Diffusion Model Study Group #1
DDPM paper

Tanishq Abraham

EleutherAI

8/27/2022

Introduction

- Weekly study group
- Everyone reads a paper (or a few), bring questions!
- One person will present
- Notebooks in preparation
- Meetings are recorded
- Resources including slides will be put on GitHub
- Feel free to ask questions or interrupt at any time!

Background

Generative models in general

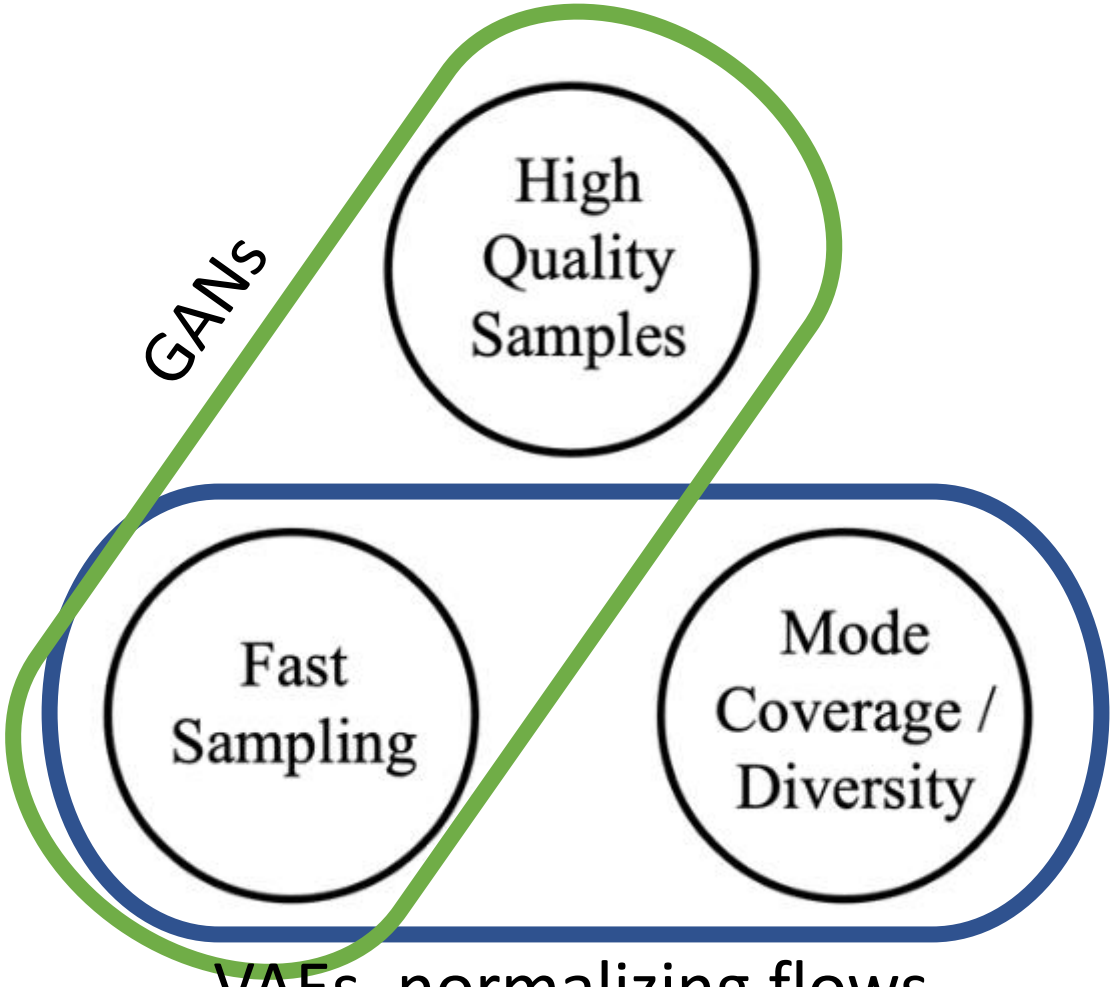
Task: estimate the true data distribution $p(\mathbf{x})$ given observed samples

Two approaches:

