Diffusion Models

Patrick Yin

July 2, 2024

1 Credits

These notes were written by parsing the papers in the references as well as the following two blogs:

- https://lilianweng.github.io/posts/2021-07-11-diffusion-models/
- https://yang-song.net/blog/2021/score/

2 What are Diffusion Models?

Diffusion models are a form of generative model which circumvents issues other generative models face such as GANs, VAEs, and Flow-based models (although they come along with their own isssues). They show impressive performance in image generation, in-painting, and manipulation. They've also recently been gaining traction in robotics, where diffusion models been used as a richer policy class which claims better performance than prior methods.

3 Forward Diffusion Process

The diffusion process is a markov chain, where the next timestep only depends on the previous timestep. Given $x_0 \sim q(x)$, we define the forward diffusion process as adding Gaussian noise to the sample in T steps to produce $x_1, ..., x_T$ controlled by variance schedule $\{\beta_t \in (0,1)\}_{t=1}^T$:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t\mathbb{I})$$

$$q(x_{1:T}|x_0) = \prod_{t=1}^{T} q(x_t|x_{t-1})$$

When $T \to \infty$, x_T is equivalent to an isotropic Gaussian distribution.

4 Reverse Diffusion Process

4.1 Derivation of Training Objective

If we can reverse the forward process and sample from $q(x_{t-1}|x_t)$, we can recreate the true sample from Gaussian noise. In the limit of infinitesimal step sizes, the true reverse process will have the same functional form as the forward process [Fel49]. As a result, with $\beta_t << 1$, $q(x_{t-1}|x_t)$ will also be Gaussian. We learn a model p_θ to approximate this posterior:

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \boldsymbol{\mu}_\theta(x_t, t), \boldsymbol{\Sigma}_\theta(x_t, t))$$

Note that the reverse process is also a markov chain, so

$$p_\theta(x_{0:T}) = p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t)$$

We would like to maximize $p_\theta(x_0) = \int p_\theta(x_{0:T})\, dx_{1:T}$, but this intractable since it requires us to marginalize over all the ways we could have arrived at x_0 starting from a noise sample. Instead, let's view p_θ as a latent variable model where x_0 is our observed variable and $x_{1:T}$ is our latent variables. We can estimate $p_\theta(x_0)$ with a variational lower bound:

$$\begin{aligned} -\log p_\theta(x_0) &\leq -\mathbb{E}_q[\log p_\theta(x_0|x_{1:T})] + D_{KL}(q(x_{1:T}|x_0)\|p_\theta(x_{1:T})) \\ &= \mathbb{E}_q[\log \frac{q(x_{1:T}|x_0)}{p_\theta(x_{0:T})}] = L_{VLB} \end{aligned}$$

We can further decompose the objective into

$$\begin{aligned} L_{VLB} &= \mathbb{E}_q[\log \frac{q(x_{1:T}|x_0)}{p_\theta(x_{0:T})}] \\ &= \mathbb{E}_q[-\log p_\theta(x_T) + \sum_{t=1}^{T} \log \frac{q(x_t|x_{t-1})}{p_\theta(x_{t-1}|x_t)}] \\ &= \mathbb{E}_q[D_{KL}(q(x_T|x_0)\|p_\theta(x_T)) + \sum_{t=2}^{T} D_{KL}(q(x_{t-1}|x_t, x_0)\|p_\theta(x_{t-1}|x_t)) - \log p_\theta(x_0|x_1)] \\ &= \mathbf{L}_T + \sum_{t=2}^{T} \mathbf{L}_{t-1} + \mathbf{L}_0 \end{aligned}$$

by applying Bayes' rule and additionally conditioning on x_0. We want to condition on x_0 because the reverse conditional probability $q(x_{t-1}|x_t, x_0)$ is Gaussian when conditioned on known x_0. As a result, the two terms in KL divergence in $\mathbf{L}_t$ are both Gaussians, so we can evaluate it in a Rao-Blackwellized fashion with closed form expressions instead of high variance Monte Carlo estimates.

Since $q(x_T|x_0)$ and $p_\theta(x_T)$ are fixed, $\mathbf{L}_T$ is constant and can be ignored during training. In DDPM [HJA20], the authors use a separate discrete decoder for $\mathbf{L}_0$ derived from $\mathcal{N}(x_0; \boldsymbol{\mu}_\theta(x_1, 1), \boldsymbol{\Sigma}_\theta(x_1, 1))$.

