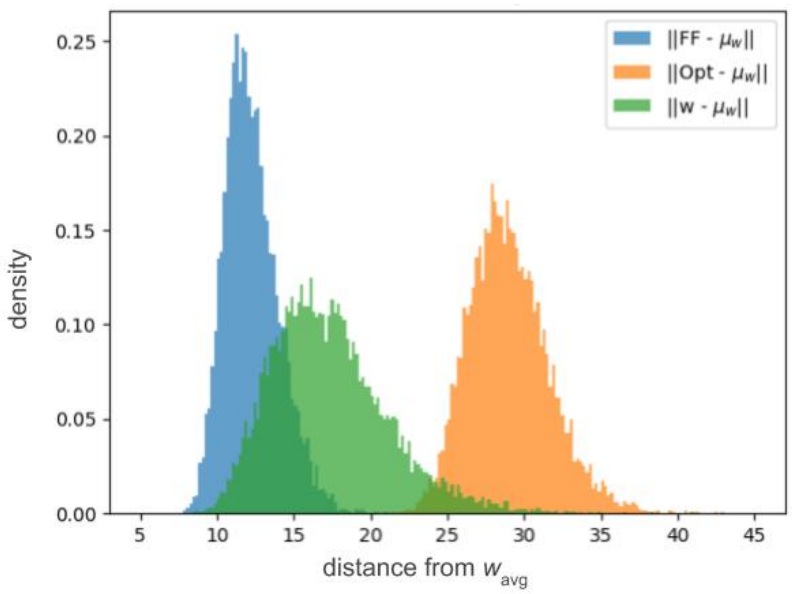

Figure 3: Distribution of distances from w_{avg} for latent reconstructions of Stanford Cars.

5.4 Fitting Aston Martins

To get a qualitative idea of how much our method is adapting to the characteristics of the private dataset, we gather a subpopulation of 157 Aston Martin images from Stanford Cars and apply our method. We choose this car because it has a fairly unique look (StyleGAN won't randomly generate it often). In this case n is too small for the DP version to yield something reasonable, but we show samples from the non-private fit with **FF**. In Appendix C, we share **OPT** results, which again yields good reconstructions but poor samples from the resulting fit.

Figure 4: **FF** results on Aston Martins. **Top 2 rows:** Real Aston Martins; **Middle 2 rows:** Latent reconstruction; **Bottom 2 rows:** Samples from fit.

6 Conclusion

We propose a method to privately generate synthetic natural images when there exists public datasets in the same domain as our private one. We present some preliminary, exploratory experiments that suggest the method is capable of generating realistic images while capturing semantic characteristics of our private dataset.

6.1 Future work

There are two immediate areas of this project that we believe should be fleshed out:

- FID scores are a useful metric for evaluating generative models, but we need to do further validation on downstream tasks to ascertain the utility of generated images. We planned to include an experiment to see if we could learn sub-population classification from synthetic data (privately fit generators on each car model, use these labels to train a classifer and test on real data). Privacy costs can actually be reasonable here due to parallel composition. Along these lines, we can also evaluate utility by assuming there is a small amount of public labelled data, and use our private generative model for self-supervised learning.
- Fixing the hacky way we privately learn the latent Gaussians: the choice of clipping bound; as well as something better than the trick we use to correct non-positive-definite covariance matrices.

Two areas that deserve further exploration:

- Testing how similar our private data needs to be our public data for this approach to work.
- A more principled, end-to-end way to learn our latent distribution. For example, we could learn μ and Σ by taking samples $z \sim \mathcal{N}(0, 1)$ and privately minimizing an approximation of $\mathrm{d}(X_\mathrm{p}, G(\Sigma z + \mu))$ where $\mathrm{d}(\cdot, \cdot)$ is some metric between distributions (e.g. KL, Wasserstein). Then our optimization objective is directly aligned with our goal of learning X_p privately, and does not require intermediate steps like finding latents.

References

[ACG+16] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318, 2016.

[AQW20] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan++: How to edit the embedded images? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8296–8305, 2020.

[BDS18] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2018.

[BW18] Borja Balle and Yu-Xiang Wang. Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. In *International Conference on Machine Learning*, pages 394–403, 2018.

[COF20] 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*, volume 33, 2020.

[CWP+19] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

[DMNS06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[DR14] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Theoretical Computer Science*, 9(3-4):211–407, 2014.

[GL20] Simson L. Garfinkel and Philip Leclerc. Randomness concerns when deploying differential privacy. In *Proceedings of the 19th Workshop on Privacy in the Electronic Society*, WPES'20, page 73–86, New York, NY, USA, 2020. Association for Computing Machinery.

[GPAM+14] 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*, volume 27, pages 2672–2680, 2014.

[HHLP20] Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable gan controls. *arXiv preprint arXiv:2004.02546*, 2020.

[HRU+17] 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*, volume 30, pages 6626–6637, 2017.

[KLA19] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.

[KLA+20] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8110–8119, 2020.

[KSDFF13] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013.

[TFN17] Faezeh Tafazzoli, Hichem Frigui, and Keishin Nishiyama. A large and diverse dataset for improved vehicle make and model recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8, 2017.

[TL19] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114, 2019.

[UV20] Jonathan Ullman and Salil Vadhan. Pcps and the hardness of generating synthetic data. *Journal of Cryptology*, 33(4):2078–2112, 2020.

[XLW+18] Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially private generative adversarial network. *arXiv preprint arXiv:1802.06739*, 2018.

[YZS+15] Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

[ZIE+18] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018.

