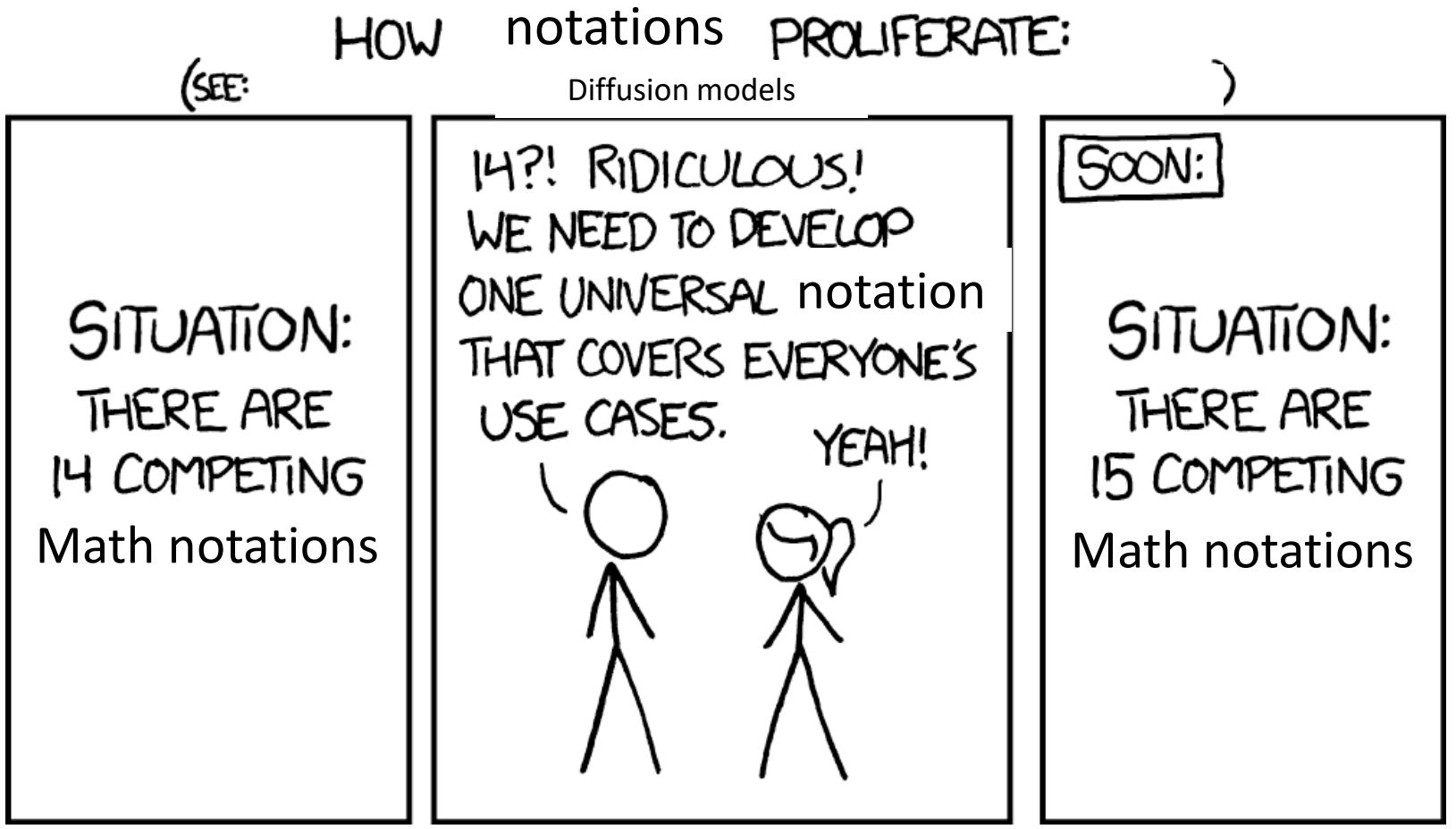
Diffusion Study Group #7

10/29/2022

Tanishq Abraham

Classifier-Free Diffusion Guidance

Differences in notation!



Standard classifier guidance

Standard conditional diffusion model:

$$\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{c}) \approx -\sigma_t \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t | \mathbf{c})$$

Classifier guidance modifies the score function:

$$\tilde{\boldsymbol{\epsilon}}_\theta(\mathbf{x}_t, \mathbf{c}) = \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{c}) - w\sigma_t \nabla_{\mathbf{x}_t} \log p(\mathbf{c} | \mathbf{x}_t)$$
$$\approx -\sigma_t \nabla_{\mathbf{x}_t} [\log p(\mathbf{x}_t | \mathbf{c}) + w \log p(\mathbf{c} | \mathbf{x}_t)]$$

where w is the guidance scale. This corresponds to a distribution of:

$$\tilde{p}_\theta(\mathbf{x}_t, \mathbf{c}) \propto p_\theta(\mathbf{x}_t | \mathbf{c}) p_\theta(\mathbf{c} | \mathbf{x}_t)^w$$

This boosts the probability of generating images that are correctly classified by classifier $p_\theta(\mathbf{c} | \mathbf{x}_t)$

Demonstration of guidance scale

Figure 2: The effect of guidance on a mixture of three Gaussians, each mixture component representing data conditioned on a class. The leftmost plot is the non-guided marginal density. Left to right are densities of mixtures of normalized guided conditionals with increasing guidance strength.