1. Explicit likelihood methods (variational autoencoders, normalizing flows, etc.)

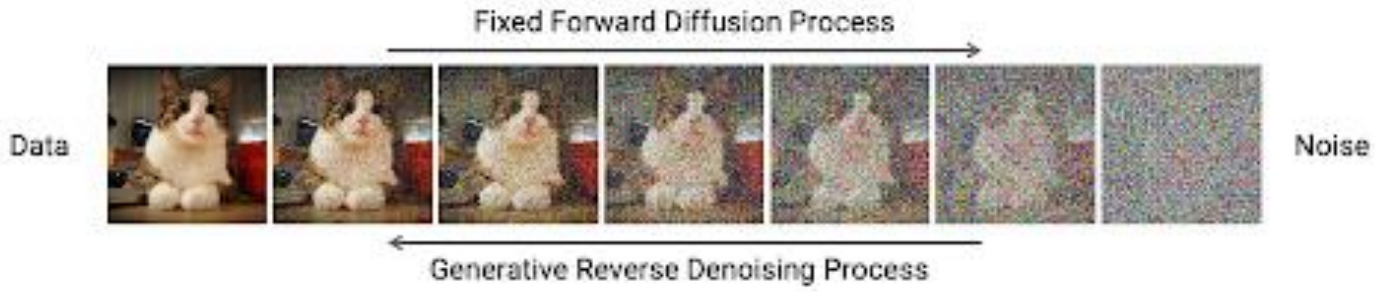
2. Implicit likelihood methods (generative adversarial networks)

The Generative Trilemma



What are diffusion models? – a short summary

- Originally invented in 2015 by Jascha Sohl-Dickstein et al.
- Gained prominence thanks to DDPMs in 2020 – first demo of high quality samples
- A short, simplified summary: We train a neural network to iteratively denoise a sample starting from pure noise, which can be shown to be equivalent to sampling from the estimated data distribution
- Still a very new field!



Diving into the DDPM paper

The forward process

- $\mathbf{x}_0$ - the original image, before the destructive, noise-adding process
- $\mathbf{x}_T$ - the final Gaussian noise$\mathcal{N}(\mathbf{x_T}; \mathbf{0}, \mathbf{I})$, after the noise-adding process
- Forward diffusion process – go from $\mathbf{x}_0$ to $\mathbf{x}_T$ by iteratively adding noise with Markov chain
- $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_t, \ldots, \mathbf{x}_{T-1}$ – intermediate *latent variables*
- Markov chain – stochastic *memoryless process*
  - Distribution for $\mathbf{x}_t$ only depends on $\mathbf{x}_{t-1}$

The forward process (continued)

Distribution for $\mathbf{x}_t$ in the forward process:

β_t is the variance at time t (predefined schedule)

Example:

Properties of forward process

- Since the forward process is a Markov chain, the following holds true:

$$q(\mathbf{x}_{1:T} \mid \mathbf{x}_0) := \prod_{t=1}^{T} q(\mathbf{x}_t \mid \mathbf{x}_{t-1})$$

Since the distributions are Gaussians, the following holds true:

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{\overline{\alpha}_t}\mathbf{x}_0, (1-\overline{\alpha}_t)\mathbf{I}\right)$$

where $\overline{\alpha}_t = \prod_{i=1}^{T} \alpha_i$ and $\alpha_t = 1 - \beta_t$

Can sample $\mathbf{x}_t$ at any arbitrary timestep t!

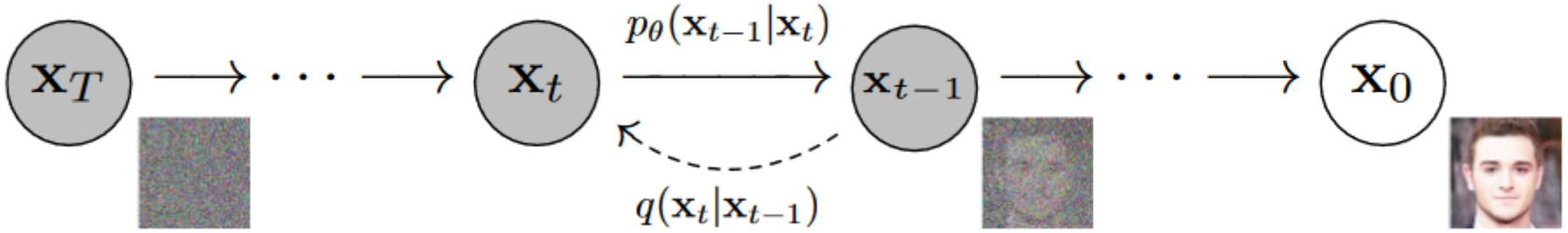
The reverse process

Reverse diffusion process – Markov chain going from $\mathbf{x}_T$ to $\mathbf{x}_0$ with *learned* Gaussian transitions

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

$\boldsymbol{\mu}_\theta(\mathbf{x}_t, t)$ and $\Sigma_\theta(\mathbf{x}_t, t)$ are actually **neural networks** parameterized by weights $\theta \in \Theta$ that we *train*!

