# Differentially Private Generative Models Through Optimal Transport

**Tianshi Cao** [1,2,3] **Alex Bie** [1,4] **Karsten Kreis** [1] **Sanja Fidler** [1,2,3]
NVIDIA[1] University of Toronto [2] Vector Institute [3] University of Waterloo[4]
{tianshic, abie, kkreis, sfidler}@nvidia.com

## Abstract

Although machine learning models trained on massive data have led to breakthroughs in several areas, their deployment in privacy-sensitive domains remains limited due to restricted access to data. Generative models trained with privacy constraints on private data can sidestep this challenge and provide indirect access to the private data instead. We propose DP-Sinkhorn, a novel optimal transport-based generative method for learning data distributions from private data with differential privacy. Compared to GAN-based methods, DP-Sinkhorn does not rely on adversarial objects, making it easy to train and deploy. Experimentally, despite our method's simplicity we improve upon the state-of-the-art on multiple image modeling benchmarks. We also show differentially private synthesis of informative RGB images, which has not been demonstrated before by differentially private generative models without the use of auxiliary public data.

## 1 Introduction

Differential Privacy (DP) is a rigorous definition of privacy that quantifies the amount of information leaked by a user participating in any data release (1; 2). It has been widely used in ML applications to protect the privacy of data used in training (3). DP learning of generative models has been studied mostly under the Generative Adversarial Networks (GAN) framework (4; 5; 6; 7; 8). While GANs in the non-private setting are successful in synthesizing complex data like high definition images (9; 10), their application in the private setting is more challenging. This is in part because GANs suffer from training instability problems (11; 12), which can be exacerbated when adding noise to the network's gradients during training, a common technique to implement DP. Thus, GANs typically require careful hyperparameter tuning and supervision during training to avoid model collapse. This goes against the principle of privacy, where repeated interactions with data need to be avoided (13).

Optimal Transport (OT) is another method to train generative models. In the optimal transport setting, the problem of learning a generative model is framed as minimizing the optimal transport distance, a type of Wasserstein distance, between the generator-induced distribution and the real data distribution (14; 15). Unfortunately, exactly computing the OT distance is generally expensive. Nevertheless, Wasserstein distance-based objectives are actually widely used to train GANs (16; 17). However, these approaches typically estimate a Wasserstein distance using an adversarially trained discriminator. Hence, training instabilities remain (12).

An alternative to adversarial-based OT estimation is provided by the Sinkhorn divergence (18; 19; 20). The Sinkhorn divergence is an entropy-regularized version of the exact OT distance, for which the optimal transport plan can be computed efficiently via the Sinkhorn algorithm (21). In this paper, we propose DP-Sinkhorn, a novel method to train differentially private generative models using the Sinkhorn divergence as objective. Since the Sinkhorn approach does not intrinsically rely on adversarial components, it avoids any potential training instabilities and removes the need for early stopping. This makes our method easy to train and deploy in practice. As a side, we also develop a

Preprint. Under review.

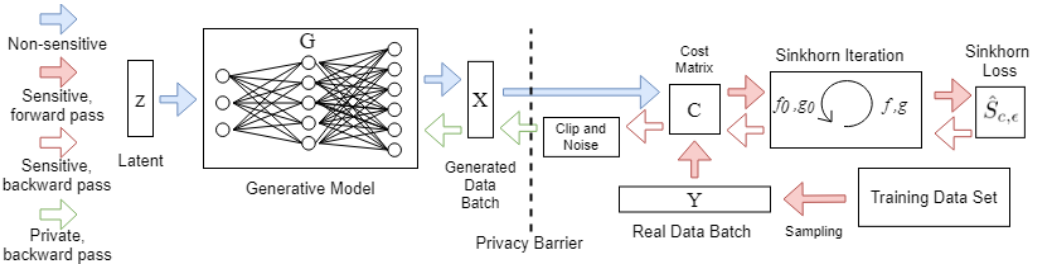


Figure 1: Flow diagram of DP-Sinkhorn for a single training iteration: Sensitive training data is combined with non-sensitive generated data in the cost matrix. Then, the loss is calculated using the Sinkhorn algorithm. In the backward pass, we impose a privacy barrier behind the generator by clipping and adding noise to the gradients at the generated image level, similar to (8).

simple yet effective way to perform conditional generation in the Sinkhorn framework, by forcing the optimal transport plan to couple same-label data closer together. To the best of our knowledge, DP-Sinkhorn is the first fully OT-based approach for differentially private generative learning.

Experimentally, despite its simplicity DP-Sinkhorn achieves state-of-the-art results on image-based classification benchmarks that use data generated under differential privacy for training. We can also generate informative RGB images, which, to the best of our knowledge, has not been demonstrated by any generative models trained with differential privacy and without auxiliary public data.

We make the following contributions: (i) We propose DP-Sinkhorn, a flexible and robust optimal transport-based framework for training differentially private generative models. (ii) We introduce a simple technique to perform label-conditional synthesis in the Sinkhorn framework. (iii) We achieve state-of-the-art performance on widely used image modeling benchmarks. (iv) We present informative RGB images generated under strict differential privacy without the use of public data.

## 2 Differentially Private Sinkhorn with Class Conditioning

For background and standard definitions, please see Appendix B. We propose DP-Sinkhorn, an OT-based method to learn differentially private generative models, which avoids the training instability problems characteristic for GAN-based techniques (algorithms are presented in Appendix C).

**Definition 2.1** *(Empirical Sinkhorn loss) The empirical Sinkhorn loss computed over a batch of $N$ generated examples and $M$ real examples is defined as (19):*

$$\hat{S}_{c,\epsilon}(\mathbf{X}, \mathbf{Y}) = 2C_{XY} \odot P^*_{\epsilon,X,Y} - C_{XX} \odot P^*_{\epsilon,X,X} - C_{YY} \odot P^*_{\epsilon,Y,Y} \tag{1}$$

*where $\odot$ denotes the Frobenius product. For two samples $A \in \mathcal{X}^N$ and $B \in \mathcal{X}^M$, $C_{AB}$ is the cost matrix obtained by applying cost function $c(a, b)$ to elements of $A$ and $B$, and $P^*_{\epsilon,A,B}$ is an approximate optimal transport plan that minimizes Eqn. 4 computed by the Sinkhorn algorithm (21).*

We use the empirical Sinkhorn loss between batches of real ($\mathbf{Y}$) and generated data ($\mathbf{X}$) as objective and use a simple pixel-wise L2-loss for $c$. See Appendix Algorithm 3 for a precise description of our method.

