Diffusion Study Group #5

10/8/2022

Tanishq Abraham

A recap (DDIM)

Non-Markovian forward processes

A family of joint distributions indexed by a real vector σ:

$$q_\sigma(\mathbf{x}_{1:T} \mid \mathbf{x}_0) \coloneqq q_\sigma(\mathbf{x}_T \mid \mathbf{x}_0) \prod_{t=2}^{T} q_\sigma(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0)$$

$$q_\sigma(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0)$$

$$= \mathcal{N}\left(\sqrt{\alpha_{t-1}}\mathbf{x}_0 + \sqrt{1 - \alpha_{t-1} - \sigma_t^2}\,\frac{\mathbf{x}_t - \sqrt{\alpha_t}\mathbf{x}_0}{\sqrt{1-\alpha_t}}, \sigma^2\mathbf{I}\right)$$

This form is chosen since it ensures $q_\sigma(\mathbf{x}_T \mid \mathbf{x}_0) = \mathcal{N}\left(\sqrt{\alpha_T}\mathbf{x}_0, (1 - \alpha_T)\mathbf{I}\right)$

Non-Markovian forward processes

The forward process is therefore given by Bayes' rule:

$$q_\sigma(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{x}_0) = \frac{q_\sigma(\mathbf{x}_{t-1} \mid \mathbf{x}_t,\mathbf{x}_0)q_\sigma(\mathbf{x}_t|\mathbf{x}_0)}{q_\sigma(\mathbf{x}_{t-1}|\mathbf{x}_0)}$$

Note that the forward process is no longer Markovian. Additionally, note that the magnitude of σ controls how stochastic the process is. It becomes completely deterministic at $\sigma = 0$.

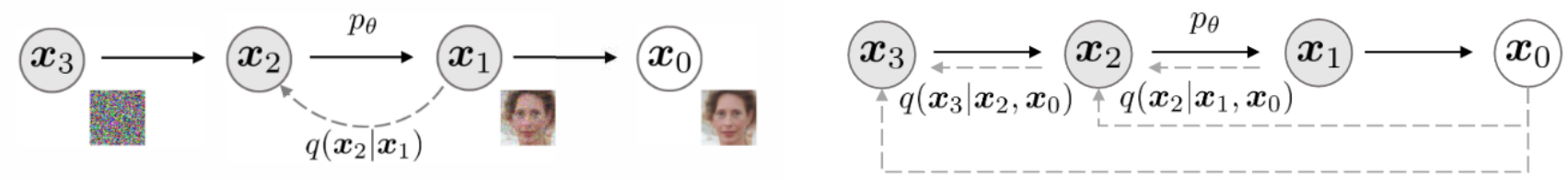


Figure 1: Graphical models for diffusion (left) and non-Markovian (right) inference models.

Key insight

If we want to train our reverse process model $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$, we get the following objective:

$$J_\sigma(\epsilon_\theta) := \mathbb{E}_{\boldsymbol{x}_{0:T}\sim q_\sigma(\boldsymbol{x}_{0:T})}[\log q_\sigma(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0) - \log p_\theta(\boldsymbol{x}_{0:T})] \tag{11}$$

$$= \mathbb{E}_{\boldsymbol{x}_{0:T}\sim q_\sigma(\boldsymbol{x}_{0:T})}\left[\log q_\sigma(\boldsymbol{x}_T|\boldsymbol{x}_0) + \sum_{t=2}^{T}\log q_\sigma(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0) - \sum_{t=1}^{T}\log p_\theta^{(t)}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) - \log p_\theta(\boldsymbol{x}_T)\right]$$

However it can be shown that if the model is time-dependent, minimizing this objective is equivalent to minimizing the simplified DDPM objective!

DDIM is simply a novel sampling process for diffusion models!

DDIM sampling

Sampling procedure:

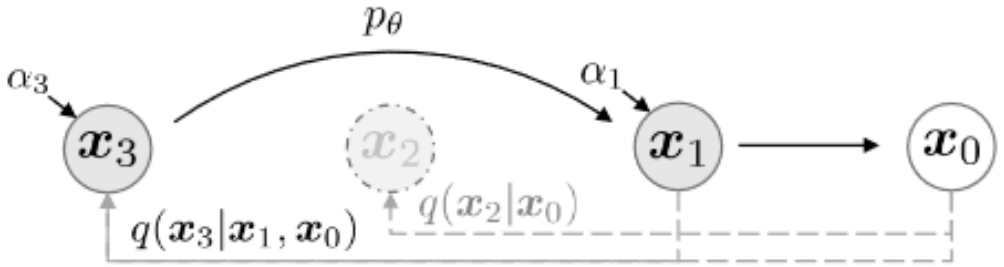
$$\mathbf{x}_{t-1} = \sqrt{\alpha_{t-1}}\left(\frac{\mathbf{x}_t - \sqrt{1-\alpha_t}\epsilon_\theta(\mathbf{x}_t, t)}{\sqrt{\alpha_t}}\right) + \sqrt{1-\alpha_{t-1}-\sigma_t^2}\epsilon_\theta(\mathbf{x}_t, t) + \sigma_t\epsilon_t$$

where $\epsilon_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

Specific parameterization of σ_t gives DDPM and $\sigma_t = 0$ gives DDIM. Usually parameterized by η in this form:

$$\sigma_{\tau_i}(\eta) = \eta\sqrt{(1-\alpha_{\tau_{i-1}})/(1-\alpha_{\tau_i})}\sqrt{1-\alpha_{\tau_i}/\alpha_{\tau_{i-1}}},$$