After training, sampling is similar to the forward process (except opposite)



How to train diffusion models?

Want to find the most optimal parameters of model that maximize likelihood of training data:

$$\theta^\star = \arg\max_{\theta \in \Theta} p_\theta(\mathbf{x}_{0:T})$$

Therefore we must minimize:

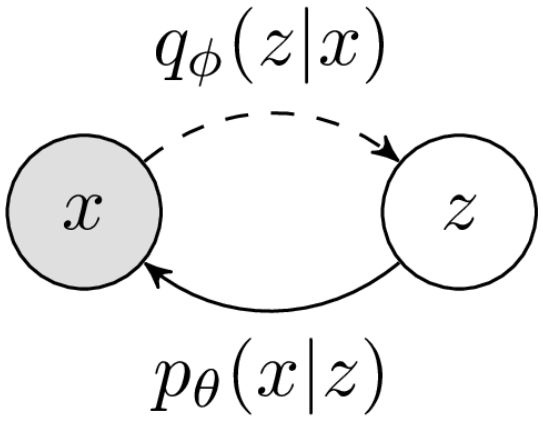
$$\mathcal{L} = E_{\mathbf{x}_0 \sim p_{data}}[-\log p_\theta(\mathbf{x}_0)]$$

This requires:

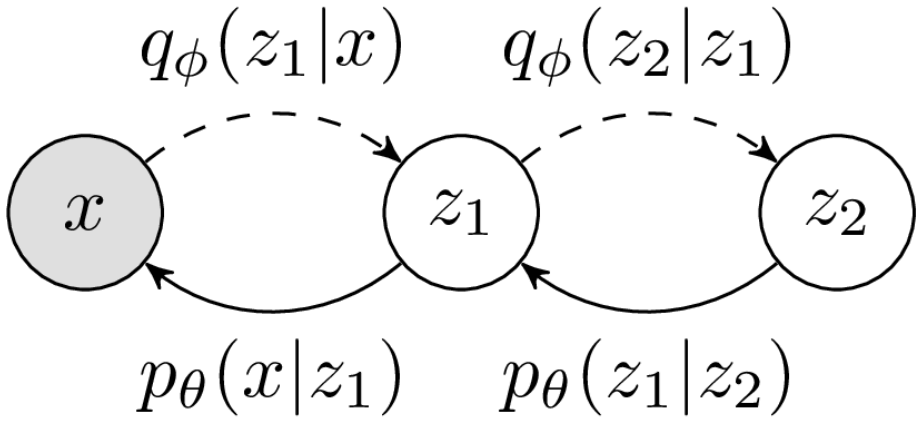
$$p_\theta(\mathbf{x}_0) = \int p_\theta(\mathbf{x}_0|\mathbf{x}_{1:T})\mathrm{d}\mathbf{x}_{1:T}$$

which is intractable!

