Sampling conditional distributions with diffusion models and arbitrary conditioning

Antonin Della Noce

Working group on diffusion models

2024, April 11th

Reminder: Score-Based Generative Modeling with SDEs

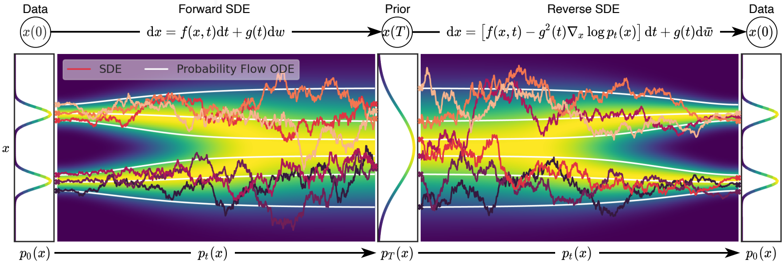
Let $x_{1:N} \sim \mu^{\otimes N}$, where $\mu \in \mathcal{P}(\mathbb{R}^d)$ represents an unknown probability distribution.
Goal: To sample a new data point $x_{N+1} \sim \mu$.

Ref: Song et al., 2020[8]

Reminder: Score-Based Generative Modeling with SDEs

Let $x_{1:N} \sim \mu^{\otimes N}$, where $\mu \in \mathcal{P}(\mathbb{R}^d)$ represents an unknown probability distribution.

Goal: To sample a new data point $x_{N+1} \sim \mu$.



Forward SDE:

$$\begin{cases} \mathrm{d}x_t = -x_t \mathrm{d}t + \sqrt{2}\mathrm{d}B_t, \\ \mathrm{Law}(x_0) = \mu \end{cases}$$

Backward SDE:

$$\begin{cases} \mathrm{d}y_t = \left(y_t + 2\nabla_x \log p_{T-t}(y_t)\right)\mathrm{d}t \\ +\sqrt{2}\mathrm{d}W_t, \\ \mathrm{Law}(y_0) = \mathcal{N}(0, I_d), \\ \mathrm{Law}(x_t) = p_t(x)\mathrm{d}x \end{cases}$$

Ref: Song et al., 2020[8]

Reminder: Score-Based Generative Modeling with SDEs

Learning the score function $s_\theta(t,y) \approx \nabla_x \log p_t(y)$

Consider $T > 0$ and a subdivision $t_{0:n}$ of $[0,T]$.
Solving the discretized SDE

$$\begin{cases} y_0 \sim \mathcal{N}(0, I_d), \\ \forall t \in [0,T], \quad \mathrm{d}y_t = (y_t + 2s_{\hat{\theta}}(T-t, y_t))\,\mathrm{d}t + \sqrt{2}\,\mathrm{d}w_t, \end{cases}$$

results in $y_T \sim \hat{\mu} \approx \mu$ in some sense.
The score $s_\theta(t,x)$ is outputted by the model.
The parameter $\hat{\theta}$ is estimated as follows:

$$\hat{\theta} \in \operatorname{Argmin} \left\{ \hat{\mathcal{I}}_{t_{1:N}}(\theta), \quad \theta \in \mathbb{R}^{d_\theta} \right\},$$

$$\text{where } \hat{\mathcal{I}}_{t_{1:N}}(\theta) = \sum_{j=1}^{n} \sum_{i=1}^{N} \left| s_\theta \left(t_j, e^{-t_j} x_i + \sqrt{1 - e^{-2t_j}} z_i \right) - \frac{z_i}{\sqrt{1 - e^{-2t_j}}} \right|^2 .$$

Ref: Song et al., 2020[8]

Reminder: Image generation from backward SDE

Ref: Song et al., 2020[8]

Motivation: sampling conditional distributions

Let our data be $(x_i, y_i)_{1 \leq i \leq N} \sim \mu^{\otimes N}$ with $\mu \in \mathcal{P}(\mathcal{X} \times \mathcal{Y})$ unknown.
Goal: given $y \in \mathcal{Y}$, sample $x_{N+1} \mid y \sim \mu(\mathrm{d}x \mid y)$ where $\mu(\mathrm{d}x \mid y)$ is **a** conditional distribution of x knowing y.