To conditionally generate images given a target class, we inject class information to both the generator and the Sinkhorn loss function during training. For the generator, we supply the class label either by concatenating an embedding of it with the sampled latent code (on DCGAN-based generators), or by using class conditioned batch-norm layers (on BigGAN-based generators). For the loss function, we concatenate a scaled one-hot class embedding to both the generated images and real images. Intuitively, this works by increasing the cost between image pairs of different classes, hence shifting the weight of the transport plan ($P^*_\epsilon$ in Eq. 6 in Appendix B) towards class-matched pairs. Uniformly sampled labels are used for the generated images and the real label is used for real images. Let $l_x$ and $l_y$ denote the labels of x and y. The class-conditional pixel-wise and label loss is:

$$c_{pixel}([\mathrm{x}, l_x], [\mathrm{y}, l_y]) = \left|\left|[\mathrm{x}, \alpha_c * \mathrm{onehot}(l_x)]^T - [\mathrm{y}, \alpha_c * \mathrm{onehot}(l_y)]^T\right|\right|_2^2. \tag{2}$$

To our best knowledge, we are the first to propose a class-conditioning method for the Sinkhorn divergence framework. This may be of independent interest in non-private learning settings.

For privacy protection, rather than adding noise to the gradients of the generator parameters $\theta$, we add noise to the gradients of the generated images $\nabla_{\mathbf{X}}\hat{S}$, which is then backpropagated to $\theta$. Compared to $\nabla_\theta \hat{S}$, the dimension of $\nabla_{\mathbf{X}}\hat{S}$ is independent from the network architecture. This allows us to train larger networks without requiring more aggressive clipping. This method has been independently proposed by (8). In each iteration, gradient descent updates the generator parameters by backpropagating the "image gradient" $\nabla_{\mathbf{X}}\hat{S}$. We clip this term such that $||\nabla_{\mathbf{X}}\hat{S}||_2 \leq C$. The sensitivity is thus $\max_{\mathbf{Y},\mathbf{Y}'} ||\nabla_{\mathbf{X}}\hat{S}(\mathbf{X},\mathbf{Y}) - \nabla_{\mathbf{X}}\hat{S}(\mathbf{X},\mathbf{Y}')||_2 \leq 2C$. By adding Gaussian noise with scale $2C\sigma$, the mechanism satisfies $(\alpha, \frac{\alpha C^2}{2\sigma^2})$-RDP (22). We use the RDP accountant with Poisson subsampling proposed in (23) for privacy composition. Note that the batch size of $\mathbf{X}$ is kept fixed, while batch size of $\mathbf{Y}$ follows a binomial distribution due to Poisson subsampling.

## 3 Experiments

We conduct labelled image generation experiments on 3 datasets: MNIST (24), Fashion-MNIST (25), and CelebA (26) downsampled to 32x32. We evaluate the usefulness of synthetic data for downstream tasks by measuring the accuracy of classifiers trained on synthetic data and tested on real data. We also compute FID (27) scores as a measure of visual quality. Full experimental details can be found in Appendix D.

**MNIST & Fashion-MNIST** We train shallow convolutional networks based on DCGAN (28). Given the same privacy budget, DP-Sinkhorn generates more informative examples than previous methods, as demonstrated by the higher accuracy achieved by the downstream classifier. The FID of images generated by DP-Sinkhorn is lower than all baselines, except GS-WGAN (8). We believe this is due to the lack of edge sharpness in images generated by DP-sinkhorn, and attribute this to the choice of L2 pixel distance as our transport cost function as shown in Figure 2.

Table 1: Differentially private image generation results on MNIST and Fashion-MNIST at $(10, 10^{-5})$-DP. Results for other methods (DP-CGAN, DP-MERF AE (29), and GS-WGAN (8)) are from (8). Results are averaged over 5 runs of synthetic dataset generation.

| Method | DP-$\epsilon$ | MNIST | | | | Fashion-MNIST | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | FID | Acc (%) | | | FID | Acc (%) | | |
| | | | Log Reg | MLP | CNN | | Log Reg | MLP | CNN |
| Real data | $\infty$ | 1.6 | 92.2 | 97.5 | 99.3 | 2.5 | 84.5 | 88.2 | 90.8 |
| Non-priv Sinkhorn | $\infty$ | 84.1 | 88.6 | 88.2 | 87.9 | 105.2 | 77.6 | 78.7 | 72.8 |
| DP-CGAN | 10 | 179.2 | 60 | 60 | 63 | 243.8 | 51 | 50 | 46 |
| DP-MERF AE[1] | 10 | 161.1 | 54 | 55 | 68 | 213.6 | 50 | 56 | 62 |
| GS-WGAN | 10 | **61.3** | 79 | 79 | 80 | **131.3** | 68 | 65 | 65 |
| DP-Sinkhorn | 10 | 124.3 | **82.0** | **80.8** | 79.9 | 193.8 | **73.4** | **72.2** | 67.5 |

**CelebA** For CelebA we use the deeper BigGAN (9) as generator backbone. To the extent of our knowledge, no DP generative learning method has been applied on such RGB image data without accessing public data. We evaluate whether DP-Sinkhorn is able to synthesize RGB images that are informative for downstream classification. Furthermore, we study whether an adversarial learning scheme is helpful for learning this dataset. We test this by using an adversarially trained feature extractor in the cost function. Given feature extractor $\phi$, we modify the cost function as:

$$c_{adv}([\mathrm{x}, l_x], [\mathrm{y}, l_y]) = \left|\left|[\mathrm{x}, \alpha_c * \mathrm{onehot}(l_x), \frac{\alpha_a \phi(\mathrm{x})}{||\phi(\mathrm{x})||_2}]^T - [\mathrm{y}, \alpha_c * \mathrm{onehot}(l_y), \frac{\alpha_a \phi(\mathrm{y})}{||\phi(\mathrm{y})||_2}]^T\right|\right|_2^2. \tag{3}$$

Training proceeds through alternating updates. To make this learning scheme private, we randomly split the training dataset into $k$ partitions and train one discriminator per partition. This way, the

[1]DP-MERF is designed for the low-$\varepsilon$ regime and does not make use of the extra privacy budget from $\varepsilon = 10$.

gradient computed on each partition can benefit from privacy amplification by subsampling. Privacy accounting for this setting is performed through (30), which analyzes fixed-size batch subsampling.