Theoretical equivalence of unconditional and conditional models

Theoretically, applying classifier guidance with weight $w + 1$ to an unconditional model should be equivalent to applying classifier guidance to a conditional model with weight w

$$\begin{aligned}&\boldsymbol{\epsilon}_\theta(\mathbf{x}_t) - (w+1)\sigma_t \nabla_{\mathbf{x}_t} \log p_\theta(\,\mathbf{c} \mid \mathbf{x}_t\,) \\ &\approx -\sigma_t \nabla_{\mathbf{x}_t}[\log p(\mathbf{x}_t) + (w+1)\log p_\theta(\,\mathbf{c} \mid \mathbf{x}_t\,)]\end{aligned}$$

Bayes' Rule tells us:

$$\log p\,(\,\mathbf{x}_t \mid \mathbf{c}\,) = \log p\,(\,\mathbf{c} \mid \mathbf{x}_t\,) + \log p\,(\mathbf{x}_t) - \log p\,(\mathbf{c})$$

Therefore:

$$\boldsymbol{\epsilon}_\theta(\mathbf{x}_t) - (w+1)\sigma_t \nabla_{\mathbf{x}_t} \log p_\theta(\,\mathbf{c} \mid \mathbf{x}_t\,) \approx -\sigma_t \nabla_{\mathbf{x}_t}[\log p(\mathbf{x}_t \mid \mathbf{c}) + w \log p(\mathbf{c} \mid \mathbf{x}_t)]$$

Classifier-Free Diffusion Guidance

A conditional model $p_\theta(\mathbf{x}_t|\mathbf{c})$ (parameterized by $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{c})$) and an unconditional model $p_\theta(\mathbf{x}_t)$(parameterized by $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t)$)

Can be represented by a *single* neural network $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{c})$ where $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t) = \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{c} = \emptyset)$

During training, randomly set $\mathbf{c}$ to $\emptyset$ with some probability p_{uncond} (a new hyperparam)

Classifier-Free Diffusion Guidance

Let's examine Bayes' Rule again:

$$\log p\left(\mathbf{x}_t \mid \mathbf{c}\right) = \log p\left(\mathbf{c} \mid \mathbf{x}_t\right) + \log p\left(\mathbf{x}_t\right) - \log p\left(\mathbf{c}\right)$$
$$\log p\left(\mathbf{c} \mid \mathbf{x}_t\right) = \log p\left(\mathbf{x}_t \mid \mathbf{c}\right) + \log p\left(\mathbf{c}\right) - \log p\left(\mathbf{x}_t\right)$$
$$\nabla_{\mathbf{x}_t} \log p\left(\mathbf{c} \mid \mathbf{x}_t\right) = \nabla_{\mathbf{x}_t}[\log p\left(\mathbf{x}_t \mid \mathbf{c}\right) - \log p\left(\mathbf{x}_t\right)]$$

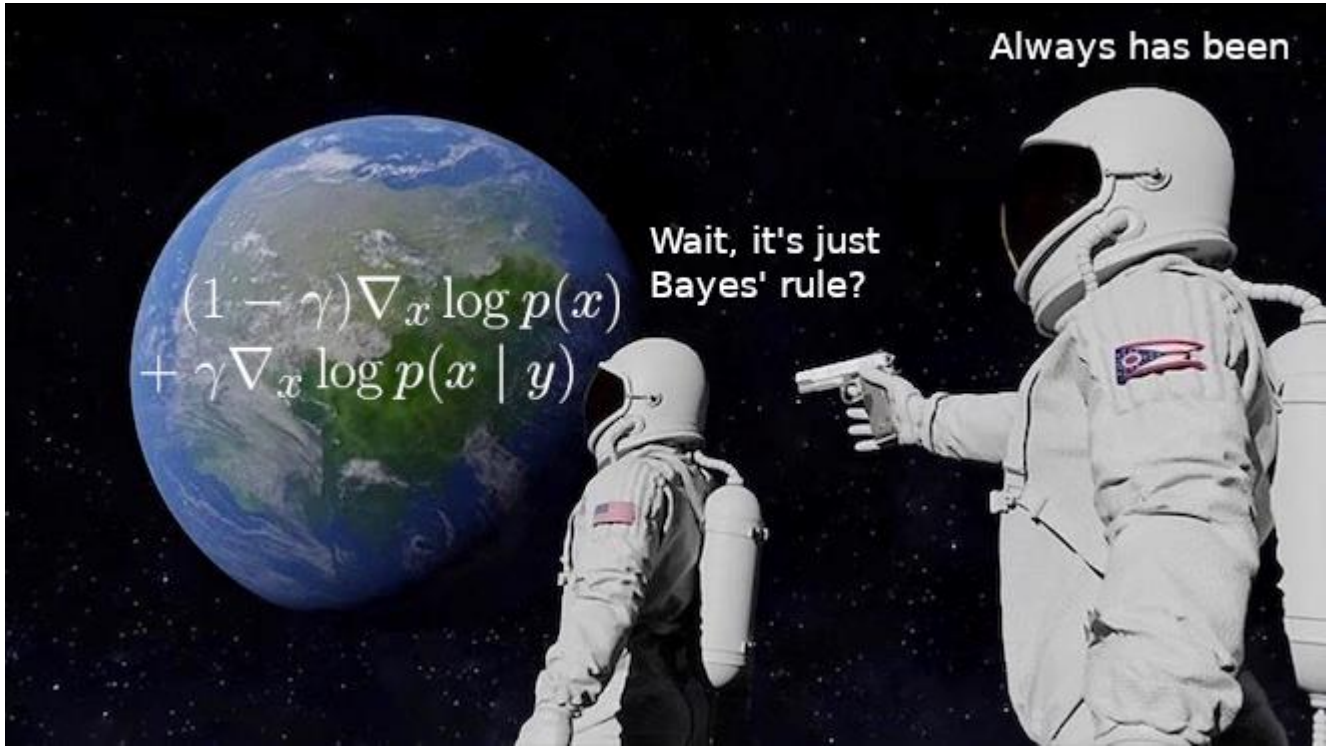
Use the gradient of this implicit classifier in classifier guidance!

$$\tilde{\boldsymbol{\epsilon}}_\theta(\mathbf{x}_t, \mathbf{c}) = \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{c}) - w\sigma_t \nabla_{\mathbf{x}_t} \log p(\mathbf{c}|\, \mathbf{x}_t)$$
$$= \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{c}) - w\sigma_t \nabla_{\mathbf{x}_t}[\log p\left(\mathbf{x}_t \mid \mathbf{c}\right) - \log p\left(\mathbf{x}_t\right)] = \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{c}) + w\,[\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{c}) - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t)]$$

Sampling is done with the following modified score function:

$$\tilde{\boldsymbol{\epsilon}}_\theta(\mathbf{x}_t, \mathbf{c}) = (1 + w)\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{c}) - w\boldsymbol{\epsilon}_\theta(\mathbf{x}_t)$$

It’s just Bayes’ Rule! (meme from Sander Dieleman)



Classifier-Free Guidance (CFG)

We have *constructed an implicit classifier* from our conditional generative model using Bayes' Rule and used that to guide the model.