Accelerated generation:



Continuous version of DDIM

Rewrite the previous deterministic iterative procedure as such:

$$\frac{\boldsymbol{x}_{t-\Delta t}}{\sqrt{\alpha_{t-\Delta t}}} = \frac{\boldsymbol{x}_t}{\sqrt{\alpha_t}} + \left(\sqrt{\frac{1-\alpha_{t-\Delta t}}{\alpha_{t-\Delta t}}} - \sqrt{\frac{1-\alpha_t}{\alpha_t}}\right) \epsilon_\theta^{(t)}(\boldsymbol{x}_t)$$

Setting $\left(\sqrt{1-\alpha}/\alpha\right) = \sigma$ and $\boldsymbol{x}/\sqrt{\alpha} = \bar{\boldsymbol{x}}$, this is an Euler integration of the following ODE:

$$\mathrm{d}\bar{\boldsymbol{x}}(t) = \epsilon_\theta^{(t)}\left(\frac{\bar{\boldsymbol{x}}(t)}{\sqrt{\sigma^2+1}}\right)\mathrm{d}\sigma(t),$$

Continuous version of DDIM

The ODE is equivalent to Song et al.'s probability flow ODE for the VE SDE, but their discretization is not equivalent:

$$\frac{\boldsymbol{x}_{t-\Delta t}}{\sqrt{\alpha_{t-\Delta t}}} = \frac{\boldsymbol{x}_t}{\sqrt{\alpha_t}} + \frac{1}{2}\left(\frac{1-\alpha_{t-\Delta t}}{\alpha_{t-\Delta t}} - \frac{1-\alpha_t}{\alpha_t}\right) \cdot \sqrt{\frac{\alpha_t}{1-\alpha_t}} \cdot \epsilon_\theta^{(t)}(\boldsymbol{x}_t)$$

Just like Song et al.'s ODEs, DDIM ODE provides unique identifiable encodings

Improved Denoising Diffusion Probabilistic Models

Log-likelihood

Original DDPM was unable to achieve competitive log-likelihoods

This is an important metric that may indicate other issues like mode coverage/diversity

A simple change: $T = 1000 \rightarrow 4000$, changes NLL from 3.99 bits/dim to 3.77

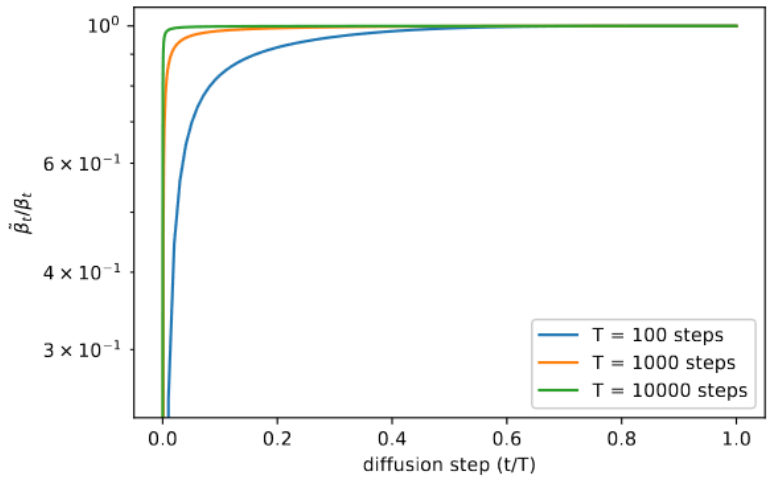
Examining the effect of reverse process variance on NLL

Remember:

$$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) := \mathcal{N}\left(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t)\right)$$

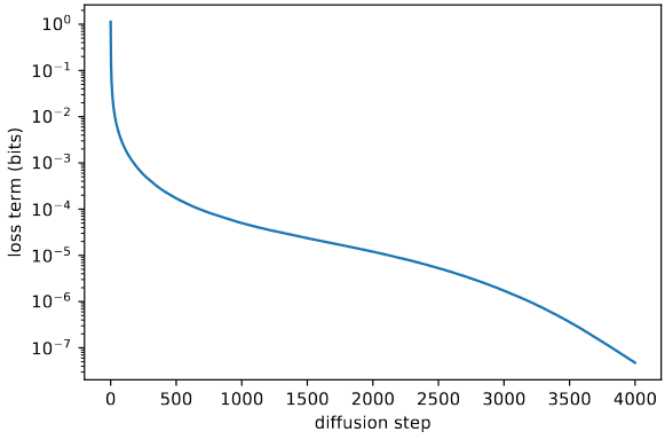
DDPM had $\Sigma_\theta(\mathbf{x}_t, t) = \sigma_t^2 \mathbf{I}$ where $\sigma_t^2 = \beta_t$ or $\sigma_t^2 = \tilde{\beta}_t$ both worked equally well.