We find that while using the adversarial feature extractor is beneficial in the non-private case, it did not provide significant improvements in the private case, as shown in Table 2. We hypothesize that clipping and gradient noise counteracts the effect of gradient shaping provided by the adversarial feature extractor, resulting in similar learning performance as in the non-adversarial approach. Despite its simplicity, DP-Sinkhorn generates informative data for gender classification. Qualitatively, Figure 3 illustrates that DP-Sinkhorn can learn rough representations of each semantic class (male and female) and also produces some in-class variations.

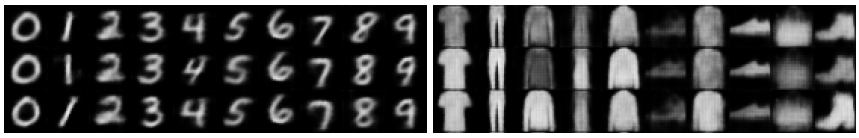

Figure 2: Samples from $(10, 10^{-5})$-DP models. Comparison with other methods can be found in Appendix E.

Table 2: Differentially private image generation results on downsampled CelebA .

| Method | $(\epsilon, 10^{-6})$-DP | FID | MLP Acc (%) | CNN Acc (%) |
|---|---|---|---|---|
| Real data | $\infty$ | 1.1 | 91.9 | 95.0 |
| Adversarial Sinkhorn (non-priv) | $\infty$ | 72.8 | 78.1 | 78.2 |
| Pixel Sinkhorn (non-priv) | $\infty$ | 140.7 | 78.9 | 77.9 |
| Adversarial DP-Sinkhorn | 10 | 187.0 | 74.7 | 74.5 |
| Pixel DP-Sinkhorn | 10 | 168.4 | 76.2 | 75.8 |

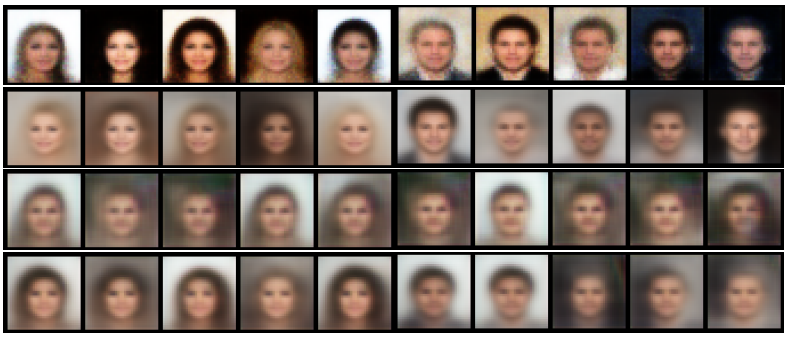

Figure 3: Images Generated on CelebA dataset. From top to botton: Adversarial Sinkhorn, Pixel Sinkhorn, Adversarial DP-Sinkhorn, Pixel DP-Sinkhorn.

## 4 Conclusions

We propose DP-Sinkhorn, a novel OT-based differentially private generative modeling method. Our approach minimizes the empirical Sinkhorn loss in a differentially private manner and does not require any adversarial techniques. Therefore, DP-Sinkhorn is easy to train and deploy, which we hope will help its adoption in practice. We also use a novel trick to force the optimal transport plan to couple same-label data together, thereby allowing for conditional data generation in the Sinkhorn divergence generative modeling framework. Our proposed method and demonstrate superior performance compared to the previous state-of-the-art in utility of generated data, as shown through downstream classification benchmarks. We also show DP synthesis of informative RGB images without using additional public data. Note that our main experiments only used a simple pixel-wise L2-loss as cost function. This suggests that in the differential privacy setting, where significant gradient perturbations are added to the model, complexity in model and objective are not necessarily beneficial. We conclude that simple and robust models such as ours are a promising direction for DP generative modeling.

## References


[1] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of cryptography conference*, pp. 265–284, Springer, 2006.

[2] C. Dwork and A. Roth, "The Algorithmic Foundations of Differential Privacy," vol. 9, no. 3–4, 2014.

[3] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," (New York, NY, USA), Association for Computing Machinery, 2016.

[4] L. Xie, K. Lin, S. Wang, F. Wang, and J. Zhou, "Differentially private generative adversarial network," *arXiv preprint arXiv:1802.06739*, 2018.

[5] R. Torkzadehmahani, P. Kairouz, and B. Paten, "Dp-cgan: Differentially private synthetic data and label generation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 0–0, 2019.

[6] L. Frigerio, A. S. de Oliveira, L. Gomez, and P. Duverger, "Differentially private generative adversarial networks for time series, continuous, and discrete open data," in *IFIP International Conference on ICT Systems Security and Privacy Protection*, pp. 151–164, Springer, 2019.

[7] J. Yoon, J. Jordon, and M. van der Schaar, "PATE-GAN: Generating synthetic data with differential privacy guarantees," in *International Conference on Learning Representations*, 2019.

[8] D. Chen, T. Orekondy, and M. Fritz, "Gs-wgan: A gradient-sanitized approach for learning differentially private generators," *arXiv preprint arXiv:2006.08265*, 2020.

[9] A. Brock, J. Donahue, and K. Simonyan, "Large scale gan training for high fidelity natural image synthesis," in *International Conference on Learning Representations*, 2019.

[10] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8110–8119, 2020.

[11] M. Arjovsky and L. Bottou, "Towards Principled Methods for Training Generative Adversarial Networks," in *International Conference on Learning Representations*, 2017.