Ref: Song et al., 2020[8]

Motivation: sampling conditional distributions

Let our data be $(x_i, y_i)_{1 \le i \le N} \sim \mu^{\otimes N}$ with $\mu \in \mathcal{P}(\mathcal{X} \times \mathcal{Y})$ unknown.
Goal: given $y \in \mathcal{Y}$, sample $x_{N+1} \mid y \sim \mu(\mathrm{d}x \mid y)$ where $\mu(\mathrm{d}x \mid y)$ is **a** conditional distribution of x knowing y.

Ref: Song et al., 2020[8]

Outline

Unconditional denoising diffusion probabilistic models

Classifier guidance

Classifier-free guidance

Latent diffusion models

Universal guidance

Outline

Probabilistic graphical model formulation

Discrete Ornstein-Uhlenbeck process (regular time-step Δt, $n\Delta t = T$):

$$\begin{cases} x_0 \sim \mu(\mathrm{d}x_0) \\ \forall k \in [\![1, n]\!], \quad x_k \mid x_{k-1} \sim \mathcal{N}\left(e^{-\Delta t} x_{k-1}, \left(1 - e^{-2\Delta t}\right) I_d\right)(\mathrm{d}x_k) \end{cases}$$

ref: Ho et al., 2020[4]

Probabilistic graphical model formulation

Discrete Ornstein-Uhlenbeck process (regular time-step Δt, $n\Delta t = T$):

$$\begin{cases} x_0 \sim \mu(\mathrm{d}x_0) \\ \forall k \in [\![1, n]\!], \quad x_k \mid x_{k-1} \sim \mathcal{N}\left(e^{-\Delta t} x_{k-1}, \left(1 - e^{-2\Delta t}\right) I_d\right)(\mathrm{d}x_k) \end{cases}$$

Forward diffusion model

Let $(\beta_k)_{1 \leq k \leq n}$ a sequence (variance schedule) in $(0,1)^n$. We consider the discrete Markov process:

$$\begin{cases} x_0 \sim \mu(\mathrm{d}x_0) \\ \forall k \in [\![1, n]\!], \quad x_k \mid x_{k-1} \sim \mathcal{N}\left(\sqrt{1 - \beta_k} x_{k-1}, \beta_k I_d\right)(\mathrm{d}x_k) \end{cases}$$

ref: Ho et al., 2020[4]

Probabilistic graphical model formulation

Discrete Ornstein-Uhlenbeck process (regular time-step Δt, $n\Delta t = T$):

$$\begin{cases} x_0 \sim \mu(\mathrm{d}x_0) \\ \forall k \in [\![1, n]\!], \quad x_k \mid x_{k-1} \sim \mathcal{N}\left(e^{-\Delta t} x_{k-1}, \left(1 - e^{-2\Delta t}\right) I_d\right)(\mathrm{d}x_k) \end{cases}$$

Forward diffusion model

Let $(\beta_k)_{1 \leq k \leq n}$ a sequence (variance schedule) in $(0,1)^n$. We consider the discrete Markov process:

$$\begin{cases} x_0 \sim \mu(\mathrm{d}x_0) \\ \forall k \in [\![1, n]\!], \quad x_k \mid x_{k-1} \sim \mathcal{N}\left(\sqrt{1 - \beta_k} x_{k-1}, \beta_k I_d\right)(\mathrm{d}x_k) \end{cases}$$

$$x_0 \xrightarrow{q_{1|0}(x_1 \mid x_0)\mathrm{d}x_1} x_1 \xrightarrow{q_{2|1}(x_2 \mid x_1)\mathrm{d}x_2} x_2 \quad \cdots \quad x_{n-1} \xrightarrow{q_{n|n-1}(x_n \mid x_{n-1})\mathrm{d}x_n} x_n$$

ref: Ho et al., 2020[4]

Probabilistic graphical model formulation

Discrete Ornstein-Uhlenbeck process (regular time-step Δt, $n\Delta t = T$):

$$\begin{cases} x_0 \sim \mu(\mathrm{d}x_0) \\ \forall k \in [\![1, n]\!], \quad x_k \mid x_{k-1} \sim \mathcal{N}\left(e^{-\Delta t} x_{k-1}, \left(1 - e^{-2\Delta t}\right) I_d\right)(\mathrm{d}x_k) \end{cases}$$

