Conditioned Generation and Speed-Up Diffusion Models

Presented by: Yue Yu[†]

[†]: Statistics Club, Indiana University

Tentative Schedule of Diffusion Model Series

- **10/10**: Overview of generative AI models in Computer Vision, including GANs (Goodfellow et al., 2014), VAEs (Kingma, 2013) and diffusion models (Sohl-Dickstein et al., 2015), and detailed introduction to GANs;
- **10/17**: Detailed introduction to Autoencoders;
- **10/24**: DDPM (Ho et al., 2020), the cornerstone paper about diffusion models, which enables diffusion models to produce high-quality, realistic samples and to be competitive with generative models like GANs and VAEs;

Tentative Schedule of Diffusion Model Series (Cont')

- **11/7**: DDIM (Song et al., 2020a), which presents a method to speed up the sampling process in diffusion models without sacrificing much in terms of sample quality, and guided diffusion models (Dhariwal and Nichol, 2021), (Ho and Salimans, 2022), which enhances the flexibility and controllability of generated samples, and are often used in tasks requiring conditional generation, such as text-to-image synthesis;
- **11/21**: Recent years' improvements and applications of diffusion models from different aspects, e.g., Latent Diffusion Models (Stable Diffusion, Rombach et al. (2022)) and its subsequent works like conditional control to text-to-image diffusion (Zhang et al., 2023) and Stable Diffusion XL (Podell et al., 2023).

Today's Presentation Outline

- Noise-conditioned Score Networks;
- Classifier Guided Diffusion;
- Classifier-Free Guidance;
- Faster Sampling Steps.

Notation (1/2)

- $\mathbf{x}_{s<t}$: Intermediate sample in the accelerated sampling process for steps $s < t$.
- $q_{\sigma,s<t}(\mathbf{x}_s|\mathbf{x}_t, \mathbf{x}_0)$: Probability distribution for the accelerated trajectory with subset steps $s < t$.
- η: Hyperparameter controlling stochasticity in the denoising diffusion implicit model (DDIM).
- σ_t: Standard deviation used in the accelerated sampling process, $\sigma_t = \eta \cdot \tilde{\beta}_t$.
- S: Reduced number of steps in the strided sampling schedule.
- $\mathbf{z}_{i/k}$: Intermediate latent variable in progressive distillation, where i/k represents fractional steps.
- $\mathbf{x}_\epsilon$: Consistent mapping of a noisy data point $\mathbf{x}_t$ back to the origin in a consistency model.

Notation (2/2)

- $f(\mathbf{x}_t, t)$: Consistency function that maps noisy data $\mathbf{x}_t$ at step t to a target.
- $c_{\text{skip}}(t)$: Scaling function for the skip connection in the consistency model, where $c_{\text{skip}}(\epsilon) = 1$.
- $c_{\text{out}}(t)$: Scaling function for the output in the consistency model, where $c_{\text{out}}(\epsilon) = 0$.
- $F_\theta(\mathbf{x}, t)$: Neural network function parameterized by θ to predict the noise component at step t.
- $f_\theta(\mathbf{x}, t, y)$: Score function in classifier-free guidance, parameterized by θ, conditioned on class label y.
- $\mathcal{N}(\mu, \sigma^2\mathbf{I})$: Normal distribution with mean μ and variance $\sigma^2\mathbf{I}$.
- $f_\theta(\mathbf{x}, t) = c_{\text{skip}}(t)\mathbf{x} + c_{\text{out}}(t)F_\theta(\mathbf{x}, t)$: Parameterized form of the consistency function.

Diffusion Model's Connection with Stochastic Gradient Langevin Dynamics

- Langevin dynamics is a concept from physics, developed for statistically modeling molecular systems. Combined with stochastic gradient descent, **stochastic gradient Langevin dynamics** (Welling and Teh, 2011) can produce samples from a probability density $p(\mathbf{x})$ using only the gradients $\nabla_{\mathbf{x}} \log p(\mathbf{x})$ in a Markov chain of updates:

$$\mathbf{x}_t = \mathbf{x}_{t-1} + \frac{\delta}{2} \nabla_{\mathbf{x}} \log p(\mathbf{x}_{t-1}) + \sqrt{\delta} \epsilon_t, \quad \text{where } \epsilon_t \sim \mathcal{N}(0, \mathbf{I})$$

  where δ is the step size. When $T \to \infty, \epsilon \to 0$, $\mathbf{x}_T$ equals to the true probability density $p(\mathbf{x})$.