[12] L. Mescheder, A. Geiger, and S. Nowozin, "Which training methods for GANs do actually converge?," vol. 80 of *Proceedings of Machine Learning Research*, (Stockholmsmässan, Stockholm Sweden), pp. 3481–3490, PMLR, 10–15 Jul 2018.

[13] K. Chaudhuri and S. A. Vinterbo, "A stability-based validation procedure for differentially private machine learning," in *Advances in Neural Information Processing Systems 26* (C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, eds.), pp. 2652–2660, Curran Associates, Inc., 2013.

[14] O. Bousquet, S. Gelly, I. Tolstikhin, C.-J. Simon-Gabriel, and B. Schoelkopf, "From optimal transport to generative modeling: the vegan cookbook," *arXiv preprint arXiv:1705.07642*, 2017.

[15] G. Peyré and M. Cuturi, "Computational Optimal Transport," *Foundations and Trends in Machine Learning*, vol. 11, no. 5-6, pp. 355–607, 2019.

[16] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017.

[17] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved Training of Wasserstein GANs," in *Advances in Neural Information Processing Systems 30* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), pp. 5767–5777, Curran Associates, Inc., 2017.

[18] A. Genevay, M. Cuturi, G. Peyré, and F. Bach, "Stochastic optimization for large-scale optimal transport," in *Advances in neural information processing systems*, pp. 3440–3448, 2016.

[19] J. Feydy, T. Séjourné, F.-X. Vialard, S.-i. Amari, A. Trouvé, and G. Peyré, "Interpolating between optimal transport and mmd using sinkhorn divergences," in *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 2681–2690, 2019.

[20] A. Genevay, G. Peyré, and M. Cuturi, "Learning generative models with sinkhorn divergences," in *International Conference on Artificial Intelligence and Statistics*, pp. 1608–1617, 2018.

[21] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," in *Advances in neural information processing systems*, pp. 2292–2300, 2013.

[22] I. Mironov, "Rényi differential privacy," in *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pp. 263–275, 2017.

[23] Y. Zhu and Y.-X. Wang, "Poission subsampled rényi differential privacy," in *International Conference on Machine Learning*, pp. 7634–7642, 2019.

[24] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[25] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[26] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[27] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Advances in neural information processing systems*, pp. 6626–6637, 2017.

[28] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[29] F. Harder, K. Adamczewski, and M. Park, "Differentially private mean embeddings with random features (dp-merf) for simple & practical synthetic data generation," *arXiv preprint arXiv:2002.11603*, 2020.

[30] Y.-X. Wang, B. Balle, and S. P. Kasiviswanathan, "Subsampled rényi differential privacy and analytical moments accountant," in *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 1226–1235, PMLR, 2019.

[31] G. Acs, L. Melis, C. Castelluccia, and E. De Cristofaro, "Differentially private mixture of generative neural networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 6, pp. 1109–1121, 2018.

[32] S. Takagi, T. Takahashi, Y. Cao, and M. Yoshikawa, "P3gm: Private high-dimensional data release via privacy preserving phased generative model," *arXiv preprint arXiv:2006.12101*, 2020.

[33] A. Sarwate, "Retraction for symmetric matrix perturbation for differentially-private principal component analysis," 2017.

[34] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27* (Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, eds.), pp. 2672–2680, Curran Associates, Inc., 2014.

[35] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in neural information processing systems*, pp. 5767–5777, 2017.

[36] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *International Conference on Learning Representations*, 2018.

[37] Z. Szabó and B. K. Sriperumbudur, "Characteristic and universal tensor product kernels," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 8724–8752, 2017.

[38] G. Peyré, M. Cuturi, *et al.*, "Computational optimal transport: With applications to data science," *Foundations and Trends® in Machine Learning*, vol. 11, no. 5-6, pp. 355–607, 2019.

[39] B. Balle, G. Barthe, and M. Gaboardi, "Privacy amplification by subsampling: Tight analyses via couplings and divergences," in *Advances in Neural Information Processing Systems*, pp. 6277–6287, 2018.

[40] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings* (Y. Bengio and Y. LeCun, eds.), 2015.

# A Related Works

Our method compares favorably against prior works that use GAN-based training schemes (4; 5; 6; 7; 8) due to training stability. As discussed, this is critical in the context of differential privacy, where the imposed gradient noise can increase training instabilities and where interaction with private data should be limited. It is worth noting that (8) proposed the same privacy barrier mechanism like us by applying gradient noise on the generated images rather than the generator parameters. Other types of generative models, including likelihood based models (31) and MMD with random Fourier features (29) have also been studied in the privacy preserving setting. DP-Sinkhorn is more expressive and thus fits real distributions more faithfully. Lastly, while (32) produced strong empirical results, their privacy analysis relies on the use of Wishart noise on sample covariance matrices, which has been proven to leak privacy (33). Hence, their privacy protection is invalid in its current form.

# B Background

## B.1 Notations and Setting

Let $\mathcal{X}$ denote a sample space, $\mathcal{P}(\mathcal{X})$ all possible measures on $\mathcal{X}$, and $\mathcal{Z} \subseteq \mathbb{R}^d$ the latent space. We are interested in training a generative model $g : \mathcal{Z} \mapsto \mathcal{X}$ such that its induced distribution $\mu = g \circ \xi$ with noise source $\xi \in \mathcal{P}(\mathcal{Z})$ is similar to observed $\nu$ through an independently sampled finite sized set of observations $D = \{y\}^N$. In our case, $g$ is a trainable parametric function with parameters $\theta$. We denote the Dirac delta distribution at $x \in \mathcal{X}$ as $\delta_x$, and the standard $n$-simplex as $\mathcal{S}^n$.

## B.2 Generative Learning with Optimal Transport

Optimal Transport-based generative learning considers minimizing variants of the Wasserstein distance between real and generated distributions (14; 15). Two key advantages of the Wasserstein distance over standard GANs, which optimize the Jensen-Shannon divergence (34), are its definiteness on distributions with non-overlapping supports, and its weak metrization of probability spaces (16). This prevents collapse during training caused by discriminators that are overfit to the training examples.