Forward diffusion model

Let $(\beta_k)_{1 \leq k \leq n}$ a sequence (variance schedule) in $(0,1)^n$. We consider the discrete Markov process:

$$\begin{cases} x_0 \sim \mu(\mathrm{d}x_0) \\ \forall k \in [\![1, n]\!], \quad x_k \mid x_{k-1} \sim \mathcal{N}\left(\sqrt{1 - \beta_k} x_{k-1}, \beta_k I_d\right)(\mathrm{d}x_k) \end{cases}$$

$$x_0 \xrightarrow{q_{1|0}(x_1 \mid x_0)\mathrm{d}x_1} x_1 \xrightarrow{q_{2|1}(x_2 \mid x_1)\mathrm{d}x_2} x_2 \quad \cdots \quad x_{n-1} \xrightarrow{q_{n|n-1}(x_n \mid x_{n-1})\mathrm{d}x_n} x_n$$

Example: $n = 1,000$, $\beta_1 = 10^{-4}$, $\beta_n = 0.02$ and

$$\forall k \in [\![1, n]\!], \quad \beta_k = \beta_1 + \frac{k-1}{n-1}(\beta_n - \beta_1)$$

ref: Ho et al., 2020[4]

Forward diffusion

By (**a tedious**) induction, $\forall k \in [\![1, n]\!]$,

$$x_k \mid x_0 \sim \mathcal{N}\left(\sqrt{\alpha_k} x_0, (1-\alpha_k) I_d\right)(\mathrm{d}x_k) =: q_{k|0}(x_k \mid x_0)\mathrm{d}x_k,$$

$$\text{with } \alpha_k = \prod_{\ell=1}^{k} (1-\beta_\ell).$$

ref: Ho et al., 2020[4]

By (**a tedious**) induction, $\forall k \in [\![1, n]\!]$,

$$x_k \mid x_0 \sim \mathcal{N}\left(\sqrt{\alpha_k} x_0, (1-\alpha_k) I_d\right)(\mathrm{d}x_k) =: q_{k|0}(x_k \mid x_0)\mathrm{d}x_k,$$

$$\text{with } \alpha_k = \prod_{\ell=1}^{k}(1-\beta_\ell).$$

Marginal distribution of x_n

$$x_n \sim \int_{\mathbb{R}^d} \mathcal{N}\left(\sqrt{\alpha_n} x_0, (1-\alpha_n) I_d\right)(\mathrm{d}x_n)\mu(\mathrm{d}x_0) =: q_n(x_n)\mathrm{d}x_n.$$

It is crucial to have $q_n(x_n)\mathrm{d}x_n \approx \mathcal{N}(0, I_d)(\mathrm{d}x_n)$ but $q_n(x_n)\mathrm{d}x_n \neq \mathcal{N}(0, I_d)(\mathrm{d}x_n)$.

---

ref: Ho et al., 2020[4]

Motivation for the backward process: informal notation

$$\mathcal{N}(0, I_d) \approx Q_n \circ \cdots \circ Q_1 \mu$$

$$Q_1^{-1} \circ \cdots \circ Q_n^{-1} \mathcal{N}(0, I_d) \approx \mu$$

ref: Ho et al., 2020[4]

Backward process

Motivation for the backward process: informal notation

$$\mathcal{N}(0, I_d) \approx Q_n \circ \cdots \circ Q_1 \mu$$
$$Q_1^{-1} \circ \cdots \circ Q_n^{-1} \mathcal{N}(0, I_d) \approx \mu$$

Distributions of $x_{k\text{-}1} \mid x_k, x_0$

$$k \geq 2, \quad x_{k\text{-}1} \mid x_k, x_0 \sim \frac{q_k(x_k \mid x_{k\text{-}1}) q_{k\text{-}1|0}(x_{k\text{-}1} \mid x_0)}{q_{k|0}(x_k \mid x_0)} \mathrm{d}x_{k\text{-}1}$$