How do these two possible σ_t^2 differ? – not much difference except close to $t = 0$



Examining the effect of reverse process variance on NLL

However, our VLB loss is higher closer to $t = 0$:



This indicates the differences between those choices for $\Sigma_\theta(\mathbf{x}_t, t) = \sigma_t^2 \mathbf{I}$ do affect the NLL

This paper therefore proposes a new form of $\Sigma_\theta(\mathbf{x}_t, t)$ altogether, one that is parameterized by a neural network!

Learned reverse process variance

The model outputs a vector v, which is used to give the following variance:

$$\Sigma_\theta(\mathbf{x}_t, t) = \exp\left(v \log \beta_t + (1 - v) \log \tilde{\beta}_t\right)$$

Since L_{simple} does not depend on the variance, a different reweighted objective is used:

$$L_{hybrid} = L_{simple} + \lambda L_{vlb}$$

Where $L_{vlb} = L_0 + L_1 + \cdots + L_{T-1} + L_T$ (KL terms that are tractable for Gaussians)

$\lambda = 0.001$ in all experiments and a stop-gradient (?) is applied to $\boldsymbol{\mu}_\theta(\mathbf{x}_t, t)$ for the L_{vlb} term so that this term mostly influences just the variance.

Noise schedules

While linear schedules are good for high-resolution images, it was observed to be sub-optimal for 32x32 and 64x64 images. A cosine schedule (for training) for $\bar{\alpha}_t$ seems to be better instead:

$$\bar{\alpha}_t = \frac{f(t)}{f(0)}, \quad f(t) = \cos\left(\frac{t/T + s}{1+s} \cdot \frac{\pi}{2}\right)^2$$

Figure 3. Latent samples from linear (top) and cosine (bottom) schedules respectively at linearly spaced values of t from 0 to T. The latents in the last quarter of the linear schedule are almost purely noise, whereas the cosine schedule adds noise more slowly

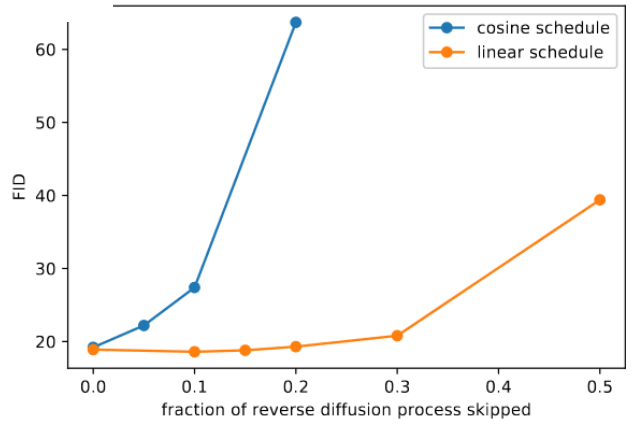
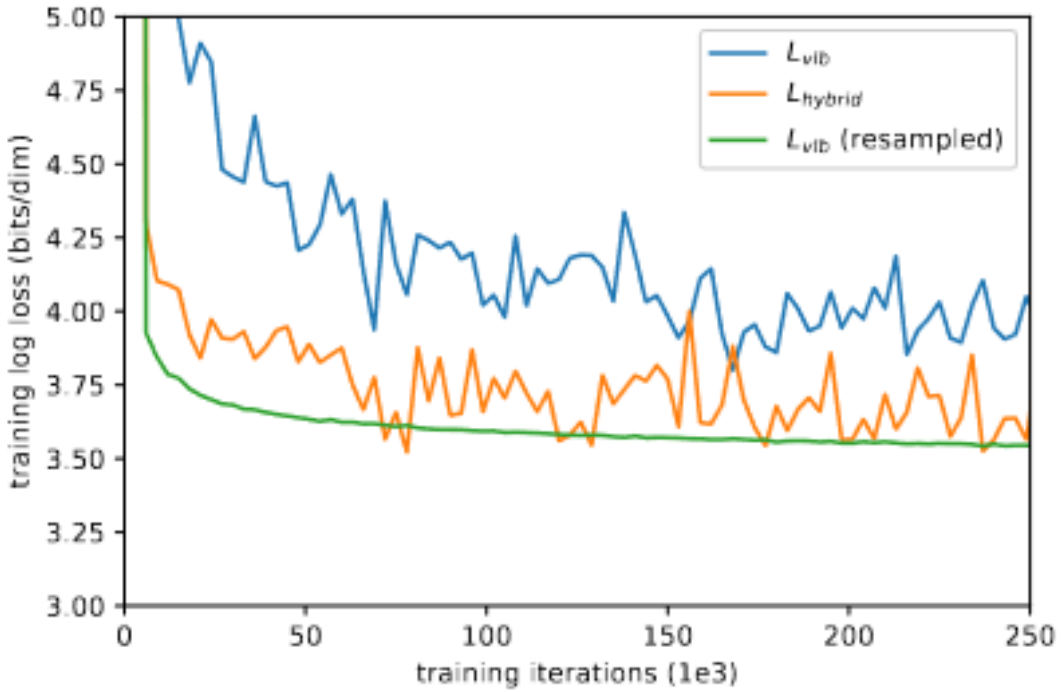