Methods implementing the OT framework use either the primal or the dual formulation. For generative learning, the dual formulation has been more popular. Under the dual formulation, the distance between the generator-induced and the data distribution is computed as the expectation of potential functions over the sample space. In WGAN and variants (16; 35; 36), the dual potential is approximated by an adversarially trained discriminator network. While theoretically sound, these methods still encounter instabilities during training since the non-optimality of the discriminator can produce arbitrarily large biases in the generator gradient (14). The primal formulation involves solving for the optimal transport plan—a joint distribution over the real and generated sample spaces. The distance between the two distributions is then measured as the expectation of a point-wise cost function between pairs of samples as distributed according to the transport plan. Thus, the properties of the point-wise cost function are of great importance. In practice, sufficiently convex cost functions allow for an efficient optimization of the generator. It is also possible to learn cost functions adversarially (37). However, as discussed earlier, adversarial training often comes with additional challenges, which can be especially problematic in the differential privacy setting.

In general, finding the optimal transport plan is a difficult optimization problem due to the constraints of equality between its marginals and the real and generated distributions. Entropy Regularized Wasserstein Distance (ERWD) imposes a strongly convex regularization term on the Wasserstein distance, making the OT problem between finite samples solvable in linear time (38). Given a positive cost function $c : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}^+$ and $\epsilon \geq 0$, the Entropy Regularized Wasserstein Distance is defined as:

$$W_{c,\epsilon}(\mu, \nu) = \min_{\pi \in \Pi} \int c(x, y)\pi(x, y) + \epsilon \int \log \frac{\pi(x, y)}{d\mu(x)d\nu(y)} d\pi(x, y) \tag{4}$$

where $\Pi = \left\{\pi(x, y) \in \mathcal{P}(\mathcal{X} \times \mathcal{X}) | \int \pi(x, \cdot)dx = \nu, \int \pi(\cdot, y)dy = \mu\right\}$. Sinkhorn divergence uses cross correlation terms to cancel out the entropic bias introduced by ERWD. This results in faithful matching between the generator and real distributions. In this paper, we use the Sinkhorn divergence as defined in (19).

**Definition B.1** *(Sinkhorn Loss) The Sinkhorn loss between measures $\mu$ and $\nu$ is defined as:*

$$S_{c,\epsilon}(\mu,\nu) = 2W_{c,\epsilon}(\mu,\nu) - W_{c,\epsilon}(\mu,\mu) - W_{c,\epsilon}(\nu,\nu) \tag{5}$$

For modeling data-defined distributions, as in our situation, an empirical version can be defined, too.

**Definition B.2** *(Empirical Sinkhorn loss) The empirical Sinkhorn loss computed over a batch of $N$ generated examples and $M$ real examples is defined as:*

$$\hat{S}_{c,\epsilon}(\hat{\mu},\hat{\nu}) = 2C_{XY} \odot P^*_{\epsilon,X,Y} - C_{XX} \odot P^*_{\epsilon,X,X} - C_{YY} \odot P^*_{\epsilon,Y,Y} \tag{6}$$

*where $\hat{\mu} = \frac{1}{N}\sum_{i=1}^{N} \delta_{x_i}$, and $\hat{\nu} = \frac{1}{M}\sum_{j=1}^{M} \delta_{y_j}$. For two samples $A \in \mathcal{X}^N$ and $B \in \mathcal{X}^M$, $C_{AB}$ is the cost matrix between $A$ and $B$, and $P^*_{\epsilon,A,B}$ is an approximate optimal transport plan that minimizes Eqn. 4 computed over $A$ and $B$.*

$P^*_\epsilon$ is arrived at by iterating the dual potentials.

(21) and (19) have shown the following dual formulation for the discritized version of $\hat{W}_{c,\epsilon}$:

$$\hat{W}_{c,\epsilon}(\hat{\mu},\hat{\nu}) = \max_{f,g \in \mathcal{S}^N \times \mathcal{S}^M} \langle \hat{\mu}, f\rangle + \langle \hat{\nu}, g\rangle - \epsilon \langle \hat{\mu} \otimes \hat{\nu}, \exp(\frac{1}{\epsilon}(f \oplus g - C)) - 1\rangle, \tag{7}$$

where $\otimes$ denotes the product measure and $\oplus$ denotes the "outer sum" such that the output is a matrix of the sums of pairs of elements from each vector. Then, the optimal transport plan $P^*_\epsilon$ relates to the dual potentials by $P^*_\epsilon = \exp(\frac{1}{\epsilon}(f \oplus g - C))(\hat{\mu} \otimes \hat{\nu})$. Thus, once we find the optimal $f$ and $g$, we can obtain $P^*_\epsilon$ through this primal-dual relationship. We also know the first-order optimal conditions for $f$ and $g$ through the Karush Kuhn Tucker theorem:

$$f_i = -\epsilon \log \sum_{j=1}^{M} \exp(\log(\hat{\nu}_j) + \frac{1}{\epsilon} g_j - \frac{1}{\epsilon} C(x_i, y_j)) \quad g_j = -\epsilon \log \sum_{i=1}^{N} \exp(\log(\hat{\mu}_i) + \frac{1}{\epsilon} f_i - \frac{1}{\epsilon} C(x_i, y_j)) \tag{8}$$

To optimize $f$ and $g$, it suffices to apply the Sinkhorn algorithm (21), see Algorithm 2 in the main text.

Works on the efficient computation of the Sinkhorn divergence have yielded algorithms that converge to the optimal transport plan within tens of iterations (21; 19). Gradient computation follows by taking the Jacobian vector product between the cost matrix Jacobian and the transport weights, which is efficiently implemented in many reverse-mode auto-differentiation frameworks. The above loss easily extends to the setting of empirical data-defined distributions, which is the situation we are interested in.

### B.3 Differential Privacy

The current gold standard for measuring the privacy risk of data releasing programs is the notion of differential privacy (DP) (1).