Comparison to a variational autoencoder

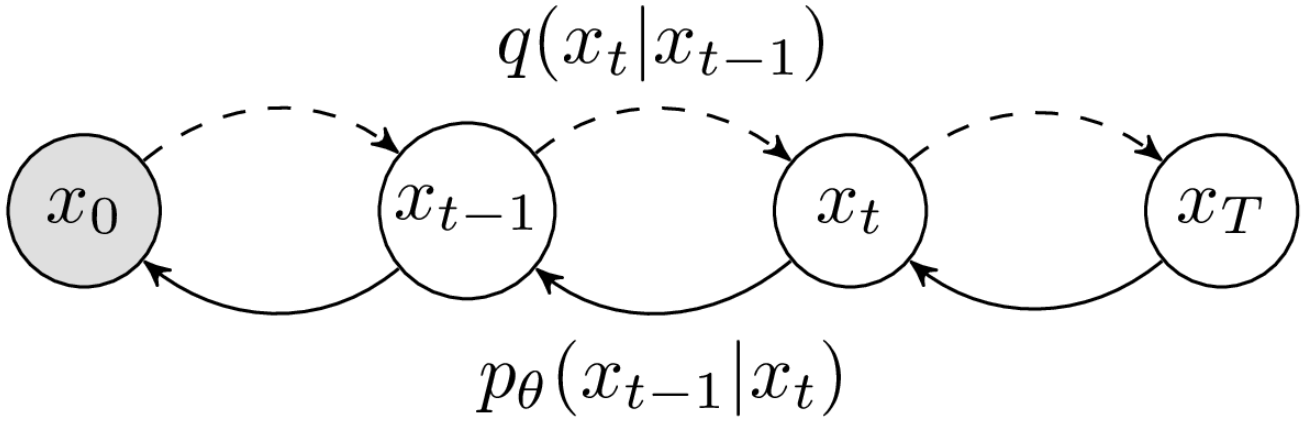


Variational autoencoder



Hierarchical variational autoencoder

A diffusion model can be considered as a hierarchical VAE with a fixed Gaussian encoder



Diffusion model

The evidence lower bound objective (ELBO)

A trick from Bayesian inference and used in VAEs

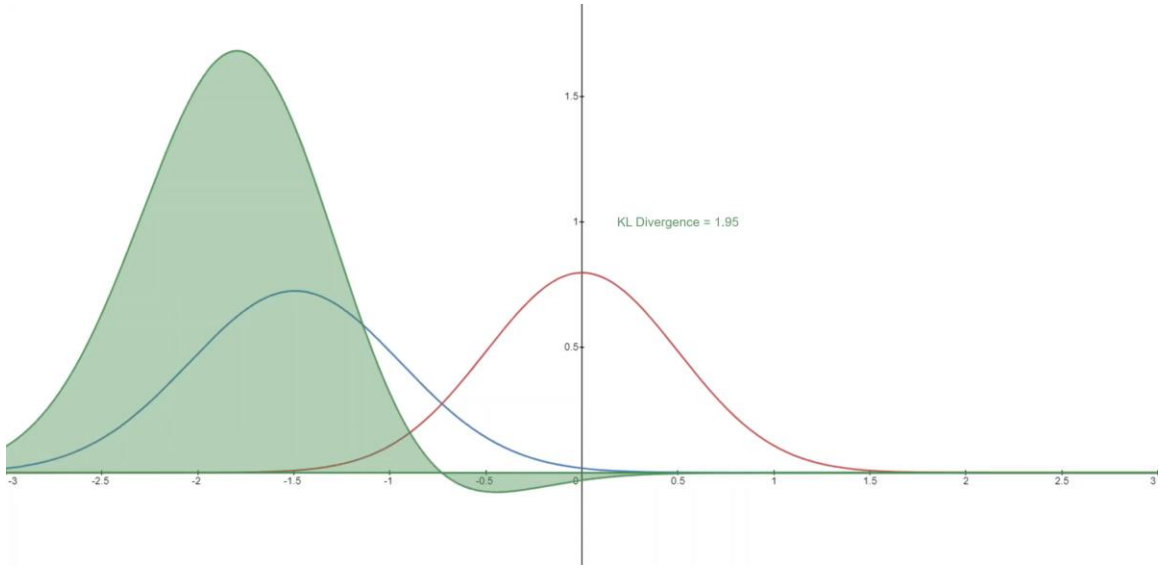
$$\mathbb{E}_{\mathbf{x}_0 \sim p_{data}}[-\log p_\theta(\mathbf{x}_0)] \leq \mathbb{E}_{q(\mathbf{x}_{0:T})}\left[-\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})}\right]$$

$$\mathcal{L}_{ELBO} := \mathbb{E}_{q(\mathbf{x}_{0:T})}\left[-\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})}\right]$$

Kullback-Liebler Divergence

*Asymmetric m*easure of how much one probability distribution differs from another:

$$D_{\mathrm{KL}}(P \parallel Q) = \int_{-\infty}^{\infty} p(x) \log\left(\frac{p(x)}{q(x)}\right) dx$$



With Gaussians: can be calculated with a closed-form expression!