Figure 4. FID when skipping a prefix of the reverse diffusion process on ImageNet 64×64.

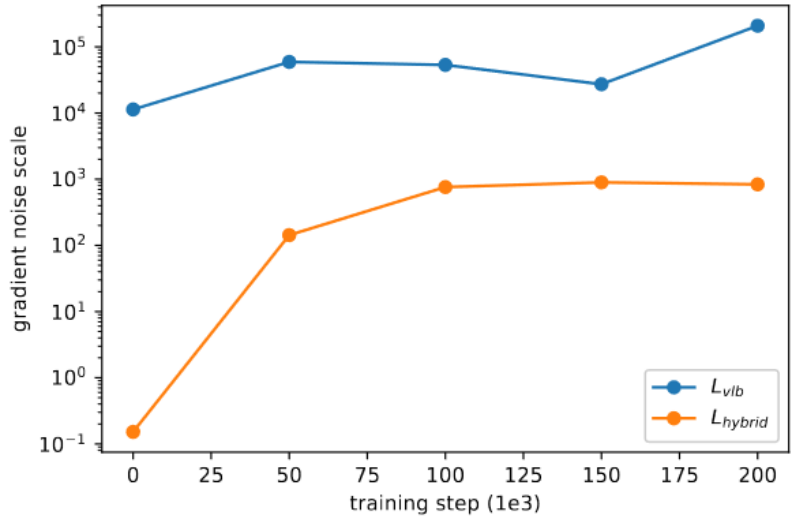
Reducing gradient noise

Directly optimizing L_{vlb} would be better than L_{hybrid} but L_{vlb} is difficult to optimize in practice:



Reducing gradient noise

The authors examined the *gradient noise scales*:



In order to address the gradient noise, they employ importance sampling (does not help with L_{hybrid}):

$$L_{\text{vlb}} = E_{t\sim p_t}\left[\frac{L_t}{p_t}\right], \text{ where } p_t \propto \sqrt{E[L_t^2]} \text{ and } \sum p_t = 1$$

Results and Ablations

L_{hybrid} and cosine schedule improves log-likelihood with comparable FID to original DDPM

Training with L_{vlb} gives best log-likelihood but harms FID

Table 1. Ablating schedule and objective on ImageNet 64 × 64.

Iters	T	Schedule	Objective	NLL	FID
200K	1K	linear	L_{simple}	3.99	32.5
200K	4K	linear	L_{simple}	3.77	31.3
200K	4K	linear	L_{hybrid}	3.66	32.2
200K	4K	cosine	L_{simple}	3.68	**27.0**
200K	4K	cosine	L_{hybrid}	3.62	28.0
200K	4K	cosine	L_{vlb}	**3.57**	56.7
1.5M	4K	cosine	L_{hybrid}	3.57	**19.2**
1.5M	4K	cosine	L_{vlb}	**3.53**	40.1

Table 2. Ablating schedule and objective on CIFAR-10.

Iters	T	Schedule	Objective	NLL	FID
500K	1K	linear	L_{simple}	3.73	3.29
500K	4K	linear	L_{simple}	3.37	**2.90**
500K	4K	linear	L_{hybrid}	3.26	3.07
500K	4K	cosine	L_{simple}	3.26	3.05
500K	4K	cosine	L_{hybrid}	3.17	3.19
500K	4K	cosine	L_{vlb}	**2.94**	11.47

Speeding up sampling

Given the original sequence of timesteps $(1, 2, \dots, T)$, choose some arbitrary subsequence S. Given the original noise schedule $\bar{\alpha}_t$, we have $\bar{\alpha}_{S_t}$ for the subsequence, and also the following sampling variances:

$$\beta_{S_t} = 1 - \frac{\bar{\alpha}_{S_t}}{\bar{\alpha}_{S_{t-1}}}, \tilde{\beta}_{S_t} = \frac{1 - \bar{\alpha}_{S_{t-1}}}{1 - \bar{\alpha}_{S_t}} \beta_{S_t}$$

which can then be plugged into $\Sigma_\theta(\mathbf{x}_{S_t}, S_t)$, allowing for sampling from $p_\theta(\mathbf{x}_{S_{t-1}} \mid \mathbf{x}_{S_t})$

