CM-GAN: Stabilizing GAN Training with Consistency Models

Haoye Lu [1 2] Yiwei Lu [1 2] Dihong Jiang [1 2] Spencer Ryan Szabados [1 2] Sun Sun [1 3] Yaoliang Yu [1 2]

Abstract

In recent years, generative adversarial networks (GANs) have gained attention for their ability to generate realistic images, despite being notoriously difficult to train. On the other hand, diffusion models have emerged as a promising alternative, offering stable training processes and avoiding mode collapse issues; however, their generation process is computationally expensive. To overcome this problem, Song et al. (2023) proposed consistency models (CMs) that are optimized through a novel consistency constraint induced by the underlying diffusion process. In this paper, we show that the same consistency constraint can be used to stabilize the training of GANs and alleviate the notorious mode collapse problem. In this way, we provide a method to combine the main strengths of diffusions and GANs while mitigating their major drawbacks. Additionally, as the technique can also be viewed as a method to fine-tune the consistency models using a discriminator, its performance is expected to outperform CM in general. We provide preliminary empirical results on MNIST to corroborate our claims.

1. Introduction

Generative adversarial networks (Goodfellow et al., 2014; Brock et al., 2019; Karras et al., 2021) have made remarkable success in generating high-resolution images that closely resemble real photos. However, practical implementation of generative adversarial networks (GANs) often encounters several challenges, such as non-convergence, training instability, and mode collapse, where the generated outputs become repetitive or limited in variation (Goodfellow, 2016; Arjovsky & Bottou, 2017; Mescheder et al., 2018). To address these challenges, many theoretical and empirical attempts have been made including: enhancing network architectures (Mescheder et al., 2017; Arjovsky & Bottou, 2017; Li et al., 2017b), developing theoretical insights into GAN training dynamics (Nowozin et al., 2016), devising new objective functions (Nowozin et al., 2016; Arjovsky et al., 2017a; Zheng & Zhou, 2021), and incorporating mappings from data to latent representations (Donahue et al., 2017; Dumoulin et al., 2017; Li et al., 2017a).

Recently, diffusion-based generative models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021a;b; 2023) have gained increasing attention and many impressive breakthroughs have been made (Croitoru et al., 2023) in generating images (Ho et al., 2020; Song et al., 2021a;b; Rombach et al., 2022; Song et al., 2023), audios (Kong et al., 2021; Yang et al., 2023) and videos (Ho et al., 2022). Due to some inherent properties, diffusion models are relatively easier to train and do not suffer from those common training difficulties of GANs. In contrast, its generation process involves iteratively applying denoising steps to progressively transform noise into data samples (Ho et al., 2020) or solving a complex ODE system using an iterative solver (Song et al., 2021b), which is computationally expensive. To alleviate this difficulty, Song et al. (2023) proposed consistency models (CMs). By adopting a novel local consistency constraint, the model can be either distilled from a pre-trained diffusion model or trained from scratch, enabling a single-step generation process.

In this paper, we introduce a novel approach that leverages the consistency constraint to enhance the training stability of GANs and overcome the well-known issue of mode collapse. Our method involves utilizing a possibly under-trained diffusion model as a prototype and enforces consistency constraints to ensure that the generator produces outputs similar to those of the diffusion model. In this way, we provide a method to combine the main strengths of diffusions and GANs while mitigating their major drawbacks. Moreover, this technique can be seen as a means of fine-tuning the CMs by integrating a discriminator, thereby potentially surpassing the performance of CMs in general. Our claims are supported by the preliminary empirical study conducted on MNIST datasets.

[1] Cheriton School of Computer Science, University of Waterloo, Waterloo, Canada [2] Vector Institute, Toronto, Canada [3] National Research Council Canada, Waterloo, Canada. Correspondence to: Haoye Lu <haoye.lu@uwaterloo.ca>.

Accepted to ICML workshop on Structured Probabilistic Inference & Generative Modeling, Honolulu, Hawaii, USA, 2023. Copyright 2023 by the author(s).

2. Preliminary

2.1. Generative adversarial networks

Generative adversarial networks (Goodfellow et al., 2014) are a family of generative models that learn a data distribution p_{data} by establishing a min-max game between two neural networks: a generator $\mathcal{G}$ and a discriminator $\mathcal{D}$.