---

ref: Ho et al., 2020[4]

Motivation for the backward process: informal notation

$$\mathcal{N}(0, I_d) \approx Q_n \circ \cdots \circ Q_1 \mu$$
$$Q_1^{-1} \circ \cdots \circ Q_n^{-1} \mathcal{N}(0, I_d) \approx \mu$$

Distributions of $x_{k\text{-}1} \mid x_k, x_0$

$$k \geq 2, \quad x_{k\text{-}1} \mid x_k, x_0 \sim \frac{q_k(x_k \mid x_{k\text{-}1}) q_{k\text{-}1|0}(x_{k\text{-}1} \mid x_0)}{q_{k|0}(x_k \mid x_0)} \mathrm{d}x_{k\text{-}1}$$
$$= \mathcal{N}\left(\gamma_k x_0 + \lambda_k x_k, \tilde{\beta}_k I_d\right)(\mathrm{d}x_{k\text{-}1})$$

ref: Ho et al., 2020[4]

Motivation for the backward process: informal notation

$$\mathcal{N}(0, I_d) \approx Q_n \circ \cdots \circ Q_1 \mu$$
$$Q_1^{-1} \circ \cdots \circ Q_n^{-1} \mathcal{N}(0, I_d) \approx \mu$$

Distributions of $x_{k\text{-}1} \mid x_k, x_0$

$$
\begin{aligned}
k \geq 2, \quad x_{k\text{-}1} \mid x_k, x_0 &\sim \frac{q_k(x_k \mid x_{k\text{-}1}) q_{k\text{-}1|0}(x_{k\text{-}1} \mid x_0)}{q_{k|0}(x_k \mid x_0)} \mathrm{d}x_{k\text{-}1} \\
&= \mathcal{N}\left(\gamma_k x_0 + \lambda_k x_k, \tilde{\beta}_k I_d\right)(\mathrm{d}x_{k\text{-}1}) \\
&=: q_{k\text{-}1|k,0}(x_{k\text{-}1} \mid x_k, x_0) \mathrm{d}x_{k\text{-}1}
\end{aligned}
$$

with $\gamma_k = \frac{\beta_k \sqrt{\alpha_{k\text{-}1}}}{1 - \alpha_k}$, $\lambda_k = \frac{1 - \alpha_{k\text{-}1}}{1 - \alpha_k} \sqrt{1 - \beta_k}$ and $\tilde{\beta}_k = \frac{1 - \alpha_{k\text{-}1}}{1 - \alpha_k} \beta_k$.

---

ref: Ho et al., 2020[4]

Motivation for the backward process: informal notation

$$\mathcal{N}(0, I_d) \approx Q_n \circ \cdots \circ Q_1 \mu$$
$$Q_1^{-1} \circ \cdots \circ Q_n^{-1} \mathcal{N}(0, I_d) \approx \mu$$

Distributions of $x_{k\text{-}1} \mid x_k, x_0$

$$k \geq 2, \quad x_{k\text{-}1} \mid x_k, x_0 \sim \frac{q_k(x_k \mid x_{k\text{-}1}) q_{k\text{-}1|0}(x_{k\text{-}1} \mid x_0)}{q_{k|0}(x_k \mid x_0)} \mathrm{d}x_{k\text{-}1}$$
$$= \mathcal{N}\left(\gamma_k x_0 + \lambda_k x_k, \tilde{\beta}_k I_d\right)(\mathrm{d}x_{k\text{-}1})$$
$$=: q_{k\text{-}1|k,0}(x_{k\text{-}1} \mid x_k, x_0) \mathrm{d}x_{k\text{-}1}$$

with $\gamma_k = \frac{\beta_k \sqrt{\alpha_{k\text{-}1}}}{1 - \alpha_k}$, $\lambda_k = \frac{1 - \alpha_{k\text{-}1}}{1 - \alpha_k} \sqrt{1 - \beta_k}$ and $\tilde{\beta}_k = \frac{1 - \alpha_{k\text{-}1}}{1 - \alpha_k} \beta_k$.