Improved performance with learned variances over fixed variances.
Near-optimal FIDs with 100 sampling steps

DDIM performs better with <50 steps but worse with >50 steps

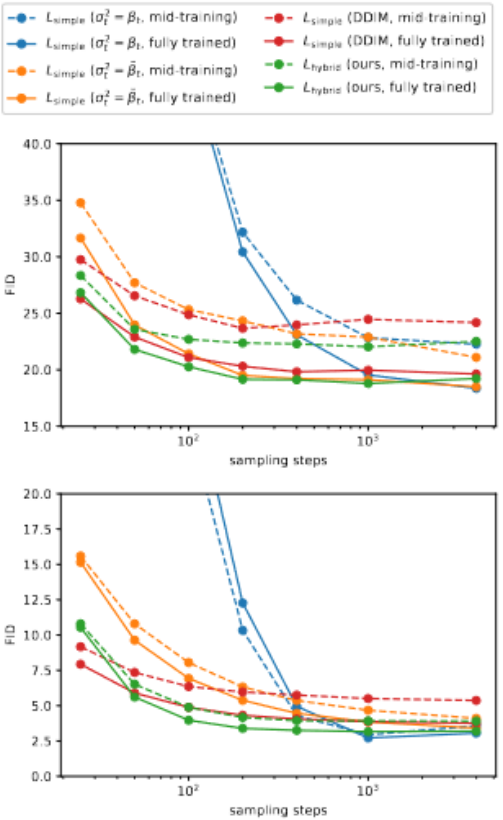


Figure 8. FID versus number of sampling steps, for models trained on ImageNet 64 × 64 (top) and CIFAR-10 (bottom). All models were trained with 4000 diffusion steps.

Class-conditional DDPM

Done to compare to class-conditional GANs

Class information is passed through the same pathway as the timesteps

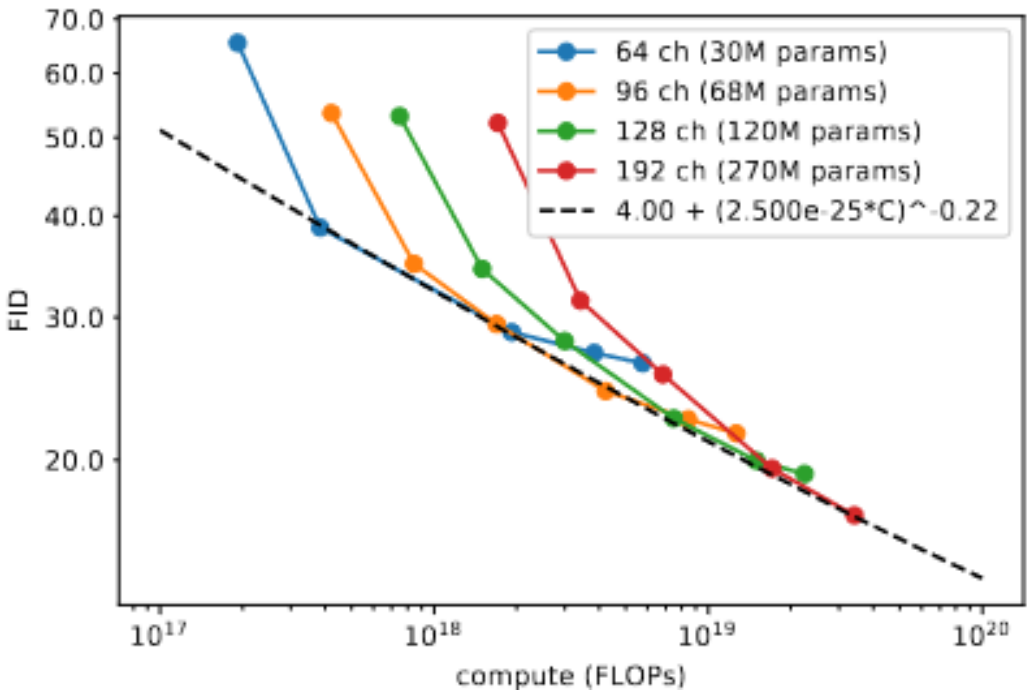
A class embedding is added to the timestep embedding, which are passed to the residual blocks throughout the model.

Table 4. Sample quality comparison on class-conditional ImageNet 64 × 64. Precision and recall (Kynkäänniemi et al., 2019) are measured using Inception-V3 features and $K = 5$. We trained BigGAN-deep for 125K iterations, and did not use truncation for sampling to maximize recall for the GAN.