Through some simplification…

$$\mathcal{L}_{ELBO} = L_0 + L_1 + \cdots + L_{T-1} + L_T$$

$$L_0 = -\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)$$

$$L_{t-1} = D_{KL}\big(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \| p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)\big)$$

$$L_T = D_{KL}\big(q(\mathbf{x}_T|\mathbf{x}_0) \| p(\mathbf{x}_T)\big)$$

Various distribution matching terms!

What is $q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0)$?

Given original image $\mathbf{x}_0$ and noisy image $\mathbf{x}_t$, get distribution for some intermediate noisy image $\mathbf{x}_{t-1}$

Through Bayes' Rule, can be derived:

$$q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_{t-1}; \tilde{\mu}(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I}\right)$$

where

$$\tilde{\mu}_t(\mathbf{x_t}, \mathbf{x_0}) := \frac{\sqrt{\overline{\alpha}_{t-1}}\beta_t}{1-\overline{\alpha}_t}\mathbf{x_0} + \frac{\sqrt{\overline{\alpha}_t}(1-\overline{\alpha}_{t-1})}{1-\overline{\alpha}_t}\mathbf{x_t}, \tilde{\beta}_t := \frac{1-\overline{\alpha}_{t-1}}{1-\overline{\alpha}_t}\beta_t$$

Why $q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0)$?

We could use $q(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$ instead

But, there are many possible $\mathbf{x}_{t-1}$, some more likely than others, leading to a high variance for this term

If we knew $\mathbf{x}_0$, we could limit our estimate of $\mathbf{x}_{t-1}$ to more likely cases

This is sometimes referred to as a *variance reduction* step

To recap the training objective:

Three loss terms:

$L_0 = -\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)$ → Decoder reconstruction term

$L_{t-1} = D_{KL}\big(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)\|p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)\big)$ → match estimated $\mathbf{x}_{t-1}$ distribution to actual $\mathbf{x}_{t-1}$ distribution from forward process

$L_T = D_{KL}\big(q(\mathbf{x}_T|\mathbf{x}_0)\|p(\mathbf{x}_T)\big)$ → match forward process to prior (no trainable terms)

All of that applies for “all” diffusion models!

DDPM assumption

Assume forward process variances β_t are fixed to constants.

Assume reverse process variances are also fixed to time-dependent constants:

$$\Sigma_\theta(\mathbf{x}_t, t) = \sigma_t^2 \mathbf{I}$$

We set $\sigma_t^2 = \beta_t$ although other choices are possible too.

Examining L_T

$$L_T = D_{KL}\big(q(\mathbf{x}_T|\mathbf{x}_0)\|p(\mathbf{x}_T)\big)$$

We defined $p(\mathbf{x}_T)$ to be a isotropic Gaussian

$q(\mathbf{x}_T|\mathbf{x}_0)$ is also completely known due to the fixed forward process

Is a constant, can be ignored during training

If $T \to \infty$, $L_T \to 0$

Examining L_0

For some reason I don’t understand, it’s completely ignored!

Examining L_{t-1}

We can simplify the term further thanks to closed-form expression of KL divergence between two Gaussians:

$$L_{T-1} = D_{KL}\big(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)\|p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)\big)$$
$$= \mathbb{E}_q\left[\frac{1}{2\sigma_t^2}\|\tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) - \mu_\theta(\mathbf{x}_t, t)\|^2\right] + C$$

MSE loss between the reverse process posterior mean and the forward process posterior mean.

Train neural network for the reverse process mean to simply predict the mean we observe for the forward process!

Reparameterize as noise “prediction”

While valid, the previous approach led to unstable training, so more reparameterization

$$\mathbf{x}_t(\mathbf{x}_0, \epsilon) = \sqrt{\overline{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\overline{\alpha}_t}\epsilon, \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

$$\mathbb{E}_{\mathbf{x}_0,\epsilon}\left[\frac{1}{2\sigma_t^2}\left\|\tilde{\boldsymbol{\mu}}_t\left(\mathbf{x}_t\left(\mathbf{x}_0,\boldsymbol{\epsilon}\right), \frac{1}{\sqrt{\bar{\alpha}_t}}\left(\mathbf{x}_t\left(\mathbf{x}_0,\boldsymbol{\epsilon}\right) - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}\right)\right) - \boldsymbol{\mu}_\theta\left(\mathbf{x}_t\left(\mathbf{x}_0,\boldsymbol{\epsilon}\right), t\right)\right\|^2\right]$$

Expanding $\tilde{\mu}_t$ and simplifying:

$$L_{t-1} - C = \mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\frac{1}{2\sigma_t^2}\left\|\frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t\left(\mathbf{x}_0,\boldsymbol{\epsilon}\right) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}\right) - \boldsymbol{\mu}_\theta\left(\mathbf{x}_t\left(\mathbf{x}_0,\boldsymbol{\epsilon}\right), t\right)\right\|^2\right]$$

Reparameterize as “noise” prediction (cont.)