The generator $\mathcal{G}$ is expected to take a random noise vector $\mathbf{z}$ sampled from some prior distribution p_{prior} (typically a spherical Gaussian distribution) and output a fake sample $\mathcal{G}(\mathbf{z})$ lying in the support of p_{data}. A discriminator $\mathcal{D}$ is simultaneously trained to distinguish $\mathcal{G}(\mathbf{z})$ from real data $\mathbf{x}$. Specifically, $\mathcal{D}$ is optimized to correctly distinguish the fake samples generated by $\mathcal{G}$ from real training samples while $\mathcal{G}$ is trained to generate more realistic samples to fool $\mathcal{D}$. The relationship between $\mathcal{D}$ and $\mathcal{G}$ can be characterized by a min-max objective function:

$$\min_{\mathcal{G}} \max_{\mathcal{D}} \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}}\Big[\log \mathcal{D}(\mathbf{x})\Big] + \mathbb{E}_{\mathbf{z}\sim p_{\text{prior}}}\Big[\log\big(1-\mathcal{D}(\mathcal{G}(\mathbf{z}))\big)\Big]. \tag{1}$$

In practice, the optimization of GANs is usually unstable and experiences the gradient vanishing problem; as a result, the objective function (1) is often modified to improve the stability and performance of GANs (Goodfellow et al., 2014; Arjovsky et al., 2017b; Miyato et al., 2018; Fedus et al., 2018) while the general idea behind the competitive dynamic between $\mathcal{G}$ and $\mathcal{D}$ remains the same.

Another common problem in GANs is mode collapse, where the generator barely produces a small set of outputs (Goodfellow, 2016; Arjovsky & Bottou, 2017; Mescheder et al., 2018). This happens because the generator $\mathcal{G}$ is trained to find the output that seems most plausible to the discriminator. Once $\mathcal{G}$ starts generating the same output (or a small set of outputs) consistently, the discriminator $\mathcal{D}$ may choose to remember this output and always reject it, which could get $\mathcal{D}$ stuck at a local optimum. As a result, for the next iteration, $\mathcal{G}$ could find the most plausible output for $\mathcal{D}$ easily while $\mathcal{D}$ fails to effectively improve its learning to escape this predicament. Consequently, the generator and discriminator end up cycling through a limited range of outputs.

In Section 3, we will show that the challenges mentioned earlier can be significantly mitigated by incorporating the consistency constraint (Song et al., 2023). This constraint is enforced by leveraging a pretrained diffusion model as a “prior” model, ensuring that the generator $\mathcal{G}$ remains in proximity to the prior and consistently generates diverse outputs. Thus, training becomes more stable and mode collapse is effectively avoided.

2.2. Probability flow ODE and consistency models

The probability Flow (PF) ODE and consistency models (CMs) are two families of generative models that are closely related to the continuous-time diffusion models (Song et al., 2021b). Diffusion models generate data by iteratively introducing Gaussian perturbations to the input data, gradually transforming it into noise, and subsequently generating samples from the noise through a series of sequential denoising steps. Given data distribution p_{data}, the forward perturbation is characterized by a stochastic differential equation:

$$\mathrm{d}\mathbf{x}_t = \boldsymbol{\mu}(\mathbf{x}_t, t)\,\mathrm{d}t + \sigma(t)\,\mathrm{d}\mathbf{w}_t, \tag{2}$$

for $t \in [0, T]$ and T is a fixed positive constant. $\boldsymbol{\mu}(\cdot,\cdot)$ and $\sigma(t)$ denote the drift and diffusion coefficients while $\{\mathbf{w}_t\}_{t\in[0,T]}$ is the standard Brownian motion. In this paper, we adopt the same configuration as Song et al.’s, where $\boldsymbol{\mu}(\mathbf{x}, t) = 0$ and $\sigma(t) = \sqrt{2t}$. When T is sufficiently large, $\mathbf{x}_T$ can be approximately seen as a sample following $\mathcal{N}(\mathbf{0}, T^2\mathbf{I})$. Let p_t denote the distribution of $\mathbf{x}_t$ (thus, $p_0 = p_{\text{data}}$ and $p_T \approx \mathcal{N}(\mathbf{0}, T^2\mathbf{I})$). Song et al. (2021b) proved that the solution $\tilde{\mathbf{x}}_t$ of the ODE:

$$\mathrm{d}\tilde{\mathbf{x}}_t = \Big[-t\,\nabla \log p_t(\tilde{\mathbf{x}}_t)\Big]\,\mathrm{d}t \quad \text{with } \tilde{\mathbf{x}}_T \sim p_T(\tilde{\mathbf{x}}_T) \tag{3}$$

is also distributed according to p_t, where the ODE in (3) is called the *PF-ODE*. Here, $\nabla \log p_t(\mathbf{x}_t)$ is the score function of $p_t(\mathbf{x}_t)$ and can be empirically estimated by a neural network $\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, t)$ which is notably easy to train due to the stable training process. (Readers may refer to (Song et al., 2021b) for its training details.) With a well-trained $\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, t)$, we then can plug it into (3) and solve the PF-ODE backward starting from $\tilde{\mathbf{x}}_T \sim \mathcal{N}(\mathbf{0}, T^2\mathbf{I})$ and the resulting $\tilde{\mathbf{x}}_0$ can be seen as an approximate sample of p_{data}.

Solving PF-ODE is generally expensive, which motivates Song et al. (2023) to propose CMs. Specifically, they train a neural network $\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ that maps any point $(\mathbf{x}_t, t)$ on the PF-ODE trajectory to its origin $(\mathbf{x}_0, 0)$. Then for $\tilde{\mathbf{x}}_T \sim \mathcal{N}(\mathbf{0}, T^2\mathbf{I})$, $\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_T, T)$ is an approximate sample of p_{data} and the iterative ODE solving process is avoided. To train $\mathbf{f}_{\boldsymbol{\theta}}$, they discretize interval $[0, T]$ into $N - 1$ sub-intervals with boundaries $0 = t_1 < t_2 < \cdots < t_N = T$ and adopt a special model architecture so that $\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_0, 0) = \mathbf{x}_0$. Then $\mathbf{f}_{\boldsymbol{\theta}}$ is trained to minimize a consistency distillation loss:

$$\mathcal{L}_{\text{CD}}(\boldsymbol{\theta}, \bar{\boldsymbol{\theta}}) = \mathbb{E}\Big[\lambda(t_n)\big\|\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_{t_{n+1}}, t_{n+1}) - \mathbf{f}_{\bar{\boldsymbol{\theta}}}(\hat{\mathbf{x}}_{t_n}, t_n)\big\|_2^2\Big] \tag{4}$$

where expectation is taken with respect to $\mathbf{x} \sim p_{\text{data}}$, $n \sim \mathcal{U}[\![1, N-1]\!]$, $\mathbf{x}_{t_{n+1}} \sim \mathcal{N}(\mathbf{x}; t_{n+1}^2\mathbf{I})$.[1] Here, $\mathcal{U}[\![1, N-1]\!]$ denotes a uniform distribution over $\{1, 2, \cdots, N-1\}$. $\hat{\mathbf{x}}_{t_n}$ is the solution at step t_n of the PF-ODE trajectory

[1] We only consider training CMs by distillation.

through $(\mathbf{x}_{t_{n+1}}, t_{n+1})$ and can be estimated through an Euler method starting from $(\mathbf{x}_{t_{n+1}}, t_{n+1})$ with a pre-trained $\mathbf{s}_\phi$. $\lambda(\cdot) \in \mathbb{R}^+$ is a positive weighting function and $\bar{\boldsymbol{\theta}}$ denotes a running average of the past values of $\boldsymbol{\theta}$.

To see why $\mathcal{L}_{\text{CD}}$ works, assume that $\mathbf{f}_{\boldsymbol{\theta}}$ is well-trained and $\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_{t_n}, t_n) = \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_{t_{n+1}}, t_{n+1})$ for $n \in \{1, 2, \ldots, N-1\}$. Applying the equality recursively starting from t_1 yields $\mathbf{x}_0 = \mathbf{f}_\theta(\mathbf{x}_{t_1}, t_1) = \cdots = \mathbf{f}_\theta(\mathbf{x}_{t_N}, t_N)$. Thus, by minimizing $\mathcal{L}_{\text{CD}}$, $\mathbf{f}_\theta(\mathbf{x}, t)$ is trained to return the origin $\mathbf{x}_0$ of the PF-ODE trajectory for all $(\mathbf{x}, t)$ along the trajectory. Then, by sampling from $\tilde{\mathbf{x}}_T \sim \mathcal{N}(\mathbf{0}, T^2\mathbf{I})$ and evaluating $\mathbf{f}_\theta(\mathbf{x}_T, T)$, CM generates an approximate sample of p_{data} in one step.