Discriminative models tend to be better than implicit classifiers derived from generative models however…

Empirically classifier-free guidance works well though!

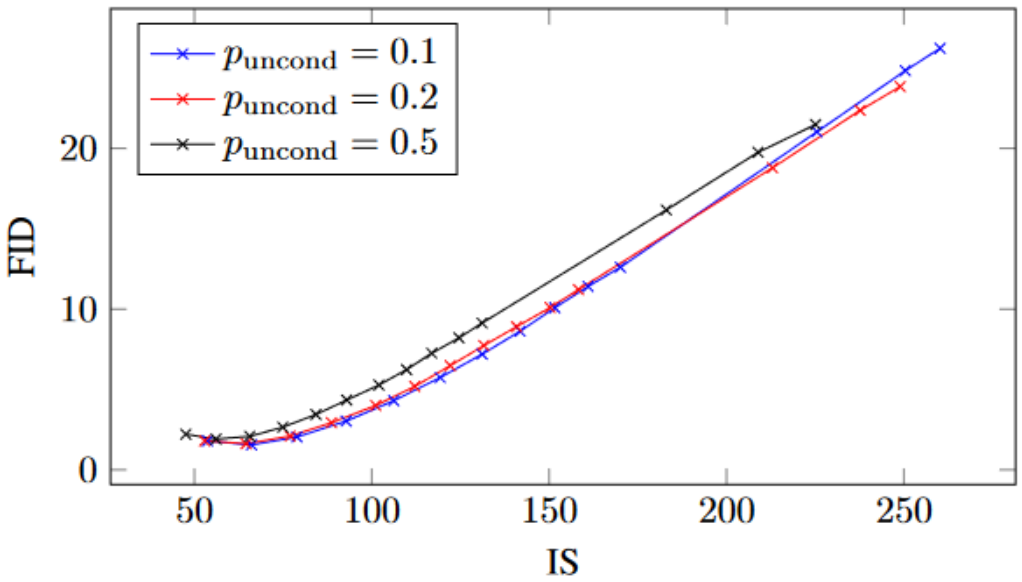
CFG decreases the unconditional likelihood of the sample while increasing the conditional likelihood.

Disadvantage: additional network evaluation needed for unconditional model for each sampling step

Results

Best FID results are obtained with small amount of guidance (w=0.1 or 0.3) and best Inception score is obtained with strong guidance (w >= 4)

Relatively small model capacity needed for unconditional generation



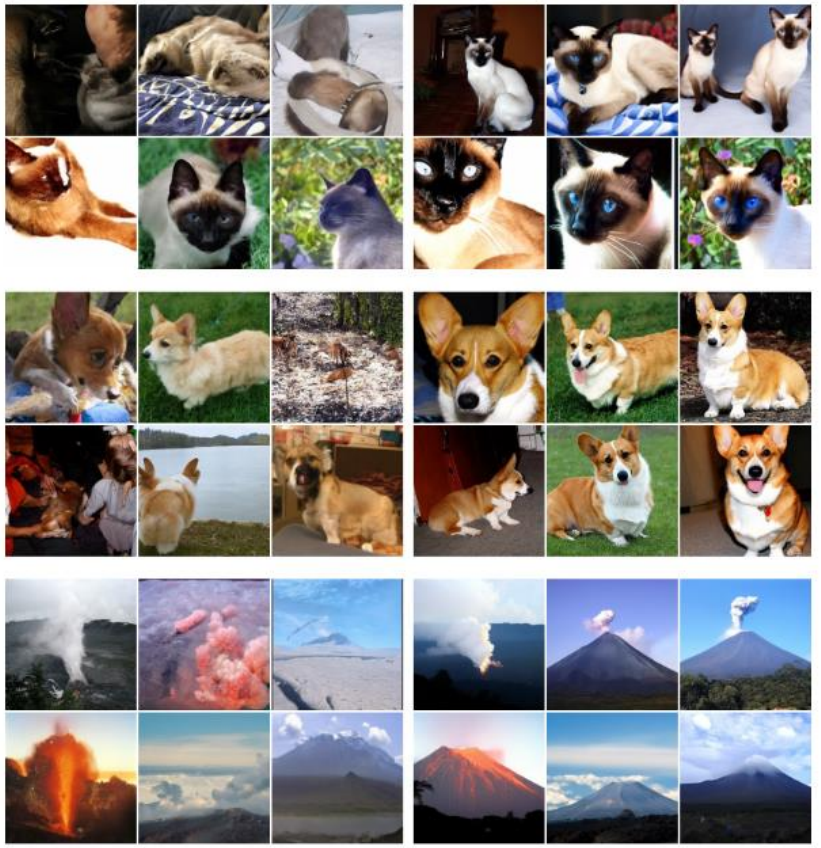
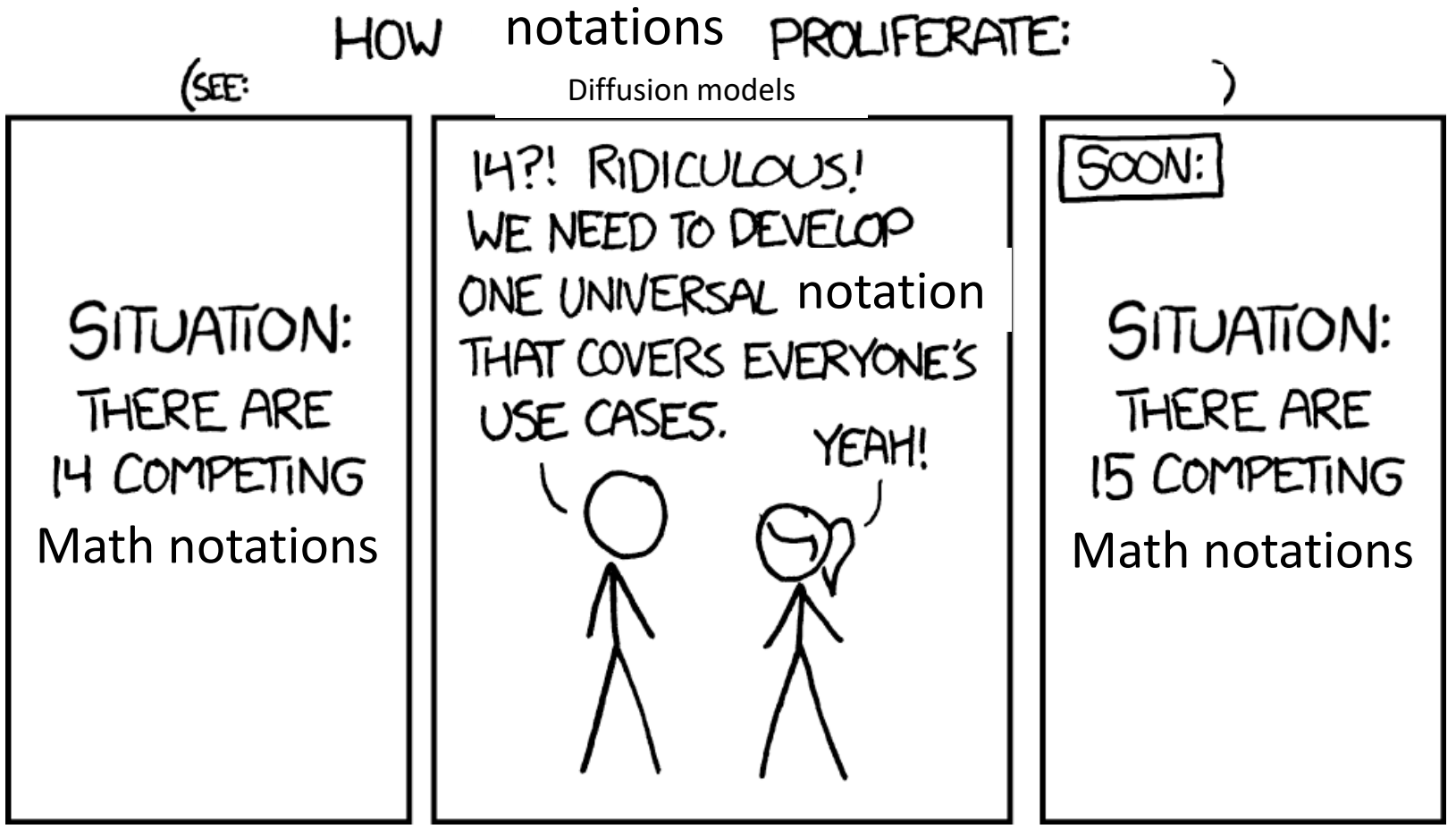