- Compared to standard SGD, stochastic gradient Langevin dynamics injects Gaussian noise into the parameter updates to avoid collapse into local minima.

Score-Based Generative Modeling with Langevin Dynamics

- Song and Ermon (2019) proposed a score-based generative modeling method where samples are produced via **Langevin dynamics** using gradients of the data distribution estimated with score matching. The score of each sample $\mathbf{x}$'s density probability is defined as its gradient $\nabla_{\mathbf{x}} \log q(\mathbf{x})$. A score network $\mathbf{s}_\theta : \mathbb{R}^D \to \mathbb{R}^D$ is trained to estimate it, $\mathbf{s}_\theta(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log q(\mathbf{x})$.

- To make it scalable with high-dimensional data in the deep learning setting, they proposed to use either **denoising score matching** (Vincent, 2011) or **sliced score matching** (use random projections) (Song et al., 2020b). Denoising score matching adds a pre-specified small noise to the data $q(\tilde{\mathbf{x}}|\mathbf{x})$ and estimates $q(\tilde{\mathbf{x}})$ with score matching.

Langevin Dynamics and the Manifold Hypothesis

- Langevin dynamics samples data points from a probability density distribution using the score $\nabla_{\mathbf{x}} \log q(\mathbf{x})$ iteratively.
- According to the manifold hypothesis, data is often concentrated in a low-dimensional manifold, limiting score estimation in sparse regions and reducing reliability.
- Adding Gaussian noise helps extend the data distribution across $\mathbb{R}^D$, stabilizing score estimation. Song and Ermon (2019) further improved this by using noise at different levels and training a noise-conditioned score network to jointly estimate scores across varying noise levels.

Diffusion Process and Score Approximation

- Increasing noise levels resembles the forward diffusion process.
- Using diffusion process notation, the score approximates $s_\theta(\mathbf{x}_t, t) \approx \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t)$.
- For Gaussian distribution $\mathbf{x} \sim \mathcal{N}(\mu, \sigma^2 \mathbf{I})$:

$$\nabla_{\mathbf{x}} \log p(\mathbf{x}) = -\frac{\mathbf{x} - \mu}{\sigma^2} = -\frac{\epsilon}{\sigma} \quad \text{where} \quad \epsilon \sim \mathcal{N}(0, \mathbf{I})$$

- Recall that $q(\mathbf{x}_t | \mathbf{x}_0) \sim \mathcal{N}(\sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1 - \bar{\alpha}_t)\mathbf{I})$, hence:

$$s_\theta(\mathbf{x}_t, t) \approx \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t) = \mathbb{E}_{q(\mathbf{x}_0)} \left[-\frac{\epsilon_\theta(\mathbf{x}_t, t)}{\sqrt{1 - \bar{\alpha}_t}} \right]$$

Conditioned Generation

- Generative models are often trained on images with conditioning information, like the ImageNet dataset.
- Common conditioning methods include class labels, input text or images.
- This allows generating samples aligned with specific classes or descriptions.

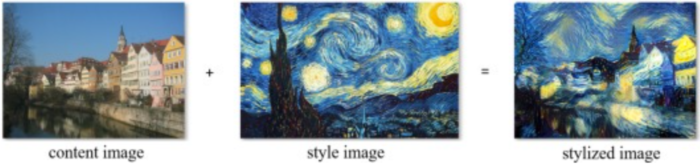


Figure: Example of conditional generation. After conditioned on the style of Van Gogh's *The Starry Night*, the image becomes a stylized version (Hua and Chen, 2022).