We would like to note that fast sampling of CMs comes with a trade-off in output quality since the pre-trained PF-ODE model cannot be perfectly distilled in general. Additionally, the performance of CMs heavily depends on the quality of the pre-trained PF-ODE model, emphasizing the significance of a well-trained model for achieving desirable results. In the subsequent section, we will demonstrate that the performance of CMs can be enhanced by incorporating an adversarial training setting. This approach not only improves the performance of CMs but also alleviates concerns regarding the imperfect training of the PF-ODE model.

3. Approach

In this section, we introduce a method that can serve as both a technique to enhance the training stability of GANs and improve the performance of CMs. The approach assumes the accessibility to a pre-trained PF-ODE model (not necessarily to be perfectly trained), which will serve as a prototype of the generator $\mathcal{G}$ (from the view of stabilizing GAN's training) or the model to be distilled (from the view of enhancing CMs). To emphasize the reliance on the consistency constraint in CMs, we name our approach CM-GAN. We will begin by presenting our method as a fine-tuning technique for CMs, which provides a clearer understanding and stronger motivation for our work.[2]

Consider the distillation process of CMs that minimizes $\mathcal{L}_{\text{CD}}$ in (4). Due to a possibly imperfect training of CM and the pretrained PF-ODE model, $\mathbf{f}_\theta$ could not output a good enough approximate sample of p_{data}. To fix this issue, we can adopt a GAN structure by simultaneously training a discriminator $\mathcal{D}$ to correct the outputs of $\mathbf{f}_\theta(\mathbf{x}_{t_n}, t_n)$ for $\mathbf{x}_{t_n} \sim p_t(\mathbf{x}_{t_n})$ and $n \sim \mathcal{U}[\![1, N-1]\!]$. In this way, the error signal from $\mathcal{D}$ guides $\mathbf{f}_\theta$ to produce more realistic outputs while the consistency constraints regularize the corrected output to stay in the neighbour of the one induced by the PF-ODE (the ground truth in the distillation of CMs).

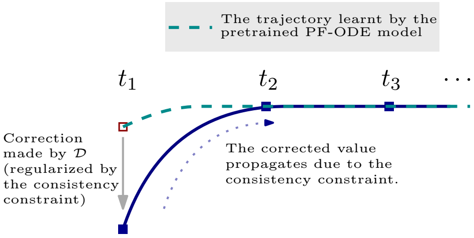
The trajectory learnt by the pretrained PF-ODE model
t_1
t_2
t_3
$\cdots$
Correction made by $\mathcal{D}$ (regularized by the consistency constraint)
The corrected value propagates due to the consistency constraint.

Figure 1: Discriminator $\mathcal{D}$ corrects the outputs of generator $\mathcal{G}$ while the consistency constraint ensures that the corrected output stays close to the one induced by the PF-ODE.

To see how discriminator $\mathcal{D}$ helps the training of $\mathbf{f}_\theta$, consider the training dynamic involving the time step $t_1 = 0$ (see Fig 1). The consistency constraint $\|\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_{t_2}, t_2) - \mathbf{f}_{\bar{\boldsymbol{\theta}}}(\mathbf{x}_{t_1}, t_1)\|^2$ enforces $\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_{t_2}, t_2)$ to stay close to the origin of the PF-ODE trajectory $\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_{t_1}, t_1)$ while $\mathcal{D}$ provides additional correction signal to make $\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_{t_2}, t_2)$ be more realistic. We apply this idea recursively and obtain the following training objective:

$$\min_{\mathbf{f}_{\boldsymbol{\theta}}} \max_{\mathcal{D}} \mathbb{E}\Big[\log \mathcal{D}(\mathbf{x})\Big] + \mathbb{E}\Big[\log\Big(1 - \mathcal{D}\big(\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_{t_n}, t_n)\big)\Big)\Big] + \mathcal{L}_{\text{CD}}(\boldsymbol{\theta}, \bar{\boldsymbol{\theta}}) \tag{5}$$