Figure 3: Classifier-free guidance on 128x128 ImageNet. Left: non-guided samples, right: classifier-free guided samples with $w = 3.0$. Interestingly, strongly guided samples such as these display saturated colors. See Fig. 8 for more.

Variational Diffusion Models

Differences in notation!



Forward process

Forward process runs between $t = 0$ and $t = 1$ and the latent variables of intermediate t is given by:

$$q(\mathbf{z}_t|\mathbf{x}) = \mathcal{N}(\alpha_t \mathbf{x}, \sigma_t^2 \mathbf{I})$$

Signal-noise ratio:

$$\mathrm{SNR}(t) = \alpha_t^2 / \sigma_t^2$$

where α_t and σ_t^2 are functions of t

Variance-preserving: $\alpha_t = \sqrt{(1 - \sigma_t^2)}$

Variance-exploding:$\alpha_t = 1$

Forward process

For $s < t$:

$$q(\mathbf{z}_t|\mathbf{z}_s) = \mathcal{N}\left(\alpha_{t|s}\mathbf{x}, \sigma_{t|s}^2\mathbf{I}\right)$$

where

$$\alpha_{t|s} = \alpha_t/\alpha_s,$$
$$\sigma_{t|s}^2 = \sigma_t^2 - \alpha_{t|s}^2\sigma_s^2$$

$$q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}_Q(\mathbf{z}_t, \mathbf{x}; s, t), \sigma_Q^2(s, t)\mathbf{I})$$

$$\text{where } \sigma_Q^{-2}(s, t) = \sigma_s^{-2} + \alpha_{t|s}^2\sigma_{t|s}^{-2} = \sigma_{t|s}^2\sigma_s^2/\sigma_t^2$$

$$\text{and } \boldsymbol{\mu}_Q(\mathbf{z}_t, \mathbf{x}; s, t) = \frac{\alpha_{t|s}\sigma_s^2}{\sigma_t^2}\mathbf{z}_t + \frac{\alpha_s\sigma_{t|s}^2}{\sigma_t^2}\mathbf{x}.$$

Noise schedule

Noise schedule is learnt (parameterized by an MLP)!

$$\sigma_t^2 = \text{sigmoid}(\gamma_\eta(t))$$

where $\gamma_\eta(t)$ is the neural network with params η

For variance-preserving diffusion processes:

$$\alpha_t^2 = \text{sigmoid}(-\gamma_\eta(t))$$

$$\text{SNR}(t) = \exp\left(-\gamma_\eta(t)\right)$$

Reverse-time generative model

Discretize time $t = 0 \rightarrow 1$ into T timesteps. $s(i) = (i-1)/T$ and $t(i) = i/T$. Our hierarchical generative model is:

$$p(\mathbf{x}) = \int_{\mathbf{z}} p(\mathbf{z}_1) p(\mathbf{x}|\mathbf{z}_0) \prod_{i=1}^{T} p(\mathbf{z}_{s(i)}|\mathbf{z}_{t(i)}).$$

The marginal distribution is a spherical Gaussian:

$$p(\mathbf{z}_1) = \mathcal{N}(\mathbf{z}_1; 0, \mathbf{I}).$$

The distribution $p(\mathbf{x}|\mathbf{z}_0)$ is given as:

$$p(\mathbf{x}|\mathbf{z}_0) = \prod_i p(x_i|z_{0,i}),$$

Reverse-time generative model

Conditional distribution:

$$p(\mathbf{z}_s|\mathbf{z}_t) = q(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x} = \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)),$$

where $\hat{\mathbf{x}}_\theta(\mathbf{z}_t; t)$ is the denoising model. It is parameterized in terms of a noise prediction model:

$$\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t) = (\mathbf{z}_t - \sigma_t \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t))/\alpha_t,$$

where $\hat{\boldsymbol{\epsilon}}_\theta(\mathbf{z}_t; t)$ is a neural network

Fourier features

Append channels $\sin(2^n \pi \mathbf{z})$ and $\cos(2^n \pi \mathbf{z})$ where $n \in \{n_{min}, \ldots, n_{max}\}$