Model	FID	Prec.	Recall
BigGAN-deep (Brock et al., 2018)	4.06	**0.86**	0.59
Improved Diffusion (small)	6.92	0.77	**0.72**
Improved Diffusion (large)	**2.92**	0.82	**0.71**

Figure 9. Class-conditional ImageNet 64 × 64 samples generated using 250 sampling steps from L_{hybrid} model (FID 2.92). The classes are 9: ostrich, 11: goldfinch, 130: flamingo, 141: redshank, 154: pekinese, 157: papillon, 97: drake and 28: spotted salamander. We see that there is a high diversity in each class, suggesting good coverage of the target distribution

Model Scaling



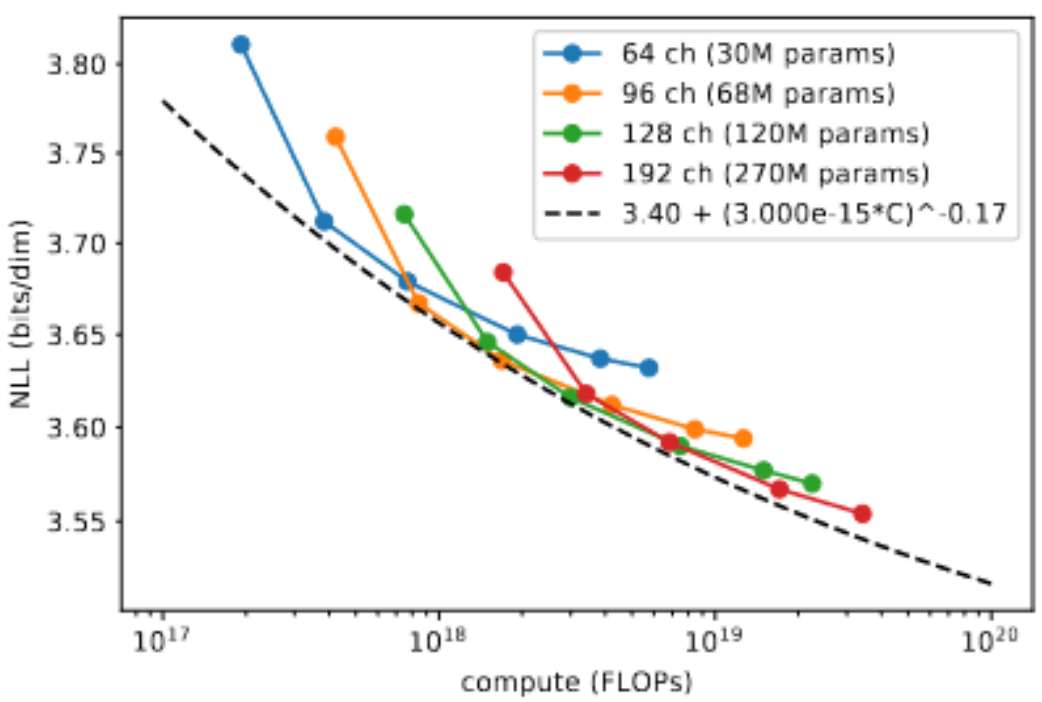


Figure 10. FID and validation NLL throughout training on ImageNet 64 × 64 for different model sizes. The constant for the FID trend line was approximated using the FID of in-distribution data. For the NLL trend line, the constant was approximated by rounding down the current state-of-the-art NLL (Roy et al., 2020) on this dataset.

Diffusion Models Beat GANs on Image Synthesis

Architectural Improvements

The authors performed a comprehensive ablation study on ImageNet 128x128, batch size=256, 250 sampling steps for the following architectural changes to the original U-net in DDPM:

- Increasing depth versus width, holding model size relatively constant.
- Increasing the number of attention heads.
- Using attention at 32×32, 16×16, and 8×8 resolutions rather than only at 16×16.
- Using the BigGAN [5] residual block for upsampling and downsampling the activations, following [60].
- Rescaling residual connections with $\frac{1}{\sqrt{2}}$, following [60, 27, 28].

Architectural Improvements

All changes have compounding improving effect except rescaling residual block (omitted in later experiments)

Increasing model depth improves FID but slows down training time and is omitted

Channels	Depth	Heads	Attention resolutions	BigGAN up/downsample	Rescale resblock	FID 700K	FID 1200K
160	2	1	16	✗	✗	15.33	13.21
128	4					-0.21	-0.48
		4				-0.54	-0.82
			32,16,8			-0.72	-0.66
				✓		-1.20	-1.21
					✓	0.16	0.25
160	2	4	32,16,8	✓	✗	**-3.14**	**-3.00**

Table 1: Ablation of various architecture changes, evaluated at 700K and 1200K iterations

Architectural Improvements

The configuration of the global attention layer at the 16x16 resolution:

- More heads or lower channels per head both lead to improved FID

Number of heads	Channels per head	FID
1		14.08
2		-0.50
4		-0.97
8		-1.17
	32	-1.36
	64	-1.03
	128	-1.08

Table 2: Ablation of various attention configurations. More heads or lower channels per heads both lead to improved FID.