**Definition B.3** *(Differential Privacy) A randomized mechanism $\mathcal{M} : \mathcal{D} \to \mathcal{R}$ with domain $\mathcal{D}$ and range $\mathcal{R}$ satisfies $(\varepsilon, \delta)$-DP if for any two adjacent inputs $d, d' \in \mathcal{D}$ differing by at most one entry, and for any subset of outputs $S \subseteq \mathcal{R}$ it holds that*

$$\mathbf{Pr}\left[\mathcal{M}(d) \in S\right] \leq e^{\varepsilon} \mathbf{Pr}\left[\mathcal{M}(d') \in S\right] + \delta. \tag{9}$$

Informally, DP measures to what degree a program's output can deviate between adjacent input datasets—sets which differ by one entry. For a user contributing their data, this translates to a guarantee on how much an adversary could learn about them from observing the program's output. In this paper, we are interested in the domain of image-label datasets where each image and its semantic label constitute an entry.

**Gradient perturbation** with the Gaussian mechanism is the most popular method for DP learning of parametric models. During stochastic gradient descent (SGD) and variants, the parameter gradients are clipped in 2-norm by a constant, and then Gaussian noise is added. By adding a sufficient amount of noise, the gradients can satisfy DP requirements. Then, the *post-processing* property of differential privacy (2) guarantees that the parameters are also private. The relation between $(\varepsilon, \delta)$ and the perturbation parameters $\Delta$ and $\sigma$ is provided by the following theorem:

**Theorem B.1** *For $c^2 > 2\log(1.25/\delta)$, Gaussian mechanism with $\sigma \geq c\Delta/\varepsilon$ satisfies $(\varepsilon, \delta)$ differential privacy. (2)*

As SGD involves computing the gradients on randomly drawn batches of data for multiple iterations, two other properties of DP, *composition* and *subsampling* are required to analyze the privacy of the algorithm. Rényi Differential Privacy (RDP) is a well-studied formulation of privacy that allows tight composition of multiple queries, and can be easily converted to standard definitions of DP.

**Definition B.4** *(Rényi Differential Privacy) A randomized mechanism $\mathcal{M} : \mathcal{D} \to \mathcal{R}$ with domain $\mathcal{D}$ and range $\mathcal{R}$ satisfies $(\alpha, \epsilon)$-RDP if for any adjacent $d, d' \in \mathcal{D}$ it holds that*

$$D_\alpha(\mathcal{M}(d)|\mathcal{M}(d')) \leq \epsilon, \tag{10}$$

*where $D_\alpha$ is the Rényi divergence of order $\alpha$. Also, any $\mathcal{M}$ that satisfies $(\alpha, \epsilon)$-RDP also satisfies $(\epsilon + \frac{\log 1/\delta}{\alpha - 1}, \delta)$-DP (22).*

For clipping threshold $\Delta$ and standard deviation of Gaussian noise $\sigma$, the Gaussian mechanism satisfies $(\alpha, \alpha\Delta^2/(2\sigma^2))$-RDP (22). Subsampling the dataset into batches also improves privacy. The effect of subsampling on the Gaussian mechanism under RDP has been studied in (30; 39; 23). Privacy analysis of a gradient-based learning algorithm entails accounting for the privacy cost of single queries, which corresponds to training iterations in our case, possibly with subsampling due to mini-batched training. The total privacy cost is obtained by summing up the privacy cost across all queries or training steps, and then choosing the best $\alpha$. For completeness, the Rényi divergence is defined as: $D_\alpha(P|Q) = \frac{1}{\alpha}\log \mathbb{E}_{x \in Q}\left[\frac{P(x)}{Q(x)}\right]^\alpha$.

# C Algorithms

**Algorithm 1** Poisson Sample

**Input** : $d = \{(\mathrm{y}, \mathrm{l}) \in \mathcal{X} \times \{0, ..., L\}\}^M$, sampling ratio $q$
**Output**: $\mathbf{Y} = \{(\mathrm{y}_j, \mathrm{l}_j) \in \mathcal{X} \times \{0, ..., L\}\}_{j=1}^m$, $m \geq 0$
$\mathrm{s} = \{\sigma_i\}_{i=1}^M \overset{i.i.d.}{\sim} \text{Bernoulli}(q)$
$\mathbf{Y} = \{d_j | \mathrm{s}_j = 1\}_{j=1}^M$

**Algorithm 2** Sinkhorn Algorithm $\hat{W}_\epsilon(\mathbf{X}, \mathbf{Y})$

**Input:** $\mathbf{X} = \{\mathrm{x}\}^n, \mathbf{Y} = \{\mathrm{y}\}^m, \epsilon$
**Output:** $W_\epsilon$
$\forall (i,j), C_{[i,j]} = c(\mathbf{X}_i, \mathbf{Y}_j)$
$\mathbf{f}, \mathbf{g} \leftarrow \vec{0}$
$\hat{\mu}, \hat{\nu} \leftarrow \text{Unif}(n), \text{Unif}(m)$
**while** not converged **do**
  $\forall i, \mathbf{f}_i \leftarrow -\epsilon \log \sum_{k=1}^m \exp(\log(\hat{\nu}_k) + \frac{1}{\epsilon}\mathbf{g}_k - \frac{1}{\epsilon}C_{[i,k]})$
  $\forall j, \mathbf{g}_j \leftarrow -\epsilon \log \sum_{k=1}^n \exp(\log(\hat{\mu}_k) + \frac{1}{\epsilon}\mathbf{f}_k - \frac{1}{\epsilon}C_{[k,j]})$
**end while**
$W_\epsilon = \langle \hat{\mu}, \mathbf{f} \rangle + \langle \hat{\nu}, \mathbf{g} \rangle$

**Algorithm 3** DP-Sinkhorn
$L$ is number of categories, $\mathcal{X}$ is sample space. $M$ is size of private data set. $backprop$ is a reverse mode auto-differentiation function that takes 'out', 'in' and 'grad weights' as input and computes the Jacobian vector product $J_{\text{in}}(\text{out}) \cdot \text{grad weights}$.