These features amplify small changes to the input data, allowing the model to capture fine scale details of the data

Leads to large improvements in likelihood. Best results obtained with $n_{min} = 7, n_{max} = 8$

Variational lower bound

Treating it as a hierarchical generative model, like a VAE, the model is trained with the standard VLB:

$$-\log p(\mathbf{x}) \leq -\text{VLB}(\mathbf{x}) = \underbrace{D_{KL}(q(\mathbf{z}_1|\mathbf{x})||p(\mathbf{z}_1))}_{\text{Prior loss}} + \underbrace{\mathbb{E}_{q(\mathbf{z}_0|\mathbf{x})}[-\log p(\mathbf{x}|\mathbf{z}_0)]}_{\text{Reconstruction loss}} + \underbrace{\mathcal{L}_T(\mathbf{x})}_{\text{Diffusion loss}}.$$

Other losses cannot be ignored because of predicted variance

We now will investigate the diffusion loss further

Discrete-time model

In the case of finite timesteps, we get the following loss function:

$$\mathcal{L}_T(\mathbf{x}) = \sum_{i=1}^{T} \mathbb{E}_{q(\mathbf{z}_{t(i)}|\mathbf{x})} D_{KL}[q(\mathbf{z}_{s(i)}|\mathbf{z}_{t(i)}, \mathbf{x})||p(\mathbf{z}_{s(i)}|\mathbf{z}_{t(i)})].$$

This simplifies significantly to:

$$\mathcal{L}_T(\mathbf{x}) = \frac{T}{2} \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}),i\sim U\{1,T\}} \left[(\mathrm{SNR}(s) - \mathrm{SNR}(t)) \, ||\mathbf{x} - \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)||_2^2 \right],$$

For the specific choices of $\sigma_t, \alpha_t, \hat{\mathbf{x}}_\theta(\mathbf{z}_t; t)$ described previously, we get:

$$\mathcal{L}_T(\mathbf{x}) = \frac{T}{2} \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}),i\sim U\{1,T\}} \left[(\exp(\gamma_{\boldsymbol{\eta}}(t) - \gamma_{\boldsymbol{\eta}}(s)) - 1) \, \|\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{z}_t; t)\|_2^2 \right]$$

We can jointly optimize γ, η by maximizing the Monte Carlo estimator of this loss.

Note $\exp(\cdot) - 1$ is a common primitive $expm1(\cdot)$ in numerical computing packages can be stably implemented in 32-bit or lower precision, allowing for lower precision implementations of discrete-time diffusion models.

Continuous-time model

It can mathematically be shown that the VLB will be better for larger number of timesteps. This motivates the usage of continuous time where $T \to \infty$

Taking this limit gives the following loss:

$$\begin{aligned}\mathcal{L}_\infty(\mathbf{x}) &= -\frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})}\int_0^1 \mathrm{SNR}'(t)\,\|\mathbf{x}-\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t;t)\|_2^2\,dt, \\ &= -\frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}),t\sim\mathcal{U}(0,1)}\left[\mathrm{SNR}'(t)\,\|\mathbf{x}-\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t;t)\|_2^2\right].\end{aligned}$$

where $\mathrm{SNR}'(t) = d\mathrm{SNR}(t)/dt$

For the specific choices of $\sigma_t, \alpha_t, \hat{\mathbf{x}}_\theta(\mathbf{z}_t;t)$ described previously, we get:

$$\mathcal{L}_\infty(\mathbf{x}) = \frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}),t\sim\mathcal{U}(0,1)}\left[\gamma'_{\boldsymbol{\eta}}(t)\,\|\boldsymbol{\epsilon}-\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{z}_t;t)\|_2^2\right],$$

Once again, optimized and evaluated with Monte Carlo estimator

Reparameterization of the model

Since $\mathrm{SNR}(t)$ is strictly monotonically decreasing in time, it is invertible and we can define v as the SNR at timestep $t = SNR^{-1}(v)$.

We can instead have α_v, σ_v, $\mathbf{z}_v$, and $\tilde{\mathbf{x}}_\theta(\mathbf{z}_v; v)$, and rewrite the loss:

$$\mathcal{L}_\infty(\mathbf{x}) = \frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})} \int_{\mathrm{SNR}_{\min}}^{\mathrm{SNR}_{\max}} \|\mathbf{x} - \tilde{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_v, v)\|_2^2 \, dv,$$

where $\mathrm{SNR}_{min} = \mathrm{SNR}(0)$ and $\mathrm{SNR}_{max} = \mathrm{SNR}(1)$

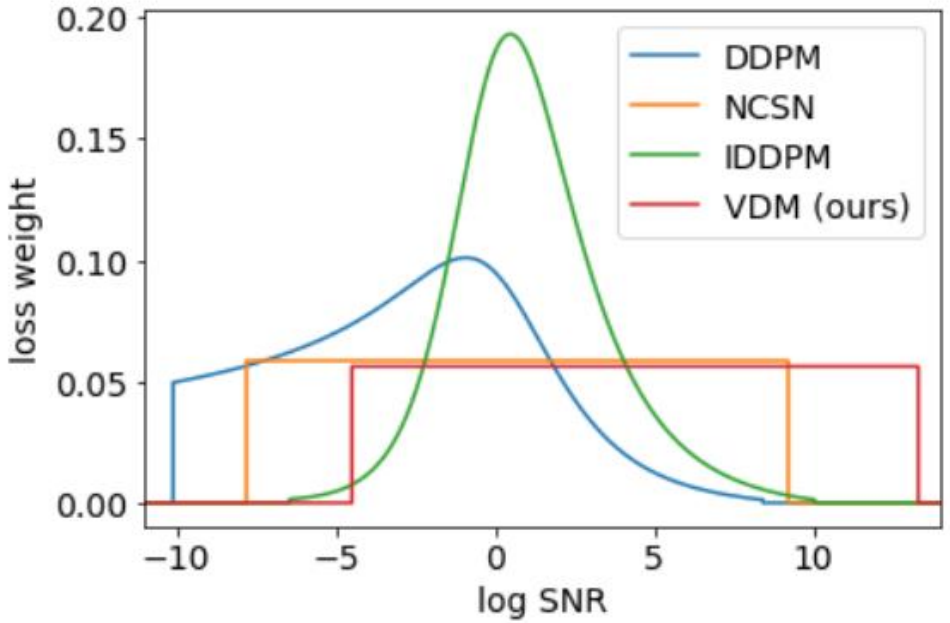
α_t & σ_t only at the endpoints affect the diffusion loss, the loss is invariant to the actual shape of the SNR function