where $\mathbf{x} \sim p_{\text{data}}(\mathbf{x})$, $n \sim \mathcal{U}[\![1, N-1]\!]$, $\mathbf{x}_t \sim p_t(\mathbf{x}_t)$. The weighting function is set to

$$\lambda(t_n; \alpha, \gamma, \mathcal{I}) = \exp\Big(\alpha(t_i - T) \cdot (1 - \gamma^{\mathcal{I}})\Big), \tag{6}$$

where $\alpha \geq 0$, $\gamma \in [0, 1]$ and $\mathcal{I}$ denotes the number of training iterations so far. Here α controls the overall strength of the consistency constraints. For a close-to-zero α, the weights approach one, injecting a stronger consistency constraint. This selection enhances training stability for the generator, although it comes at the cost of limiting its ability to refine outputs by incorporating error correction signals from the discriminator. Conversely, increasing the value of α provides the generator with greater flexibility to enhance its performance, which however comes at the expense of losing training stability. (In Section 4, we provide evidence of a sweet spot where the optimal balance between flexibility and stability can be achieved.)

Additionally, the weighting function progressively reduces the emphasis on consistency constraints as training proceeds (with the transition speed controlled by γ). As a result, it allows the initial stage of training to resemble that of CMs, enabling fast learning and ensuring stability, and provides the generator with greater flexibility to refine its outputs as training advances. To further improve training stability, we adopt a weight change mechanism that gradually slows down as t_i approaches T (and for $t_i = T$, the weight equals one throughout the training).

[2] Applying the method on a trained CM is not a requirement. Instead, it can be incorporated from the initial stages of CM training as demonstrated in Section 4.

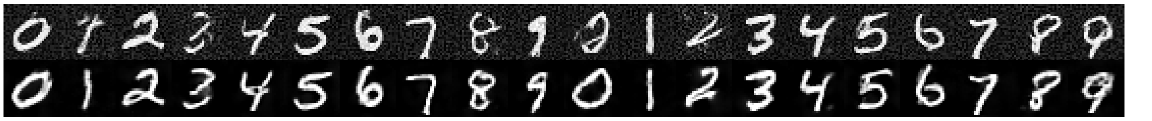

Figure 2: The images generated by CM (first row) and CM-GAN (second row).

The proposed approach can also be seen as a method to stabilize the training of GAN. In particular, the approach utilizes a generator $\mathcal{G}$ that has the same architecture as CM's (Song et al., 2023), where $\mathcal{G}(\boldsymbol{\epsilon}) = \mathbf{f}_\theta(\boldsymbol{\epsilon}, T)$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, T^2\mathbf{I})$. Apart from the main generation task, $\mathcal{G}$ is also trained to complete a sequence of auxiliary denoising tasks with various levels of noise added. The outputs are then regularized by consistency constraints in combination with a pre-trained PF-ODE model. In this way, the pre-trained model serves as a prototype of $\mathcal{G}$ where the consistency constraints require $\mathcal{G}(\boldsymbol{\epsilon})$ to be close to $\tilde{\mathbf{x}}_0(\boldsymbol{\epsilon})$, the origin of the PF-ODE trajectory ending with $(\boldsymbol{\epsilon}, T)$. This requirement excludes the possibility of the generator fooling the discriminator by utilizing a single most plausible sample for all input $\boldsymbol{\epsilon}$. Instead, the generator is compelled to generate distinct and appropriate outputs for different $\boldsymbol{\epsilon}$ to meet the additional closeness constraint. Consequently, this approach alleviates the mode collapse problem, ensuring a more diverse set of generated samples. Moreover, the introduced consistency constraints discourage the generator from blindly following the error signal provided by the discriminator. Thereby, training stability is improved, as the generator is less likely to be swayed arbitrarily by the discriminator's feedback.

4. Experiments

In this section, we empirically demonstrate the effectiveness of CM-GAN on the MNIST dataset (LeCun et al., 2010).

Experimental Settings and implementation. For the generator $\mathcal{G}$, we adopt the generator architecture (U-Net) from Song et al. (2023). Our implementation is based on the consistency distillation method of CMs (Song et al., 2023), where we use a pretrained EDM diffusion model (Karras et al., 2022) as the pretrained PF-ODE model.

To establish an adversarial training setup, we adopt the training objective in (5). Unless otherwise specified, the weighting function hyperparameters $\gamma = 0.99995$ and $\alpha = 0.025$. The discriminator is built upon the ResNet-18 architecture, utilizing only the first two blocks.