**Input:** private data set $d = \{(\mathrm{y}, \mathrm{l}) \in \mathcal{X} \times \{0, ..., L\}\}^M$, sampling ratio $q$, noise scale $\sigma$, clipping coefficient $\Delta$, generator $g_\theta$, learning rate $\alpha$, entropy regularization $\epsilon$, total steps $T$.
**Output:** $\theta$
$n = q * M$
**for** $t = 1$ **to** $T$ **do**
  Sample $\mathbf{Y} \leftarrow$ Poisson Sample$(d, q)$,
  $Z \leftarrow (\mathrm{z}_i)_{i=1}^n \overset{i.i.d.}{\sim} \text{Unif}(0, 1)$
  $L_x \leftarrow \{\mathrm{l}_i\}_{i=1}^n \overset{i.i.d.}{\sim} \text{Unif}(0, ..., L)$
  $\mathbf{X} \leftarrow \{x_i = g_\theta(\mathrm{z}_i, \mathrm{l}_i)\}_{i=1}^n$
  $\text{grad}_{\mathbf{X}} \leftarrow \nabla_{\mathbf{X}}\left[2\hat{W}_\epsilon(\mathbf{X}, \mathbf{Y}) - \hat{W}_\epsilon(\mathbf{X}, \mathbf{X})\right]$
  $\text{grad}_{\mathbf{X}} \leftarrow clip(\text{grad}_{\mathbf{X}}, \Delta) + 2\Delta\sigma\mathcal{N}(\vec{0}, \mathbb{I})$
  $\text{grad}_\theta \leftarrow backprop(\mathbf{X}, \theta, \text{grad}_{\mathbf{X}})$
  $\theta \leftarrow \theta - \alpha * Adam(\text{grad}_\theta)$
**end for**

# D Experiment Details

## D.1 Datasets

MNIST and Fashion-MNIST both consist of 28x28 grayscale images, partitioned into 60k training images and 10k test images. The 10 labels of the original classification task correspond to digit/object

class. For calculating FID scores, we repeat the channel dimension 3 times. CelebA is composed of $\sim$200k colour images of celebrity faces tagged with 40 binary attributes. We downsample all images to 32x32, and use all 162770 train images for training and all 19962 test images for evaluation. Generation is conditioned on the gender attribute.

### D.2 Metrics

In all experiments, we compute metrics against a synthetic dataset of 60k image-label pairs sampled from the model. Each sample is formed by selecting a class uniformly at random and generating an image conditioned on that class. For a quantitative measure of visual quality, we report FID (27). We compute FID scores between our synthetic datasets of size 60k and the full test data (either 10k or 19962 images). To measure the utility of generated data, we assess the class prediction accuracy of classifiers trained with synthetic data on the real test sets. We consider logistic regression, MLP, and CNN classifiers. Previous work also reports classification accuracies on a large suite of classifiers from scikit-learn. We omit them, since we focus on images which are best processed via neural network-based classifiers.

### D.3 Classifiers

For logistic regression, we use scikit-learn's implementation, using the L-BFGS solver and capping the maximum number of iterations at 5000. The MLP and CNN are implemented in PyTorch. The MLP has one hidden layer with 100 units and a ReLU activation. The CNN has two hidden layers with 32 and 64 filters, and uses ReLU activations. We train the CNN with dropout ($p = 0.5$) between all intermediate layers. Both the MLP and CNN are trained with Adam under default parameters, and use 10% of training data as holdout for early stopping. Training stops after no improvement is seen in holdout accuracy for 10 consecutive epochs.

### D.4 Architecture, Hyperparameter, and Implementation

Our DCGAN-based architecture uses 4 transposed convolutional layers with ReLU activations at the hidden layers and tanh activation at the output layer. A latent dimension of 12 and class embedding dimension of 4 is used for MNIST and Fashion-MNIST experiments. CelebA experiments use a latent dimension of 32 and embedding dimension of 4. The latent and class embeddings are concatenated and then fed to the convolutional stack. The first transposed convolutional layer projects the input to $256 \times 7 \times 7$, with no padding. Layers 2,3 and 4 have output depth $[128, 64, 1]$, kernel size $[4, 4, 3]$, stride $[2, 2, 1]$, and padding $[1, 1, 1]$.

Our BigGAN-based architecture uses 4 residual blocks of depth 256, and a latent dimension of 32. Each residual block consists of three convolutional layers with ReLU activations and spectral normalization between each layer. Please refer to (9) for more implementation details. Our implementation is based on `https://github.com/ajbrock/BigGAN-PyTorch`.

In our experiments with the adversarially trained feature extractor, we used a simple feature extractor with 3 convolutional layers with hidden depth of 64 and output depth of 32. Between each convolutional layer are batchnorm and ReLU layers, followed a $2 \times 2$ maxpool layer. The first two layers have kernel size 3 and padding 1, while the last layers have kernel size 1 with no padding. For the generator $G_\theta$ and feature extractor $\phi_\omega$, the adversarial loss objective can be formally expressed as:

$$\begin{aligned}\min_\theta \max_\omega \hat{S}_{c(\omega),\epsilon}(\hat{\mu}(\theta), \hat{\nu}) = \min_\theta \max_\omega & 2\langle c_\phi(G_\theta(z), \mathrm{y}), P^*_\epsilon(G_\theta(z), \mathrm{y})\rangle \\ & - \langle c_\phi(G_\theta(z), G_\theta(z)), P^*_\epsilon(G_\theta(z), G_\theta(z))\rangle \\ & - \langle c_\phi(\mathrm{y}, \mathrm{y}), P^*_\epsilon(\mathrm{y}, \mathrm{y})\rangle\end{aligned}$$

Where $c_\phi(\mathrm{a}, \mathrm{a}) = ||[\mathrm{a}_{img}, \alpha_c \mathrm{a}_{label}, \alpha_a \frac{\phi(\mathrm{a}_{img})}{||\phi(\mathrm{a}_{img})||_2}]^T - [\mathrm{b}_{img}, \alpha_c \mathrm{b}_{label}, \alpha_a \frac{\phi(\mathrm{b}_{img})}{||\phi(\mathrm{b}_{img})||_2}]^T||_2^2 = c_{adv}([\mathrm{a}_{img}, \mathrm{a}_{label}], [\mathrm{b}_{img}, \mathrm{b}_{label}])$

Hyperparameters of the Sinkhorn loss used were: $\alpha_c = 15$, and entropy regularization $\epsilon = 0.05$ in MNIST and Fashion-MNIST experiments. $\epsilon = 5$ is used for CelebA experiments. We use the implementation publically available at `https://www.kernel-operations.io/geomloss/api/install` and all other hyperparameters are kept at their default values. For all experiments, we use the Adam (40) optimizer with learning rate $10^{-5}$, $\beta = (0.9, 0.999)$, weight decay $2 \times 10^{-5}$.