A Latent optimization trajectories with norm penalty

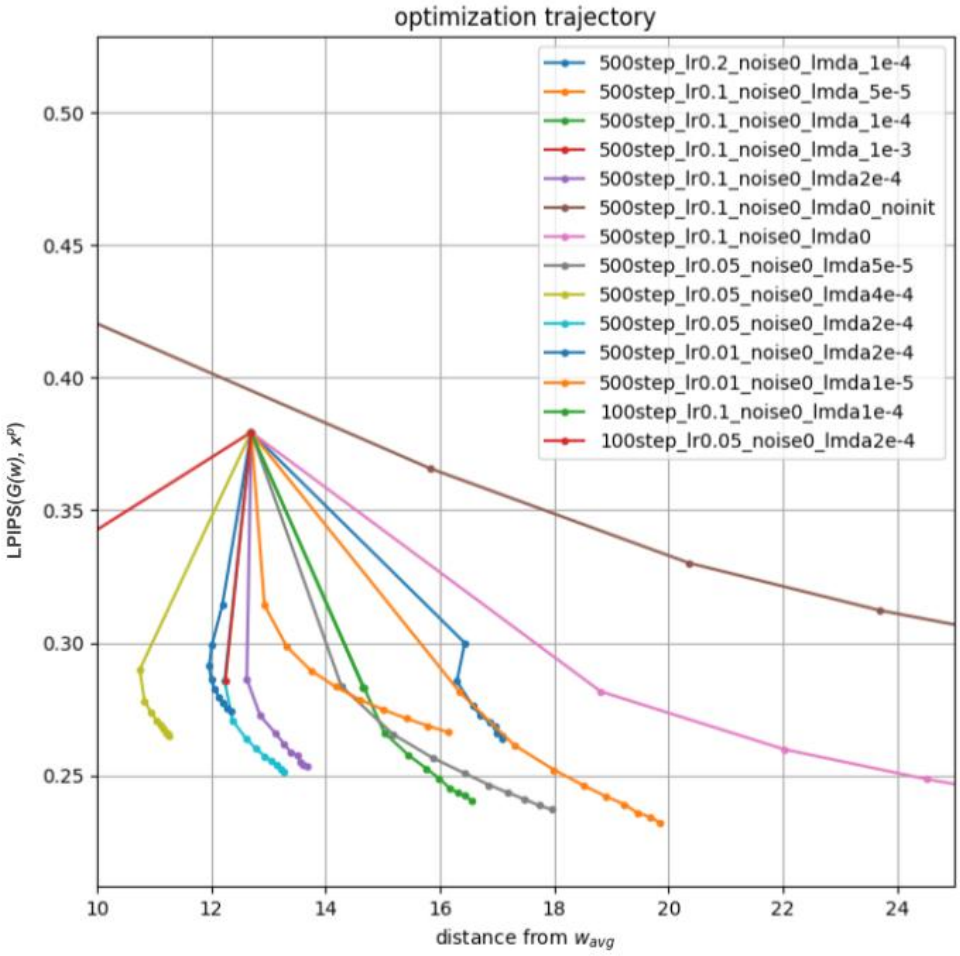

Figure 5: Optimization trajectories for various configurations. Dots represent 10 steps of gradient descent. Results are the average of 500 images. All configurations except brown are initialized at w_{FF}.

B Extra results for VMMRdb

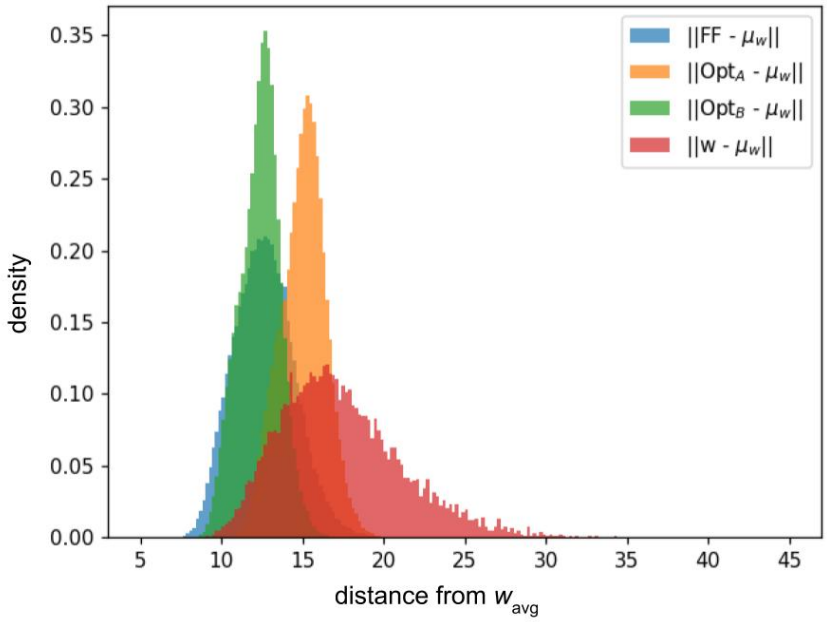

Figure 6: Distribution of distances from w_{avg} for latent reconstructions of VMMRdb.

Figure 7: **FF** results on VMMRdb. **Top 2 rows:** Latent reconstruction; **Middle 2 rows:** Samples from non-private fit; **Bottom 2 rows:** Samples from DP-fit.

Figure 8: **OPT**$_A$ results on VMMRdb. **Top 2 rows:** Latent reconstruction; **Middle 2 rows:** Samples from fit; **Bottom 2 rows:** Samples from DP-fit.

Figure 9: **OPT**$_B$ results on VMMRdb. **Top 2 rows:** Latent reconstruction; **Middle 2 rows:** Samples from fit; **Bottom 2 rows:** Samples from DP-fit.

C OPT results for Aston Martins

Figure 10: **OPT** results on Aston Martins. **Top 2 rows:** Latent reconstruction; **Bottom 2 rows:** Samples from fit.