4.2 Reparameterization and Simplification of Objective

In DDPM [HJA20], the authors elect to fix $\mathbf{\Sigma}_\theta(x_t, t)$ to time-specific constants σ_t due to better empirical performance and training stability. They also elect to reparameterize the loss function in terms of that noise that was added and try to learn the noise rather than $\boldsymbol{\mu}_\theta(x_t, t)$. More specifically, let $\alpha_t = 1 - \beta_t$, $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$, and $\epsilon \sim \mathcal{N}(0, I)$. Recall that

$$\begin{aligned} x_t &= \sqrt{\alpha_t} x_{t-1} + \sqrt{1-\alpha_t}\epsilon \\ &= \sqrt{\alpha_t \alpha_{t-1}} x_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\epsilon \\ &= ... \\ &= \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon \end{aligned}$$

so $q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathbf{I})$. Note that conditioned on x_0, $q(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}; \tilde{\mu}(x_t, x_0), \tilde{\beta}_t\mathbf{I})$. We want to find a closed form expression for $\tilde{\mu}$ since that would be the target our network would regress to if we were optimizing for $\boldsymbol{\mu}_\theta(x_t, t)$. By Bayes' rule, we have:

$$\begin{aligned} q(x_{t-1}|x_t, x_0) &= q(x_t|x_{t-1}, x_0)\frac{q(x_{t-1}|x_0)}{q(x_t|x_0)} \\ &\propto \exp\Big(-\frac{1}{2}\big(\frac{(x_t - \sqrt{\alpha_t}x_{t-1})^2}{\beta_t} + \frac{(x_{t-1} - \sqrt{\bar{\alpha}_{t-1}}x_0)^2}{1-\bar{\alpha}_{t-1}} - \frac{(x_t - \sqrt{\bar{\alpha}}_t x_0)^2}{1-\bar{\alpha}_t}\big)\Big) \\ &= \exp\Big(-\frac{1}{2}\big((\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}})x_{t-1}^2 - (\frac{2\sqrt{\alpha_t}}{\beta_t}x_t + \frac{2\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}x_0)x_{t-1} + \mathbf{C}(x_t, x_0)\big)\Big) \end{aligned}$$

where $C(x_t, x_0)$ is some function that doesn't involve x_{t-1}. Therefore, we can derive the mean and variance of the Gaussian to be:

$$\begin{aligned} \tilde{\beta}_t &= 1/(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}) = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha})_t} \cdot \beta_t \\ \tilde{\mu}(x_t, x_0) &= (\frac{2\sqrt{\alpha_t}}{\beta_t}x_t + \frac{2\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}x_0)/(-2\cdot(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}})) \\ &= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0 \end{aligned}$$

Because $x_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \sqrt{1-\bar{\alpha}_t}\epsilon)$, we can plug it in to obtain:

$$\begin{aligned} \tilde{\mu}_t &= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \sqrt{1-\bar{\alpha}_t}\epsilon) \\ &= \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon) \end{aligned}$$

We have successfully parameterized $\boldsymbol{\mu}_\theta(x_t, t)$ in terms of $\boldsymbol{\epsilon}_\theta(x_t, t)$:

$$\boldsymbol{\mu}_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(x_t, t))$$

Since $\mathbf{\Sigma}_\theta(x_t, t)$ is fixed, the KL-divergence between the two Gaussians is simplified down to

$$\begin{aligned}
\mathbf{L}_t &= \mathbb{E}_{x_0,\epsilon}\Big[\frac{1}{2\|\mathbf{\Sigma}_\theta(x_t,t)\|_2^2}\|\tilde{\boldsymbol{\mu}}(x_t,x_0) - \boldsymbol{\mu}_\theta(x_t,t)\|^2\Big] \\
&= \mathbb{E}_{x_0,\epsilon}\Big[\frac{1}{2\|\mathbf{\Sigma}_\theta(x_t,t)\|_2^2}\|\frac{1}{\sqrt{\alpha_t}}(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_t) - \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(x_t,t))\|^2\Big] \\
&= \mathbb{E}_{x_0,\epsilon}\Big[\frac{(1-\alpha_t)^2}{2\alpha_t(1-\bar{\alpha}_t)\|\mathbf{\Sigma}_\theta(x_t,t)\|_2^2}\|\epsilon_t - \boldsymbol{\epsilon}_\theta(x_t,t)\|^2\Big]
\end{aligned}$$

In DDPM [HJA20], the authors found that empirically the diffusion models works better without the weighting term:

$$\begin{aligned}
\mathbf{L}_t &= \mathbb{E}_{t\sim[1,T],x_0,\epsilon_t}\Big[\|\epsilon_t - \boldsymbol{\epsilon}_\theta(x_t,t)\|^2\Big] \\
&= \mathbb{E}_{t\sim[1,T],x_0,\epsilon_t}\Big[\|\epsilon_t - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t, t)\|^2\Big]
\end{aligned}$$

To sample $x_{t-1} \sim p_\theta(x_{t-1}|x_t)$, we can compute $x_{t-1} = \boldsymbol{\mu}_\theta(x_t,t) + \sigma_t\boldsymbol{z} = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(x_t,t)) + \sigma_t\boldsymbol{z}$ where $\boldsymbol{z} \sim \mathcal{N}(0,1)$. This results in two very simple algorithms for training and sampling from the diffusion model:

Algorithm 1 Training

1: **loop**
2: $x_0 \sim q(x_0)$
3: $t \sim \text{Uniform}(1, ..., T)$
4: $\epsilon \sim \mathcal{N}(0, I)$
5: Take gradient descent step on $\nabla_\theta\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t)\|^2$
6: **end loop**

Algorithm 2 Sampling

1: $x_T \sim \mathcal{N}(0,1)$
2: **for** $t = T, ..., 1$ **do**
3: $\boldsymbol{z} \sim \mathcal{N}(0, I)$ if $t > 1$, else $\boldsymbol{z} = 0$
4: $x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}(x_t,t)) + \sigma_t\boldsymbol{z}$
5: **end for**
6: **return** x_0s

5 Connection to Score Matching Models

Langevin dynamics provides an MCMC procedure to sample from a distribution $q(x)$ using only $\nabla_x \log q(x)$:

$$x_t = x_{t-1} + \frac{\delta}{2}\nabla_x \log q(x_{t-1}) + \sqrt{\delta}\epsilon_t$$

where $\epsilon_t \sim \mathcal{N}(0, I)$. As $T \to \infty$, $\epsilon \to 0$, x_T equals the true probability density $q(x)$. We can estimate $\nabla_x \log q(x)$ by learning a score network $s_\theta(x) \approx \nabla_x \log q(x)$ estimated with score matching [SE19].

However, in regions where data density is low, score estimation is less reliable. As a result, the authors of [SE19] perturb the data with noise of different levels and train a noise-conditioned score network to jointly estimate the scores of the perturbed data at the different noise levels. The schedule of increasing noise resembles the forward diffusion process, and it turns out that $\boldsymbol{s}_\theta$ is equivalent to $\boldsymbol{\epsilon}_\theta$ under some scaling factor:

$$\boldsymbol{s}_\theta(x_t, t) \approx \nabla_{x_t} \log q(x_t) = \mathbb{E}_{q(x_0)}[\nabla_{x_t} q(x_t|x_0)] = \mathbb{E}_{q(x_0)}\big[-\frac{\boldsymbol{\epsilon}(x_t, t)}{\sqrt{1-\bar{\alpha}_t}}\big] = -\frac{\boldsymbol{\epsilon}(x_t, t)}{\sqrt{1-\bar{\alpha}_t}}$$

since $q(x_t|x_0) \sim \mathcal{N}(\sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathbf{I})$ and given some Gaussian $x \sim \mathcal{N}(\mu, \sigma^2\mathbf{I})$, $\nabla_x \log p(x) = \nabla_x\big(-\frac{1}{2\sigma^2}(x-\mu)^2\big) = -\frac{\epsilon}{\sigma}$ where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$.

6 Diffusion Policy

Diffusion policy [Chi+23] proposes using diffusion models as the policy class for robot imitation learning and claims benefits of multimodal action distributions, suitability to high-dimensional action spaces, and training stability. They take DDPM and make two major changes:

- Changing output x from images to closed-loop action-sequence prediction, allowing for receding horizon control during inference time
- Conditioning ϵ_θ additionally on input observation O_t, enabling end-to-end training of the vision encoder.

7 Quick Summary

Pros: Diffusion models can be analytically evaluated and cheaply fit data while fitting arbitrary structures in data.

Cons: Diffusion models rely on a long Markov chain of diffusion steps for sample generation, making sampling slower than GANs.

8 Things I didn't cover

- Classifier guided diffusion [DN21]
- Classifier-free diffusion guidance [HS22]
- Variational lower bound on log-likelihood [Kin+21]
- Log-likelihood approximation via probability flow ODEs [Son+20]

References

[Fel49] William Feller. "On the Theory of Stochastic Processes, with Particular Reference to Applications". In: 1949. URL: https://api.semanticscholar.org/CorpusID:121027442.

[SE19] Yang Song and Stefano Ermon. "Generative modeling by estimating gradients of the data distribution". In: *Advances in neural information processing systems* 32 (2019).

[HJA20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models". In: *Advances in neural information processing systems* 33 (2020), pp. 6840–6851.

[Son+20] Yang Song et al. "Score-based generative modeling through stochastic differential equations". In: *arXiv preprint arXiv:2011.13456* (2020).

[DN21] Prafulla Dhariwal and Alexander Nichol. "Diffusion models beat gans on image synthesis". In: *Advances in neural information processing systems* 34 (2021), pp. 8780–8794.

[Kin+21] Diederik Kingma et al. "Variational diffusion models". In: *Advances in neural information processing systems* 34 (2021), pp. 21696–21707.

[HS22] Jonathan Ho and Tim Salimans. "Classifier-free diffusion guidance". In: *arXiv preprint arXiv:2207.12598* (2022).

[Chi+23] Cheng Chi et al. "Diffusion policy: Visuomotor policy learning via action diffusion". In: *arXiv preprint arXiv:2303.04137* (2023).