Any SNR function (inc. VP and VE) gives equivalent diffusion models

Weighted diffusion loss

A weighted diffusion loss (which includes previous models like DDPM and NCSN) has similar properties:

$$\mathcal{L}_\infty(\mathbf{x}, w) = \frac{1}{2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I})} \int_{\mathrm{SNR}_{\min}}^{\mathrm{SNR}_{\max}} w(v)\, \|\mathbf{x} - \tilde{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_v, v)\|_2^2\, dv,$$



Results

Continuous time formulation, learned variance, variance minimization, and Fourier features all improve performance as demonstrated by ablation studies

Figure 3: Non cherry-picked unconditional samples from our Imagenet 64x64 model, trained in continuous time and generated using $T = 1000$. The model's hyper-parameters and parameters are optimized w.r.t. the likelihood bound, so the model is not optimized for synthesis quality.

Model (Bits per dim on test set)	Type	CIFAR10 no data aug.	CIFAR10 data aug.	ImageNet 32x32	ImageNet 64x64
Previous work					
ResNet VAE with IAF [Kingma et al., 2016]	VAE	3.11			
Very Deep VAE [Child, 2020]	VAE	2.87		3.80	3.52
NVAE [Vahdat and Kautz, 2020]	VAE	2.91		3.92	
Glow [Kingma and Dhariwal, 2018]	Flow		$3.35^{(B)}$	4.09	3.81
Flow++ [Ho et al., 2019a]	Flow	3.08		3.86	3.69
PixelCNN [Van Oord et al., 2016]	AR	3.03		3.83	3.57
PixelCNN++ [Salimans et al., 2017]	AR	2.92			
Image Transformer [Parmar et al., 2018]	AR	2.90		3.77	
SPN [Menick and Kalchbrenner, 2018]	AR				3.52
Sparse Transformer [Child et al., 2019]	AR	2.80			3.44
Routing Transformer [Roy et al., 2021]	AR				3.43
Sparse Transformer + DistAug [Jun et al., 2020]	AR		$2.53^{(A)}$		
DDPM [Ho et al., 2020]	Diff		$3.69^{(C)}$		
EBM-DRL [Gao et al., 2020]	Diff		$3.18^{(C)}$		
Score SDE [Song et al., 2021b]	Diff	2.99			
Improved DDPM [Nichol and Dhariwal, 2021]	Diff	2.94			3.54
Concurrent work					
CR-NVAE [Sinha and Dieng, 2021]	VAE		$2.51^{(A)}$		
LSGM [Vahdat et al., 2021]	Diff	2.87			
ScoreFlow [Song et al., 2021a] (variational bound)	Diff		$2.90^{(C)}$	3.86	
ScoreFlow [Song et al., 2021a] (cont. norm. flow)	Diff	2.83	$2.80^{(C)}$	3.76	
Our work					
VDM (variational bound)	Diff	**2.65**	**$2.49^{(A)}$**	**3.72**	**3.40**

Tackling the Generative Learning Trilemma with Denoising Diffusion GANs

The Generative Learning Trilemma

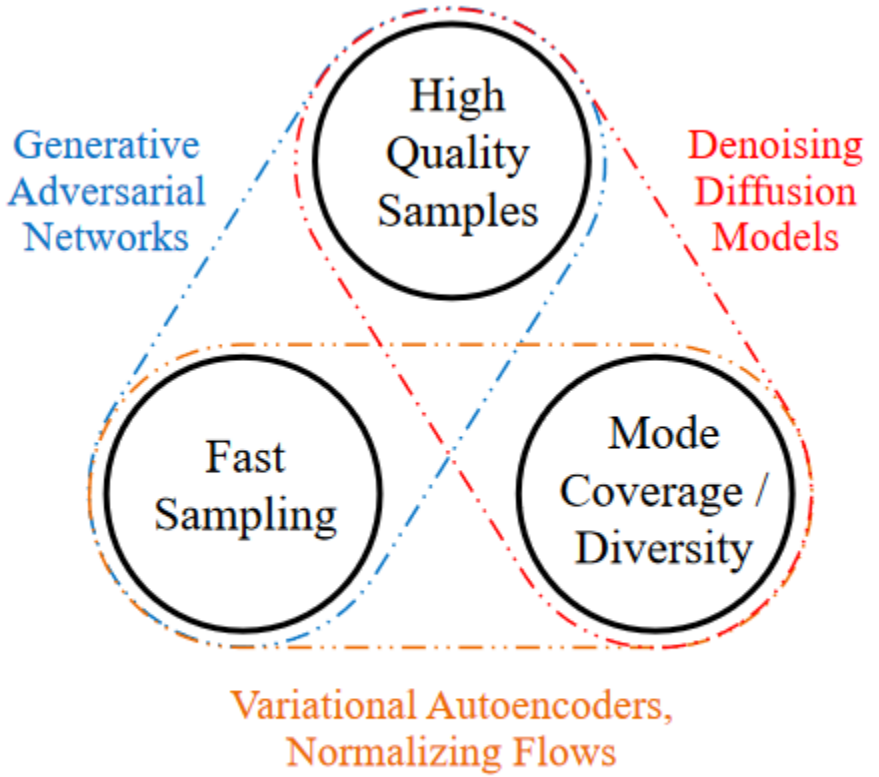
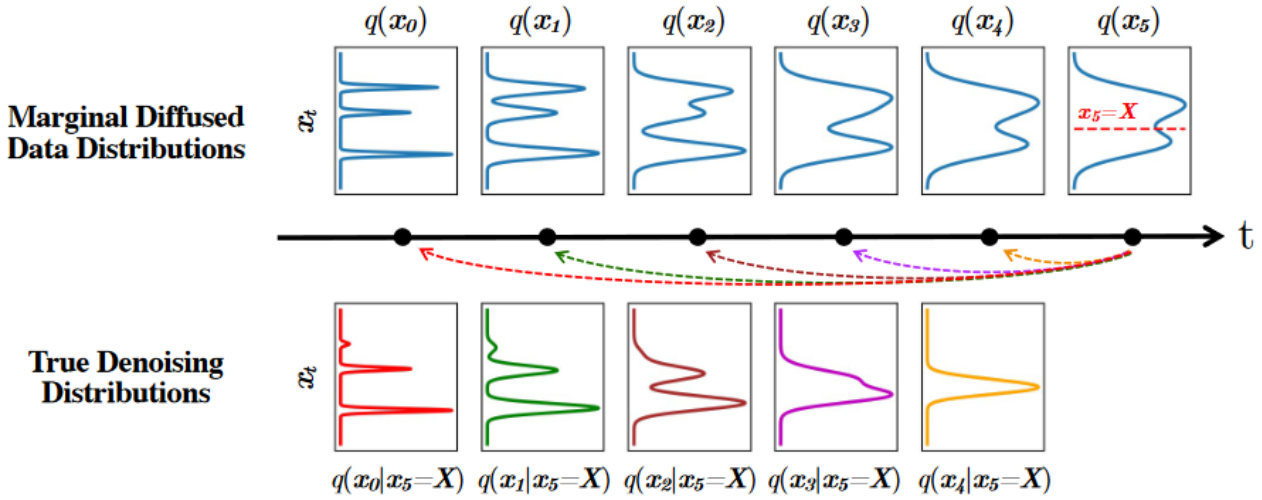