CM-GAN improves the performance of CM. Our discussion in Section 3 suggests that CM-GAN can be seen as a fine-tuning method to boost the performance of the CM models as the discriminator enforces the CM model to generate samples toward the true data distribution. In Fig 2, we present the outputs of CM (first row) and CM-GAN (second row) with the shared input $\boldsymbol{\epsilon}$ for each column. The figure shows that CM-GAN can effectively correct out-of-distribution samples. Additionally, due to the consistency constraints, the outputs of the two models are expected to be similar given the same input. Indeed, in Fig 2, we observe that CM-GAN successfully enhances image quality while largely preserving the distinct style of the numbers.

CM-GAN stabilizes the training of GAN. In Section 3, we mentioned that, from the perspective of GAN, CM-GAN is expected to stabilize the training process and there is supposed to be a sweet spot for α (defined in (6)). In Fig 3, we present the outputs of CM-GAN for different selections of α. It is observed that for small values of α ($\alpha = 0.005, 0.01$), the strong consistency constraints prioritize similarity to the pretrained EDM model, limiting the generator's flexibility to refine its outputs based on the discriminator's feedback. On the other hand, when $\alpha = 0.25$, the regularization applied to the generator becomes too weak, leading to an unstable training dynamic and a decrease in image quality. (In Appendix A, we show the dynamics become further unstable when training the model in a pure GAN setting.) The best performance is observed when setting $\alpha = 0.025$, which strikes an optimal balance between the generator's training flexibility and stability.

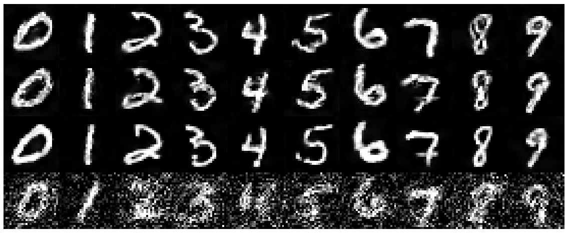

Figure 3: The images generated by CM-GAN with different choices of α. From row 1-4: $\alpha = 0.005, 0.01, 0.025, 0.25$.

5. Conclusion

In this paper, we presented CM-GAN, a technique that enhances the training stability of GANs while also acting as a fine-tuning method for CMs. Preliminary empirical study was conducted using the MNIST dataset to demonstrate its effectiveness. Our future work aims to enhance CM-GAN by better leveraging the consistency constraints to eliminate the need of a pretrained PF-ODE model. We also plan to evaluate CM-GAN on other datasets and applications.

References

Arjovsky, M. and Bottou, L. Towards principled methods for training generative adversarial networks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Hk4_qw5xe.

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 214–223. PMLR, 06–11 Aug 2017a. URL https://proceedings.mlr.press/v70/arjovsky17a.html.

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 214–223. PMLR, 06–11 Aug 2017b. URL https://proceedings.mlr.press/v70/arjovsky17a.html.

Brock, A., Donahue, J., and Simonyan, K. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=B1xsqj09Fm.

Croitoru, F.-A., Hondru, V., Ionescu, R. T., and Shah, M. Diffusion models in vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–20, 2023.

Donahue, J., Krähenbühl, P., and Darrell, T. Adversarial feature learning. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=BJtNZAFgg.

Dumoulin, V., Belghazi, I., Poole, B., Lamb, A., Arjovsky, M., Mastropietro, O., and Courville, A. Adversarially learned inference. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=B1ElR4cgg.

Fedus, W., Rosca, M., Lakshminarayanan, B., Dai, A. M., Mohamed, S., and Goodfellow, I. Many paths to equilibrium: GANs do not need to decrease a divergence at every step. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=ByQpn1ZA-.

Goodfellow, I. Neurips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

Ho, J., Salimans, T., Gritsenko, A. A., Chan, W., Norouzi, M., and Fleet, D. J. Video diffusion models. In *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=f3zNgKga_ep.

Karras, T., Laine, S., and Aila, T. A style-based generator architecture for generative adversarial networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 43 (12):4217–4228, dec 2021. ISSN 0162-8828. doi: 10.1109/TPAMI.2020.2970919. URL https://doi.org/10.1109/TPAMI.2020.2970919.