Let’s make the “noise” a trainable function $\boldsymbol{\epsilon_\theta}(\mathbf{x}_t, t)$, we can redefine $\boldsymbol{\mu_\theta}(\mathbf{x}_t, t)$ as such:

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

$$E_{\mathbf{x_0},\boldsymbol{\epsilon}}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)}\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon_\theta}\left(\sqrt{\overline{\alpha_t}}\mathbf{x}_0 + \sqrt{1-\overline{\alpha_t}}\boldsymbol{\epsilon}, t\right)\right\|^2\right]$$

Simplified training objective

$$L(\theta) = E_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\boldsymbol{\theta}}\left(\sqrt{\overline{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}, t\right)\right\|^2\right]$$

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2$$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

Experimental setup

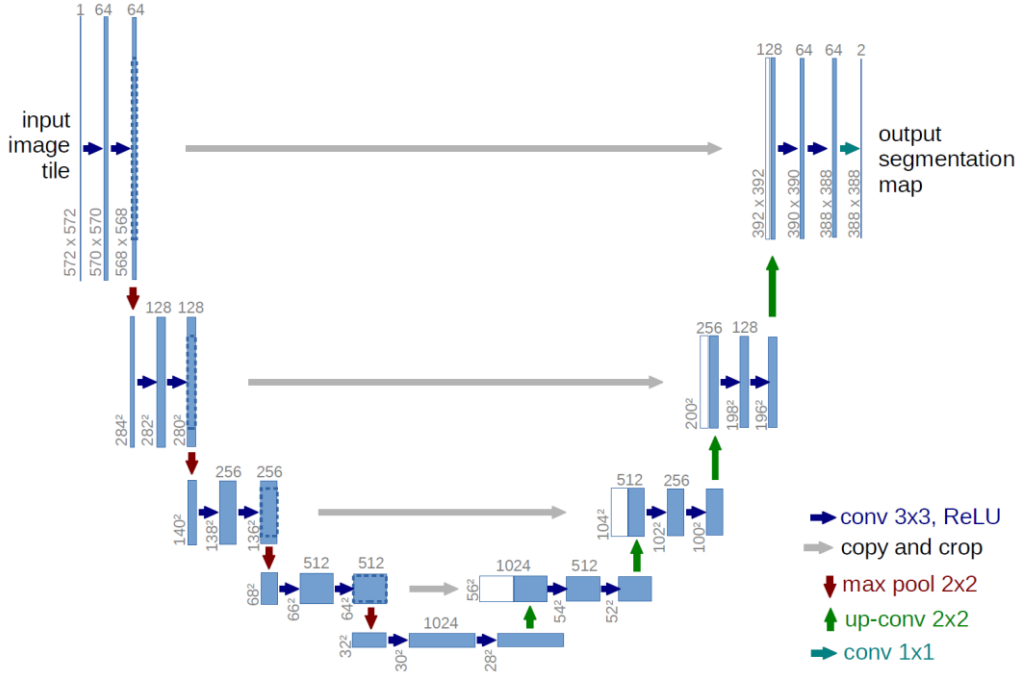
Timesteps $T = 100$

$\beta_1 = 10^{-4}$ linearly increases to $\beta_T = 0.02$

Model is a modified U-net, parameters shared across time, timesteps specified via a sinusoidal position embedding