Figure 1: Generative learning trilemma.

Incorrect assumptions under few sampling steps

The reverse (denoising) process $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ is assumed to be a Gaussian distribution

With fewer denoising steps, this assumption no longer holds true, and the true denoising distribution is multimodal:



Denoising Diffusion GAN

Introduce a time-dependent discriminator $D_\phi(\mathbf{x}_{t-1}, \mathbf{x}_t, t)$ that decides if $\mathbf{x}_{t-1}$ is a plausible denoised version of $\mathbf{x}_t$

$$\mathbb{E}_{q(\mathbf{x}_t)q(\mathbf{x}_{t-1}|\mathbf{x}_t)}[-\log(D_\phi(\mathbf{x}_{t-1}, \mathbf{x}_t, t))] = \mathbb{E}_{q(\mathbf{x}_0)q(\mathbf{x}_{t-1}|\mathbf{x}_0)q(\mathbf{x}_t|\mathbf{x}_{t-1})}[-\log(D_\phi(\mathbf{x}_{t-1}, \mathbf{x}_t, t))].$$

The generator is also given a random latent vector:

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) := \int p_\theta(\mathbf{x}_0|\mathbf{x}_t)q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)d\mathbf{x}_0 = \int p(\mathbf{z})q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0 = G_\theta(\mathbf{x}_t, \mathbf{z}, t))d\mathbf{z},$$

Denoising Diffusion GAN

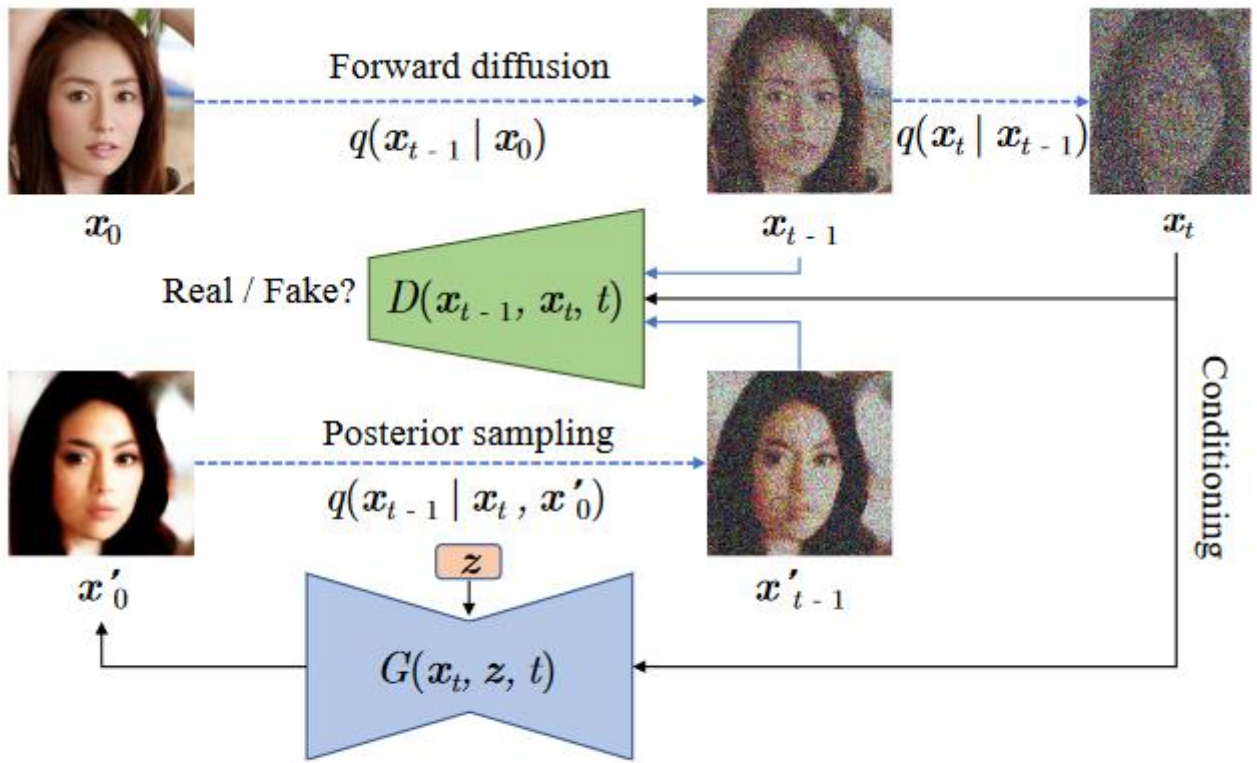


Figure 3: The training process of denoising diffusion GAN.

Advantages over standard GANs

This model breaks the generation process into several conditional denoising diffusion steps in which each step is relatively simple to model, due to the strong conditioning on $\mathbf{x}_t$

Additionally, the diffusion process smoothens the data distribution making the discriminator less likely to overfit.