Architectural Improvements

Timestep+label embeddings are incorporated through shift and scaling of the group normalization

$$AdaGN(h, y) = y_s \text{GroupNorm}(h) + y_b$$

where h are the intermediate activations of a residual block, and $y = [y_s, y_b]$ are obtained from a linear projection of the embeddings

Operation	FID
AdaGN	13.06
Addition + GroupNorm	15.08

Table 3: Ablating the element-wise operation used when projecting timestep and class embeddings into each residual block. Replacing AdaGN with the Addition + GroupNorm layer from Ho et al. [25] makes FID worse.

Ablated Diffusion Model (ADM): Variable width with2 residual blocks per resolution, multiple heads with 64 channels per head, attention at 32, 16 and 8 resolutions, BigGAN residual blocks for up and downsampling, and AdaGN for injecting timestep+label embeddings into residual blocks.

Classifier Guidance

Mathematical derivation in paper demonstrates that the mean for the reverse process can be updated to be:

$$\mu_y = \mu + \Sigma g$$

where $g = \nabla_{x_t} \log p_\phi(y|x_t)$ (the gradient of the classifier output w.r.t. the input image x_t)

Algorithm 1 Classifier guided diffusion sampling, given a diffusion model $(\mu_\theta(x_t), \Sigma_\theta(x_t))$, classifier $p_\phi(y|x_t)$, and gradient scale s.

Input: class label y, gradient scale s
$x_T \leftarrow$ sample from $\mathcal{N}(0, \mathbf{I})$
for all t from T to 1 **do**
  $\mu, \Sigma \leftarrow \mu_\theta(x_t), \Sigma_\theta(x_t)$
  $x_{t-1} \leftarrow$ sample from $\mathcal{N}(\mu + s\Sigma \nabla_{x_t} \log p_\phi(y|x_t), \Sigma)$
end for
return x_0

Classifier Guidance

The score-based formulation allows us to easily modify the DDIM sampling for classifier guidance. Specifically:

$$\begin{aligned}\nabla_{x_t} \log(p_\theta(x_t)p_\phi(y|x_t)) &= \nabla_{x_t} \log p_\theta(x_t) + \nabla_{x_t} \log p_\phi(y|x_t) \\ &= -\frac{1}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t) + \nabla_{x_t} \log p_\phi(y|x_t)\end{aligned}$$

So we can derive an updated noise predictor function to use for classifier guidance:

$$\hat{\epsilon}(x_t) := \epsilon_\theta(x_t) - \sqrt{1-\bar{\alpha}_t}\,\nabla_{x_t} \log p_\phi(y|x_t)$$

Classifier details

Classifier is simply the downsampling trunk of the U-net with an attention pooling at the 8x8 layer to produce the final output.

Classifier is trained on the noisy images, along with random crops to reduce overfitting

Scaling the classifier gradient to above 1 tends to result in better results

- Higher fidelity but lower diversity

Classifier Guidance Results

With a high enough scale, a guided unconditional model can get quite close in FID to an unguided conditional model (but still worse than guided conditional)

Figure 3: Samples from an unconditional diffusion model with classifier guidance to condition on the class "Pembroke Welsh corgi". Using classifier scale 1.0 (left; FID: 33.0) does not produce convincing samples in this class, whereas classifier scale 10.0 (right; FID: 12.0) produces much more class-consistent images.

Conditional	Guidance	Scale	FID	sFID	IS	Precision	Recall
✗	✗		26.21	**6.35**	39.70	0.61	0.63
✗	✓	1.0	33.03	6.99	32.92	0.56	**0.65**
✗	✓	10.0	**12.00**	10.40	**95.41**	**0.76**	0.44
✓	✗		10.94	6.02	100.98	0.69	**0.63**
✓	✓	1.0	**4.59**	**5.25**	186.70	0.82	0.52
✓	✓	10.0	9.11	10.93	**283.92**	**0.88**	0.32

Table 4: Effect of classifier guidance on sample quality. Both conditional and unconditional models were trained for 2M iterations on ImageNet 256×256 with batch size 256.