Horizontal flips for CIFAR10

Training with Adam, used EMA



Results

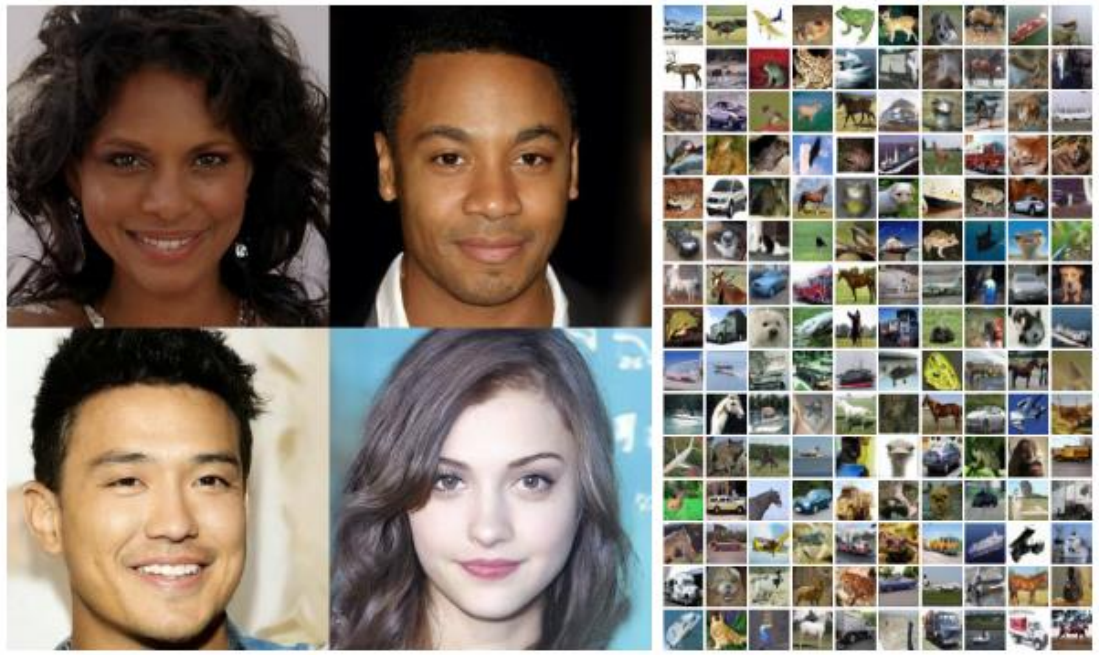


Figure 1: Generated samples on CelebA-HQ 256 × 256 (left) and unconditional CIFAR10 (right)

Table 1: CIFAR10 results. NLL measured in bits/dim.

Model	IS	FID	NLL Test (Train)
Conditional			
EBM [11]	8.30	37.9	
JEM [17]	8.76	38.4	
BigGAN [3]	9.22	14.73	
StyleGAN2 + ADA (v1) [29]	**10.06**	**2.67**	
Unconditional			
Diffusion (original) [53]			$\leq$ 5.40
Gated PixelCNN [59]	4.60	65.93	3.03 (2.90)
Sparse Transformer [7]			**2.80**
PixelIQN [43]	5.29	49.46	
EBM [11]	6.78	38.2	
NCSNv2 [56]		31.75	
NCSN [55]	8.87$\pm$0.12	25.32	
SNGAN [39]	8.22$\pm$0.05	21.7	
SNGAN-DDLS [4]	9.09$\pm$0.10	15.42	
StyleGAN2 + ADA (v1) [29]	**9.74** $\pm$ 0.05	3.26	
Ours (L, fixed isotropic $\mathbf{\Sigma}$)	7.67$\pm$0.13	13.51	$\leq$ 3.70 (3.69)
Ours (L_{simple})	9.46$\pm$0.11	**3.17**	$\leq$ 3.75 (3.72)

References

- Denoising Diffusion Probabilistic Models by Ho et al.
- Introduction to Diffusion Models for Machine Learning by Ryan O’Connor, AssemblyAI
- What are Diffusion Models? by Lilian Weng

Appendix

Derivation of forward process distribution at arbitrary timestep

A nice property of the above process is that we can sample $\mathbf{x}_t$ at any arbitrary time step t in a closed form using reparameterization trick. Let $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{T} \alpha_i$:

$$
\begin{aligned}
\mathbf{x}_t &= \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\mathbf{z}_{t-1} && \text{;where } \mathbf{z}_{t-1}, \mathbf{z}_{t-2}, \cdots \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \\
&= \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\bar{\mathbf{z}}_{t-2} && \text{;where } \bar{\mathbf{z}}_{t-2} \text{ merges two Gaussians (*).} \\
&= \dots \\
&= \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\mathbf{z} \\
q(\mathbf{x}_t|\mathbf{x}_0) &= \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})
\end{aligned}
$$