Empirically, training was stable with no loss explosions

Results

Used a standard U-net, the latent variable controlled adaptive group normalization layers in the network, timestep as embeddings

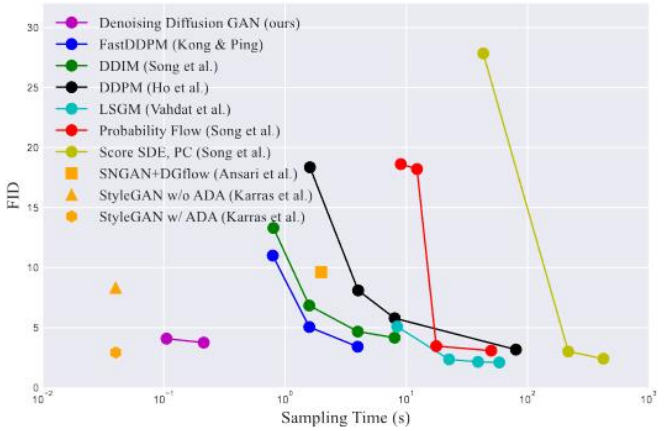


Figure 4: Sample quality vs sampling time trade-off.

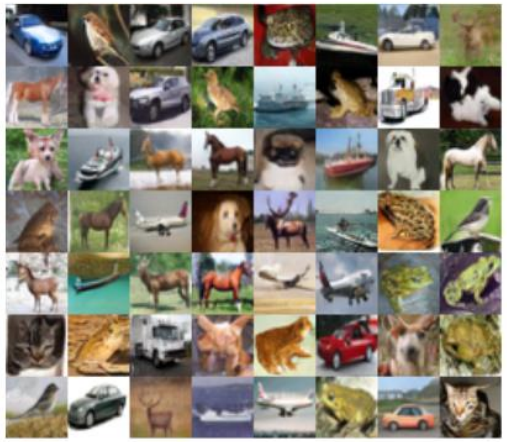


Figure 5: CIFAR-10 qualitative samples.

Model	IS↑	FID↓	Recall↑	NFE ↓	Time (s) ↓
Denoising Diffusion GAN (ours), T=4	9.63	3.75	0.57	4	0.21
DDPM (Ho et al., 2020)	9.46	3.21	0.57	1000	80.5
NCSN (Song & Ermon, 2019)	8.87	25.3	-	1000	107.9
Adversarial DSM (Jolicoeur-Martineau et al., 2021b)	-	6.10	-	1000	-
Likelihood SDE (Song et al., 2021b)	-	2.87	-	-	-
Score SDE (VE) (Song et al., 2021c)	9.89	2.20	0.59	2000	423.2
Score SDE (VP) (Song et al., 2021c)	9.68	2.41	0.59	2000	421.5
Probability Flow (VP) (Song et al., 2021c)	9.83	3.08	0.57	140	50.9
LSGM (Vahdat et al., 2021)	9.87	2.10	0.61	147	44.5
DDIM, T=50 (Song et al., 2021a)	8.78	4.67	0.53	50	4.01
FastDDPM, T=50 (Kong & Ping, 2021)	8.98	3.41	0.56	50	4.01
Recovery EBM (Gao et al., 2021)	8.30	9.58	-	180	-
Improved DDPM (Nichol & Dhariwal, 2021)	-	2.90	-	4000	-
VDM (Kingma et al., 2021)	-	4.00	-	1000	-
UDM (Kim et al., 2021)	10.1	2.33	-	2000	-
D3PMs (Austin et al., 2021)	8.56	7.34	-	1000	-
Gotta Go Fast (Jolicoeur-Martineau et al., 2021a)	-	2.44	-	180	-
DDPM Distillation (Luhman & Luhman, 2021)	8.36	9.36	0.51	1	-

Results

Table 2: Ablation studies on CIFAR-10.

Model Variants	IS↑	FID↓	Recall↑
T = 1	8.93	14.6	0.19
T = 2	**9.80**	4.08	0.54
T = 4	9.63	**3.75**	**0.57**
T = 8	9.43	4.36	0.56
One-shot w/ aug	8.96	13.2	0.25
Direct denoising	9.10	6.03	0.53
Noise generation	8.79	8.04	0.52
No latent variable	8.37	20.6	0.42

Table 3: Mode coverage on StackedMNIST.

Model	Modes↑	KL↓
VEEGAN (Srivastava et al.)	762	2.173
PacGAN (Lin et al.)	992	0.277
PresGAN (Dieng et al.)	**1000**	0.115
InclusiveGAN (Yu et al.)	997	0.200
StyleGAN2 (Karras et al.)	940	0.424
Adv. DSM (Jolicoeur-Martineau et al.)	**1000**	1.49
VAEBM (Xiao et al.)	**1000**	0.087
Denoising Diffusion GAN (ours)	**1000**	**0.071**

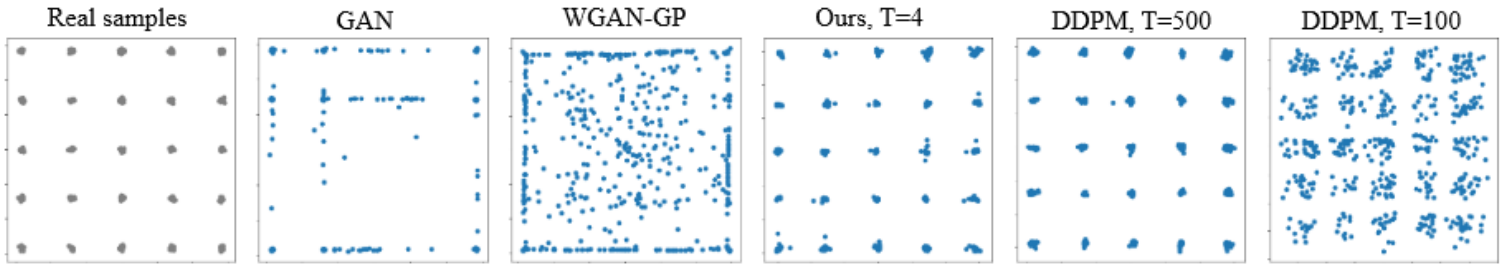


Figure 6: Qualitative results on the 25-Gaussians dataset.