Karras, T., Aittala, M., Aila, T., and Laine, S. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems*, 2022.

Kong, Z., Ping, W., Huang, J., Zhao, K., and Catanzaro, B. Diffwave: A versatile diffusion model for audio synthesis. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=a-xFK8Ymz5J.

LeCun, Y., Cortes, C., and Burges, C. Mnist handwritten digit database. *ATT Labs [Online]*, 2, 2010. URL http://yann.lecun.com/exdb/mnist.

Li, C., Liu, H., Chen, C., Pu, Y., Chen, L., Henao, R., and Carin, L. Alice: Towards understanding adversarial learning for joint distribution matching. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017a. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/ade55409d1224074754035a5a937d2e0-Paper.pdf.

Li, C.-L., Chang, W.-C., Cheng, Y., Yang, Y., and Poczos, B. Mmd gan: Towards deeper understanding of moment matching network. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017b. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/dfd7468ac613286cdbb40872c8ef3b06-Paper.pdf.

Mescheder, L., Nowozin, S., and Geiger, A. The numerics of gans. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/4588e674d3f0faf985047d4c3f13ed0d-Paper.pdf.

Mescheder, L., Geiger, A., and Nowozin, S. Which training methods for GANs do actually converge? In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3481–3490. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/mescheder18a.html.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=B1QRgziT-.

Nowozin, S., Cseke, B., and Tomioka, R. f-gan: Training generative neural samplers using variational divergence minimization. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/file/cedebb6e872f539bef8c3f919874e9d7-Paper.pdf.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695, June 2022.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 2256–2265, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/sohl-dickstein15.html.

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=St1giarCHLP.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=PxTIG12RRHS.

Song, Y., Dhariwal, P., Chen, M., and Sutskever, I. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023.

Yang, D., Yu, J., Wang, H., Wang, W., Weng, C., Zou, Y., and Yu, D. Diffsound: Discrete diffusion model for text-to-sound generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:1720–1733, 2023. doi: 10.1109/TASLP.2023.3268730.

Zheng, H. and Zhou, M. Exploiting chain rule and bayes' theorem to compare probability distributions. In *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=f-ggKIDTu5D.

A. Additional empirical results

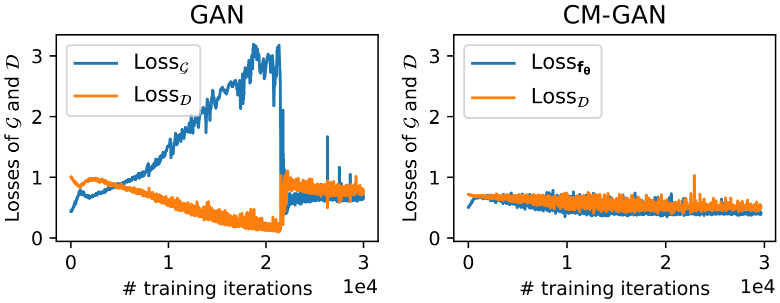

Figure 4: The dynamics of the losses associated with the generator $\mathcal{G}$ (or $\mathbf{f}_{\boldsymbol{\theta}}$ in CM-GAN) and the discriminator $\mathcal{D}$.

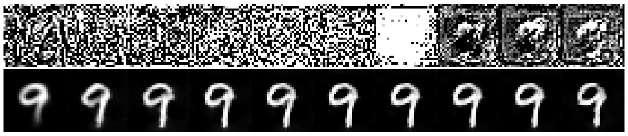

Figure 5: The images generated by GAN (first row) and CM-GAN (second row) as the training proceeds. The images are sampled every 3K training iterations until 30K iterations.

In Section 4, we show that overly weak consistency constraints fail to offer enough regularization to stabilize the training dynamics. In order to inject the consistency constraints and present how its strength affects the generator's performance, the generators have to perform auxiliary multi-scale denoising tasks, which is different from the regular GANs setting.

In order to make our results more convincing, we provide additional empirical evidence with a classical GAN configuration that is not equipped with the auxiliary denoising tasks. (The generator of GAN has the same architecture as CM-GAN's and it takes an input $\boldsymbol{\epsilon}$ that has the same size as the output.)