For privacy accounting, we use the implementation of the RDP Accountant available in Tensorflow Privacy.[1] All experiments employing the pixel-wise $L2$ cost use Poisson sampling, and are amenable to the analysis implemented in `compute_rdp`. For the experiments with the batch-wise feature extractor, examples are batched into records at the start of training, and a single record is drawn at random in each iteration. The privacy amplification for this fixed-size sampling scheme is studied in (30), and we use the author's implementation of Theorem 9 for bounding the RDP cost of our queries.

For MNIST and Fashion-MNIST results reported in the main body, we use a noise scale of $\sigma = 1.1$ and a batch size of 50 resulting in $q = 1/1200$, which gives us $\sim 3.4$ million training iterations to reach $\varepsilon = 10$ for $\delta = 10^{-5}$. For the non-private runs, we use a batch size of 500, which improves image quality and diversity. When training with DP, increasing batch size significantly increases the privacy cost per iteration, resulting in poor image quality for fixed $\varepsilon = 10$.

For CelebA results reported in the main body, we use a noise scale of $\sigma = 0.8$ and a batch size of 200 resulting in $q = 0.00123$. At $\delta = 10^{-6}$, we train for 1.1 million steps to reach $\varepsilon = 10$.

# E Additional Results

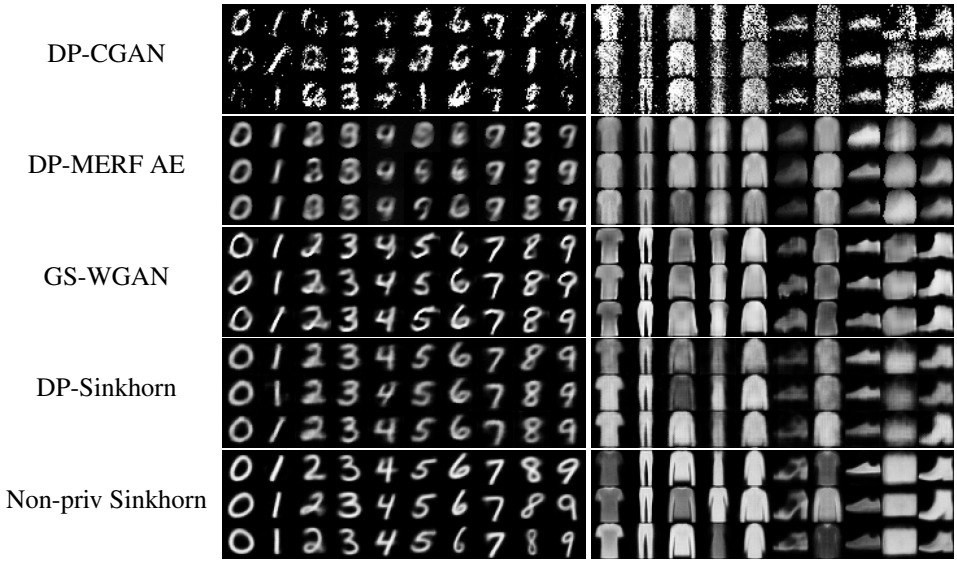


Figure 4: Samples from the methods in Table 1. All private models are $(10, 10^{-5})$-DP. The first 3 rows showcasing other methods are from (8).

We evaluate the impact of architecture choice on the performance in the CelebA task by comparing DP-Sinkhorn+BigGAN with DP-Sinkhorn+DCGAN, under pixel loss. Results are summarized in Table 3 and visualized in Figure 5. Qualitatively, despite reaching lower FID score, the DCGAN-based generator's images have visible artifacts that are not present in models trained with BigGAN-generators.

Table 3: Differetially private image generation results on downsampled CelebA.

| Method | DP-$\epsilon$ | FID | Acc (%) | |
|---|---|---|---|---|
| | | | MLP | CNN |
| Real data | $\infty$ | 1.1 | 91.9 | 95.0 |
| DCGAN+DP-Sinkhorn | 10 | 156.7 | 75.0 | 74.6 |
| BigGAN+DP-Sinkhorn | 10 | 168.4 | 76.2 | 75.8 |

[1] `https://github.com/tensorflow/privacy/`

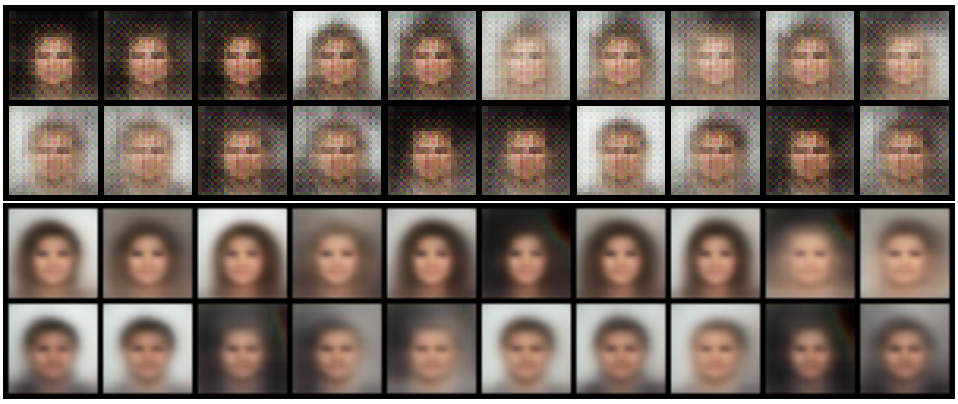

Figure 5: Additional DP-Sinkhorn generated images under $(10, 10^{-6})$differential privacy. Top two rows use DCGAN based generator, while bottom two rows use BigGAN based generator.