Classifier Guided Diffusion

- To incorporate class information, Dhariwal and Nichol (2021) trained a classifier $f_\phi(y|\mathbf{x}_t, t)$ on noisy images $\mathbf{x}_t$.
- The gradient $\nabla_{\mathbf{x}_t} \log f_\phi(y|\mathbf{x}_t)$ guides the diffusion process toward conditioning information y (e.g., a target class).
- Recall:

$$\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t) = -\frac{1}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(\mathbf{x}_t, t)$$

- Score function for joint distribution $q(\mathbf{x}_t, y)$:

$$\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t, y) \approx -\frac{1}{\sqrt{1-\bar{\alpha}_t}} \left(\epsilon_\theta(\mathbf{x}_t, t) - \sqrt{1-\bar{\alpha}_t} \nabla_{\mathbf{x}_t} \log f_\phi(y|\mathbf{x}_t) \right)$$

Classifier-Guided Predictor Adjustment

- A classifier-guided predictor $\bar{\epsilon}_\theta$ takes the form:

$$\bar{\epsilon}_\theta(\mathbf{x}_t, t) = \epsilon_\theta(\mathbf{x}_t, t) - \sqrt{1 - \bar{\alpha}_t}\nabla_{\mathbf{x}_t} \log f_\phi(y|\mathbf{x}_t)$$

- To adjust classifier guidance strength, introduce weight w to the delta term:

$$\bar{\epsilon}_\theta(\mathbf{x}_t, t) = \epsilon_\theta(\mathbf{x}_t, t) - \sqrt{1 - \bar{\alpha}_t}\, w\nabla_{\mathbf{x}_t} \log f_\phi(y|\mathbf{x}_t)$$

- Resulting models in Dhariwal and Nichol (2021):
  - Ablated diffusion model (ADM)
  - Classifier-guided model (ADM-G) – achieves superior results over SOTA models like BigGAN.

U-Net Modifications for Improved Diffusion Performance

- Dhariwal and Nichol (2021) demonstrated improved performance over GANs with modifications to the U-Net architecture in diffusion models.
- Key architectural changes:
  - Increased model depth and width;
  - Additional attention heads and multi-resolution attention;
  - BigGAN residual blocks for up/downsampling;
  - Residual connection rescale by $1/\sqrt{2}$;
  - Adaptive group normalization (AdaGN), which normalizes feature maps by dividing them into groups and normalizing each group separately.

Classifier-Free Guidance

- Conditional diffusion steps can be achieved without an independent classifier f_ϕ by combining scores from a conditional and unconditional diffusion model Ho and Salimans (2022).
- Define:
  - Unconditional model $p_\theta(\mathbf{x})$ with score estimator $\epsilon_\theta(\mathbf{x}_t, t)$
  - Conditional model $p_\theta(\mathbf{x}|y)$ with score estimator $\epsilon_\theta(\mathbf{x}_t, t, y)$
- Both models are trained via a single neural network. In training, conditioning y is randomly dropped to allow both conditional and unconditional generation:

$$\epsilon_\theta(\mathbf{x}_t, t) = \epsilon_\theta(\mathbf{x}_t, t, y = \varnothing)$$

Classifier-Free Guidance: Modified Score

- The gradient of an implicit classifier can be represented using conditional and unconditional score estimators.
- In the modified score, dependency on a separate classifier is removed:

$$\nabla_{\mathbf{x}_t} \log p(y|\mathbf{x}_t) = \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|y) - \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t)$$

$$= -\frac{1}{\sqrt{1-\bar{\alpha}_t}} \left(\epsilon_\theta(\mathbf{x}_t, t, y) - \epsilon_\theta(\mathbf{x}_t, t)\right)$$

- New classifier-free predictor:

$$\bar{\epsilon}_\theta(\mathbf{x}_t, t, y) = (w+1)\epsilon_\theta(\mathbf{x}_t, t, y) - w\epsilon_\theta(\mathbf{x}_t, t)$$

- Results: Classifier-free guidance achieves a good balance between:
  - FID (distinguishes synthetic from generated images)
  - IS (quality and diversity of images)

GLIDE: Guided Diffusion Model Strategies

- GLIDE Nichol and Dhariwal (2021) explored two guidance strategies:
  - CLIP guidance
  - Classifier-free guidance
- Findings:
  - Classifier-free guidance is generally preferred.
  - Hypothesis: CLIP guidance may lead to adversarial examples, optimizing for CLIP alignment rather than generating well-matched images.