Fig 4 plots the dynamics of the losses associated with the generator $\mathcal{G}$ (or $\mathbf{f}_{\boldsymbol{\theta}}$ in CM-GAN) and the discriminator $\mathcal{D}$. In particular, for GAN,

$$\mathrm{loss}_{\mathcal{G}} = \mathbb{E}_{\mathbf{x}\sim p_{\mathrm{data}}}\Big[\log \mathcal{D}(\mathbf{x})\Big] + \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},T^2\mathbf{I})}\Big[\log\big(1-\mathcal{D}(\mathcal{G}(\boldsymbol{\epsilon}))\big)\Big] \tag{7}$$

and

$$\mathrm{loss}_{\mathcal{D}} = \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},T^2\mathbf{I})}\Big[\log\big(1-\mathcal{D}(\mathcal{G}(\boldsymbol{\epsilon}))\big)\Big]. \tag{8}$$

For CM-GAN, it involves multi-scale denoising, and its losses become

$$\mathrm{loss}_{\mathbf{f}_{\boldsymbol{\theta}}} = \mathbb{E}_{\mathbf{x}\sim p_{\mathrm{data}}}\Big[\log \mathcal{D}(\mathbf{x})\Big] + \mathbb{E}_{\mathbf{x}\sim p_{\mathrm{data}},\,\mathbf{x}_{t_n}\sim p_{t_n},\,n\sim\mathcal{U}[\![1,N-1]\!]}\Big[\log\Big(1-\mathcal{D}\big(\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_{t_n},t_n)\big)\Big)\Big] \tag{9}$$

and

$$\mathrm{loss}_{\mathcal{D}} = \mathbb{E}_{\mathbf{x}\sim p_{\mathrm{data}},\,\mathbf{x}_{t_n}\sim p_{t_n},\,n\sim\mathcal{U}[\![1,N-1]\!]}\Big[\log\Big(1-\mathcal{D}\big(\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}_{t_n},t_n)\big)\Big)\Big]. \tag{10}$$

A large $\mathrm{loss}_{\mathcal{G}}$ (or $\mathrm{loss}_{\mathbf{f}_{\boldsymbol{\theta}}}$) indicates the discriminator is overly strong and the generator cannot find a way to fool it and thus cannot be effectively optimized. In contrast, a large $\mathrm{loss}_{\mathcal{D}}$ suggests that the discriminator is too weak to provide useful error signals that guide the generator to refine its outputs. In practice, we desire both $\mathrm{loss}_{\mathcal{G}}$ (or $\mathrm{loss}_{\mathbf{f}_{\boldsymbol{\theta}}}$) and $\mathrm{loss}_{\mathcal{D}}$ to stay at intermediate values to enable a successful training.

From Fig 4, we observe that the training of GAN is very unstable, especially for the first 25K iterations. As a result, the generator cannot produce recognizable digits as presented in Fig 5. We argue that the superficially stabler training dynamics since 25K iterations were largely due to the gradient vanishing problem. As suggested by the large $\mathrm{loss}_{\mathcal{G}}$ (and small $\mathrm{loss}_{\mathcal{D}}$),

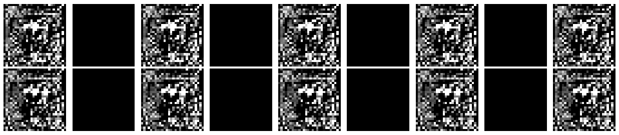

Figure 6: The odd columns plot the images generated by GAN with two randomly picked inputs ϵ (the first and second rows, respectively) for different numbers of training iterations. From left to right, the images were sampled every 1K training iterations, starting from iteration 26K and continuing until iteration 30K. The even columns display the differences between the images on their two sides. The close-to-zero differences (as suggested by the black patches) indicate that the GAN's outputs are nearly unchanged.

the discriminator is overly strong in the first 22K training iterations. The unmatched performances of the generator $\mathcal{G}$ and discriminator $\mathcal{D}$ result in the overfitting of $\mathcal{D}$, making it easier for $\mathcal{G}$ to find outputs that can deceive $\mathcal{D}$ and balance their losses (during the 22K-26K iterations). However, the overfitted $\mathcal{D}$ cannot provide strong enough error gradients to refine the outputs of $\mathcal{G}$. As a result, the outputs of $\mathcal{G}$ remain largely unchanged since 26K iterations as shown in Fig 6.

In contrast, the training of CM-GAN is fairly stable throughout the entire process, and the generator consistently produces high quality images.