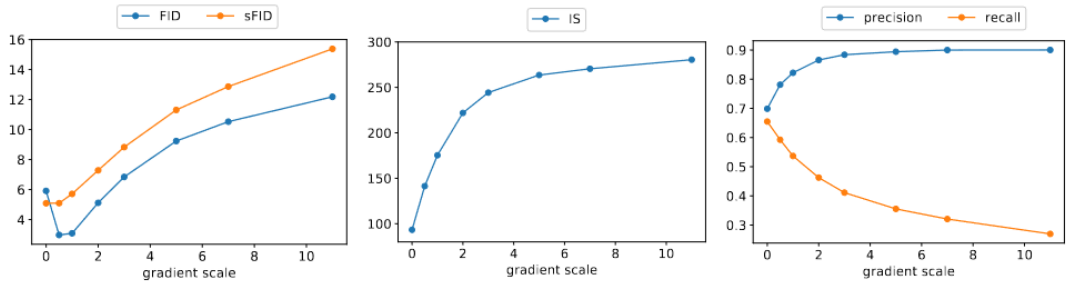


Figure 4: Change in sample quality as we vary scale of the classifier gradients for a class-conditional ImageNet 128×128 model.

Results

Model	FID	sFID	Prec	Rec
LSUN Bedrooms 256×256				
DCTransformer† [42]	6.40	6.66	0.44	**0.56**
DDPM [25]	4.89	9.07	0.60	0.45
IDDPM [43]	4.24	8.21	0.62	0.46
StyleGAN [27]	2.35	6.62	0.59	0.48
ADM (dropout)	**1.90**	**5.59**	**0.66**	0.51
LSUN Horses 256×256				
StyleGAN2 [28]	3.84	6.46	0.63	0.48
ADM	2.95	**5.94**	0.69	**0.55**
ADM (dropout)	**2.57**	6.81	**0.71**	**0.55**
LSUN Cats 256×256				
DDPM [25]	17.1	12.4	0.53	0.48
StyleGAN2 [28]	7.25	**6.33**	0.58	0.43
ADM (dropout)	**5.57**	6.69	**0.63**	**0.52**
ImageNet 64×64				
BigGAN-deep* [5]	4.06	3.96	**0.79**	0.48
IDDPM [43]	2.92	**3.79**	0.74	0.62
ADM	2.61	**3.77**	0.73	0.63
ADM (dropout)	**2.07**	4.29	0.74	**0.63**

Model	FID	sFID	Prec	Rec
ImageNet 128×128				
BigGAN-deep [5]	6.02	7.18	**0.86**	0.35
LOGAN† [68]	3.36			
ADM	5.91	**5.09**	0.70	**0.65**
ADM-G (25 steps)	5.98	7.04	0.78	0.51
ADM-G	**2.97**	**5.09**	0.78	0.59
ImageNet 256×256				
DCTransformer† [42]	36.51	8.24	0.36	**0.67**
VQ-VAE-2†‡ [51]	31.11	17.38	0.36	0.57
IDDPM‡ [43]	12.26	5.42	0.70	0.62
SR3†‡ [53]	11.30			
BigGAN-deep [5]	6.95	7.36	**0.87**	0.28
ADM	10.94	6.02	0.69	0.63
ADM-G (25 steps)	5.44	5.32	0.81	0.49
ADM-G	**4.59**	**5.25**	0.82	0.52
ImageNet 512×512				
BigGAN-deep [5]	8.43	8.13	**0.88**	0.29
ADM	23.24	10.19	0.73	**0.60**
ADM-G (25 steps)	8.41	9.67	0.83	0.47
ADM-G	**7.72**	**6.57**	0.87	0.42

Table 5: Sample quality comparison with state-of-the-art generative models for each task. ADM refers to our ablated diffusion model, and ADM-G additionally uses classifier guidance. LSUN diffusion models are sampled using 1000 steps (see Appendix J). ImageNet diffusion models are sampled using 250 steps, except when we use the DDIM sampler with 25 steps. *No BigGAN-deep model was available at this resolution, so we trained our own. †Values are taken from a previous paper, due to lack of public models or samples. ‡Results use two-resolution stacks.

Results

Two-stage pipeline (originally proposed in iDDPM, in appendix): guidance at lower resolution and then upsample gets best 512x512 results

Model	S_{base}	$S_{upsample}$	FID	sFID	IS	Precision	Recall
ImageNet 256×256							
ADM	250		10.94	6.02	100.98	0.69	**0.63**
ADM-U	250	250	7.49	**5.13**	127.49	0.72	**0.63**
ADM-G	250		4.59	5.25	186.70	0.82	0.52
ADM-G, ADM-U	250	250	**3.94**	6.14	**215.84**	**0.83**	0.53
ImageNet 512×512							
ADM	250		23.24	10.19	58.06	0.73	0.60
ADM-U	250	250	9.96	**5.62**	121.78	0.75	**0.64**
ADM-G	250		7.72	6.57	172.71	**0.87**	0.42
ADM-G, ADM-U	25	25	5.96	12.10	187.87	0.81	0.54
ADM-G, ADM-U	250	25	4.11	9.57	219.29	0.83	0.55
ADM-G, ADM-U	250	250	**3.85**	5.86	**221.72**	0.84	0.53

Results

Figure 1: Selected samples from our best ImageNet 512×512 model (FID 3.85)