Speed Concerns in DDPM (Ho et al., 2020)

- Generating samples from DDPMs is slow due to the need to follow a Markov chain with up to thousands of steps.
- Example from Song et al. (2020c):

  "It takes around 20 hours to sample 50k images of size 32×32 from a DDPM, but less than a minute to do so from a GAN on an Nvidia 2080 Ti GPU."
- This highlights the challenge of improving sampling speed for practical applications.

Speeding Up Diffusion Sampling

- **Strided Sampling Schedule** Nichol and Dhariwal (2021):
  - Reduce sampling steps from T to S by updating every $\lceil T/S \rceil$ steps.
  - New schedule: $\{\tau_1, \dots, \tau_S\}$ where $\tau_1 < \tau_2 < \cdots < \tau_S \in [1, T]$ and $S < T$.
- Reparameterization with Desired Standard Deviation σ_t:
  - Rewrite $q_\sigma(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0)$ to use a "nice property":

$$\mathbf{x}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2}\, \epsilon_\theta^{(t)}(\mathbf{x}_t) + \sigma_t \epsilon$$

  - Formulation of $q_\sigma(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0)$:

$$q_\sigma(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_{t-1}; \sqrt{\bar{\alpha}_{t-1}} \left(\frac{\mathbf{x}_0 - \sqrt{1 - \bar{\alpha}_t}\, \epsilon_\theta^{(t)}(\mathbf{x}_t)}{\sqrt{\bar{\alpha}_t}} \right), \sigma_t^2 \mathbf{I} \right)$$

  - where $\epsilon_\theta^{(t)}(\cdot)$ predicts ϵ_t from $\mathbf{x}_t$.

Controlling Stochasticity in DDIM

- Recall that in $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\mu}(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I})$, we have:

$$\tilde{\beta}_t = \sigma_t^2 = \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} \cdot \beta_t$$

- Define $\sigma_t^2 = \eta \cdot \tilde{\beta}_t$ with $\eta \in \mathbb{R}^+$ as a hyperparameter to control sampling stochasticity.
- Special case: $\eta = 0$ leads to a deterministic process, known as the *Denoising Diffusion Implicit Model (DDIM)* (Song et al., 2020a).
- DDIM Properties:
    - Maintains the same marginal noise distribution as the original model.
    - Deterministically maps noise to original data samples.

Accelerated Sampling with DDIM

- During generation, we don't need to follow the entire chain $t = 1, \ldots, T$, but can use a subset of steps.
- DDIM Update Step (for steps $s < t$):

$$q_{\sigma,s<t}(\mathbf{x}_s|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_s; \sqrt{\bar{\alpha}_s}\left(\frac{\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\,\epsilon_\theta^{(t)}(\mathbf{x}_t)}{\sqrt{\bar{\alpha}_t}}\right)\right. \\ \left. + \sqrt{1-\bar{\alpha}_s-\sigma_t^2}\,\epsilon_\theta^{(t)}(\mathbf{x}_t), \sigma_t^2\mathbf{I}\right)$$

- Observations:
  - All models are trained with $T = 1000$ diffusion steps.
  - **DDIM** $(\eta = 0)$: Best sample quality when S (subset size) is small.
  - **DDPM** $(\eta = 1)$: Performs better when $S = T = 1000$ (full chain).
- With DDIM, we can train with arbitrary forward steps but sample from a reduced set in generation.

Stochastic (DDPM, $\eta = 1$) vs Deterministic (DDIM, $\eta = 0$)

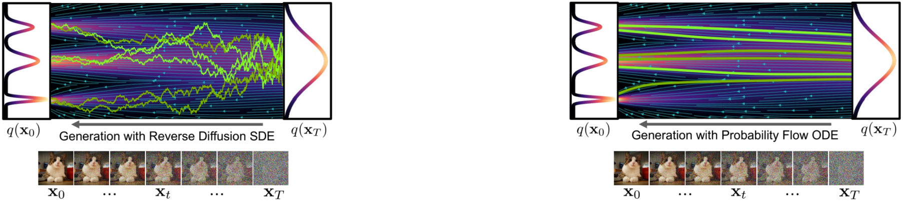
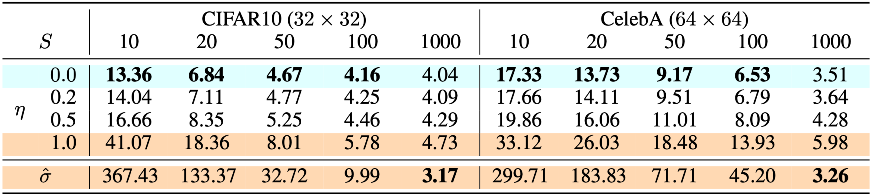