(*) Recall that when we merge two Gaussians with different variance, $\mathcal{N}(\mathbf{0}, \sigma_1^2\mathbf{I})$ and $\mathcal{N}(\mathbf{0}, \sigma_2^2\mathbf{I})$, the new distribution is $\mathcal{N}(\mathbf{0}, (\sigma_1^2 + \sigma_2^2)\mathbf{I})$. Here the merged standard deviation is $\sqrt{(1-\alpha_t) + \alpha_t(1-\alpha_{t-1})} = \sqrt{1-\alpha_t\alpha_{t-1}}$.

Derivation of ELBO for diffusion models

$$
\begin{aligned}
L_{\text{CE}} &= -\mathbb{E}_{q(\mathbf{x}_0)} \log p_\theta(\mathbf{x}_0) \\
&= -\mathbb{E}_{q(\mathbf{x}_0)} \log \Big(\int p_\theta(\mathbf{x}_{0:T}) d\mathbf{x}_{1:T} \Big) \\
&= -\mathbb{E}_{q(\mathbf{x}_0)} \log \Big(\int q(\mathbf{x}_{1:T}|\mathbf{x}_0) \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} d\mathbf{x}_{1:T} \Big) \\
&= -\mathbb{E}_{q(\mathbf{x}_0)} \log \Big(\mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \Big) \\
&\leq -\mathbb{E}_{q(\mathbf{x}_{0:T})} \log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \\
&= \mathbb{E}_{q(\mathbf{x}_{0:T})} \Big[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})} \Big] = L_{\text{VLB}}
\end{aligned}
$$

Derivation of $q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0)$

Using Bayes' rule, we have:

$$
\begin{aligned}
q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) &= q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)\frac{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)} \\
&\propto \exp\Big(-\frac{1}{2}\big(\frac{(\mathbf{x}_t - \sqrt{\alpha_t}\mathbf{x}_{t-1})^2}{\beta_t} + \frac{(\mathbf{x}_{t-1} - \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0)^2}{1-\bar{\alpha}_{t-1}} - \frac{(\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\mathbf{x}_0)^2}{1-\bar{\alpha}_t}\big)\Big) \\
&= \exp\Big(-\frac{1}{2}\big(\frac{\mathbf{x}_t^2 - 2\sqrt{\alpha_t}\mathbf{x}_t\mathbf{x}_{t-1} + \alpha_t\mathbf{x}_{t-1}^2}{\beta_t} + \frac{\mathbf{x}_{t-1}^2 - 2\sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0\mathbf{x}_{t-1} + \bar{\alpha}_{t-1}\mathbf{x}_0^2}{1-\bar{\alpha}_{t-1}} - \frac{(\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\mathbf{x}_0)^2}{1-\bar{\alpha}_t}\big)\Big) \\
&= \exp\Big(-\frac{1}{2}\big((\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}})\mathbf{x}_{t-1}^2 - (\frac{2\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t + \frac{2\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_0)\mathbf{x}_{t-1} + C(\mathbf{x}_t, \mathbf{x}_0)\big)\Big)
\end{aligned}
$$

where $C(\mathbf{x}_t, \mathbf{x}_0)$ is some function not involving $\mathbf{x}_{t-1}$ and details are omitted. Following the standard Gaussian density function, the mean and variance can be parameterized as follows (recall that $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{T} \alpha_i$):

$$
\begin{aligned}
\tilde{\beta}_t &= 1/(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}) = 1/(\frac{\alpha_t - \bar{\alpha}_t + \beta_t}{\beta_t(1-\bar{\alpha}_{t-1})}) = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t} \cdot \beta_t \\
\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) &= (\frac{\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_0)/(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}) \\
&= (\frac{\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_0)\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t} \cdot \beta_t \\
&= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0
\end{aligned}
$$

Thanks to the nice property, we can represent $\mathbf{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\mathbf{z}_t)$ and plug it into the above equation and obtain:

$$
\begin{aligned}
\tilde{\boldsymbol{\mu}}_t &= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\mathbf{z}_t) \\
&= \frac{1}{\sqrt{\alpha_t}}\Big(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\mathbf{z}_t\Big)
\end{aligned}
$$