$q(\mathbf{x}_0)$
Generation with Reverse Diffusion SDE
$q(\mathbf{x}_T)$
$\mathbf{x}_0$ $\cdots$ $\mathbf{x}_t$ $\cdots$ $\mathbf{x}_T$
$q(\mathbf{x}_0)$
Generation with Probability Flow ODE
$q(\mathbf{x}_T)$
$\mathbf{x}_0$ $\cdots$ $\mathbf{x}_t$ $\cdots$ $\mathbf{x}_T$

Figure: Illustration of the differences between DDPM and DDIM. The key discrepancy is whether the reverse process is a stochastic differential equation (SDE) or probability flow ordinary differential equation (ODE) problem. Image source: Kreis et al. (2022)

FID Scores on CIFAR10 and CelebA with Different η



		CIFAR10 (32 × 32)					CelebA (64 × 64)				
	S	10	20	50	100	1000	10	20	50	100	1000
η	0.0	**13.36**	**6.84**	**4.67**	**4.16**	4.04	**17.33**	**13.73**	**9.17**	**6.53**	3.51
	0.2	14.04	7.11	4.77	4.25	4.09	17.66	14.11	9.51	6.79	3.64
	0.5	16.66	8.35	5.25	4.46	4.29	19.86	16.06	11.01	8.09	4.28
	1.0	41.07	18.36	8.01	5.78	4.73	33.12	26.03	18.48	13.93	5.98
	$\hat{\sigma}$	367.43	133.37	32.72	9.99	**3.17**	299.71	183.83	71.71	45.20	**3.26**

Figure: FID scores on CIFAR10 and CelebA datasets by diffusion models of different settings, including DDIM ($\eta = 0$) and DDPM ($\hat{\sigma}$).

Advantages of DDIM over DDPM

Compared to DDPM, DDIM is able to:

- Generate high-quality samples using a much fewer number of steps.
- Achieve a ”consistency” property since the generative process is deterministic, meaning that multiple samples conditioned on the same latent variable should have similar high-level features.
- Perform semantically meaningful interpolation in the latent variable due to the consistency property.

Progressive Distillation for Efficient Sampling

Progressive Distillation (Salimans and Ho, 2022) is a method for distilling trained deterministic samplers into new models with halved sampling steps.

- **Process**: The student model is initialized from the teacher model and denoises towards a target where one student DDIM step matches two teacher steps, instead of using the original sample $\mathbf{x}_0$ as the denoise target.
- **Iteration**: In each progressive distillation iteration, the sampling steps are halved, allowing for faster generation without significant loss in quality.
- **Goal**: This technique aims to retain sample quality while significantly reducing computation time.

Illustration of Progressive Distillation Framework

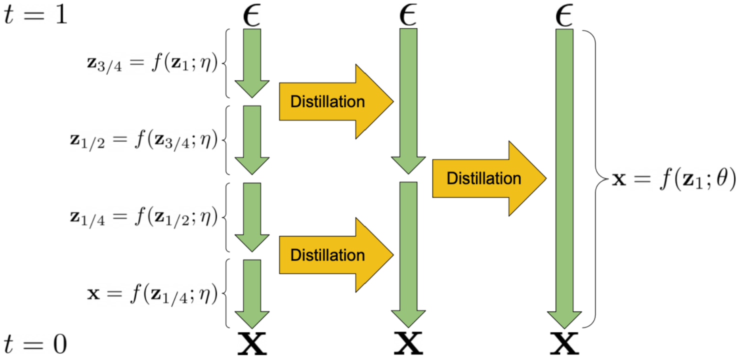
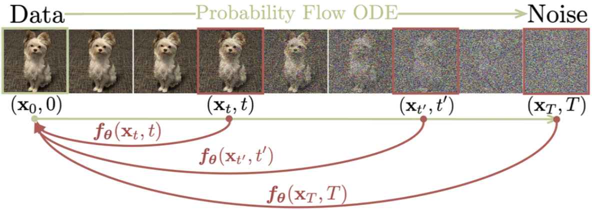

$t = 1$
$\mathbf{z}_{3/4} = f(\mathbf{z}_1; \eta)$
$\mathbf{z}_{1/2} = f(\mathbf{z}_{3/4}; \eta)$
$\mathbf{z}_{1/4} = f(\mathbf{z}_{1/2}; \eta)$
$\mathbf{x} = f(\mathbf{z}_{1/4}; \eta)$
$t = 0$
ϵ
ϵ
ϵ
Distillation
Distillation
Distillation
$\mathbf{x} = f(\mathbf{z}_1; \theta)$
$\mathbf{x}$
$\mathbf{x}$
$\mathbf{x}$

Figure: A visualization of two iterations of our proposed progressive distillation algorithm. A sampler $f(\mathbf{z}; \eta)$, mapping random noise ϵ to samples $\mathbf{x}$ in 4 deterministic steps, is distilled into a new sampler $f(\mathbf{z}; \theta)$ taking only one single step. The original sampler is derived by approximately integrating DDIM, and distillation can thus be understood as learning to integrate in fewer steps.

Consistency Models

- Consistency models (Song et al., 2023) learn to map any intermediate noisy data points $\mathbf{x}_t$, where $t > 0$, on the diffusion sampling trajectory directly back to its origin $\mathbf{x}_0$.
- Named for their self-consistency property, meaning any data points on the same trajectory are consistently mapped to the same origin.
- This approach ensures that noisy samples are accurately reversed to the original data, optimizing the diffusion process.


Data
Probability Flow ODE
Noise
$(\mathbf{x}_0, 0)$
$(\mathbf{x}_t, t)$
$(\mathbf{x}_{t'}, t')$
$(\mathbf{x}_T, T)$
$\boldsymbol{f}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$
$\boldsymbol{f}_{\boldsymbol{\theta}}(\mathbf{x}_{t'}, t')$
$\boldsymbol{f}_{\boldsymbol{\theta}}(\mathbf{x}_T, T)$

Consistency Function for Diffusion Models

- Given a trajectory $\{\mathbf{x}_t | t \in [\epsilon, T]\}$, the **consistency function** f is defined as:

$$f : (\mathbf{x}_t, t) \mapsto \mathbf{x}_\epsilon$$

  and satisfies $f(\mathbf{x}_t, t) = f(\mathbf{x}_{t'}, t') = \mathbf{x}_\epsilon$ for all $t, t' \in [\epsilon, T]$.

- When $t = \epsilon$, f acts as an identity function.
- The model is parameterized as:

$$f_\theta(\mathbf{x}, t) = c_{\text{skip}}(t)\mathbf{x} + c_{\text{out}}(t)F_\theta(\mathbf{x}, t)$$

  where $c_{\text{skip}}(\epsilon) = 1$ and $c_{\text{out}}(\epsilon) = 0$.

- This setup allows the model to generate samples in a single step, while preserving the option to trade off computational efficiency for improved quality through a multi-step sampling process.

Questions?

Thank you!

References I

P. Dhariwal and A. Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

J. Ho and T. Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

G. Hua and D. Chen. Deep conditional image generation: Towards controllable visual pattern modeling. In *Advanced Methods and Deep Learning in Computer Vision*, pages 191–219. Elsevier, 2022.

D. P. Kingma. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

References II

K. Kreis, R. Gao, and A. Vahdat. Denoising diffusion-based generative modeling: Foundations and applications. In *CVPR*, 2022.

A. Q. Nichol and P. Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021.

D. Podell, Z. English, K. Lacey, A. Blattmann, T. Dockhorn, J. Müller, J. Penna, and R. Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

T. Salimans and J. Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

References III

J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020a.

Y. Song and S. Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

Y. Song, S. Garg, J. Shi, and S. Ermon. Sliced score matching: A scalable approach to density and score estimation. In *Uncertainty in Artificial Intelligence*, pages 574–584. PMLR, 2020b.

Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020c.

References IV

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

P. Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

M. Welling and Y. W. Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688. Citeseer, 2011.

L. Zhang, A. Rao, and M. Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023.