If for some k, $\beta_k = 1$ then $\alpha_k = 0$ and

$$q_{k\text{-}1|k,0}(x_{k\text{-}1} \mid x_k, x_0) = q_{k\text{-}1|0}(x_{k\text{-}1} \mid x_0).$$

---

ref: Ho et al., 2020[4]

Learning the backward process

Expression of the backward process:

$$x_{k\text{-}1} \mid x_k \sim \int_{\mathbb{R}^d} \mathcal{N}\left(\gamma_k x_0 + \lambda_k x_k, \tilde{\beta}_k I_d\right)(\mathrm{d}x_{k\text{-}1})\mu(\mathrm{d}x_0) =: q_{k\text{-}1|k}(x_{k\text{-}1} \mid x_k)\mathrm{d}x_{k\text{-}1}.$$

ref: Ho et al., 2020[4]

Learning the backward process

Expression of the backward process:

$$x_{k\text{-}1} \mid x_k \sim \int_{\mathbb{R}^d} \mathcal{N}\left(\gamma_k x_0 + \lambda_k x_k, \tilde{\beta}_k I_d\right)(\mathrm{d}x_{k\text{-}1})\mu(\mathrm{d}x_0) =: q_{k\text{-}1|k}(x_{k\text{-}1} \mid x_k)\mathrm{d}x_{k\text{-}1}.$$

Denoising diffusion probabilistic model

$\theta \in \mathbb{R}^p$ the parameters of the model.

$$
\begin{aligned}
&x_n \sim p_n(x_n)\mathrm{d}x_n := \mathcal{N}(0, I_d)(\mathrm{d}x_n) \\
&x_{n\text{-}1} \mid x_n \sim p_{n\text{-}1|n}(x_{n\text{-}1} \mid x_n; \theta)\mathrm{d}x_{n\text{-}1} := \mathcal{N}\left(\mu_n(x_n, \theta), \Sigma_n(x_n, \theta)\right)(\mathrm{d}x_{n\text{-}1}) \\
&\vdots \\
&x_0 \mid x_1 \sim p_{0|1}(x_0 \mid x_1; \theta)\mathrm{d}x_0 := \mathcal{N}(\mu_1(x_1, \theta), \Sigma_1(x_1, \theta))(\mathrm{d}x_0).
\end{aligned}
$$

ref: Ho et al., 2020[4]

Learning the backward process

Expression of the backward process:

$$x_{k\text{-}1} \mid x_k \sim \int_{\mathbb{R}^d} \mathcal{N}\left(\gamma_k x_0 + \lambda_k x_k, \tilde{\beta}_k I_d\right)(\mathrm{d}x_{k\text{-}1})\mu(\mathrm{d}x_0) =: q_{k\text{-}1|k}(x_{k\text{-}1} \mid x_k)\mathrm{d}x_{k\text{-}1}.$$

Denoising diffusion probabilistic model

$\theta \in \mathbb{R}^p$ the parameters of the model.

$$x_n \sim p_n(x_n)\mathrm{d}x_n := \mathcal{N}(0, I_d)(\mathrm{d}x_n)$$

$$x_{n\text{-}1} \mid x_n \sim p_{n\text{-}1|n}(x_{n\text{-}1} \mid x_n; \theta)\mathrm{d}x_{n\text{-}1} := \mathcal{N}\left(\mu_n(x_n, \theta), \Sigma_n(x_n, \theta)\right)(\mathrm{d}x_{n\text{-}1})$$

$$\vdots$$

$$x_0 \mid x_1 \sim p_{0|1}(x_0 \mid x_1; \theta)\mathrm{d}x_0 := \mathcal{N}(\mu_1(x_1, \theta), \Sigma_1(x_1, \theta))(\mathrm{d}x_0).$$

Marginal distribution of the data

$$p_0(x_0; \theta) := \int_{\left(\mathbb{R}^d\right)^n} p_n(x_n) \prod_{k=1}^{n} p_{k\text{-}1|k}(x_{k\text{-}1} \mid x_k; \theta)\mathrm{d}x_{1:n}.$$

ref: Ho et al., 2020[4]

Expected lower-bound

Let $\left(x_0^{(1)}, \cdots, x_0^{(N)}\right)$ be our dataset, represented by the empirical measure $\hat{\mu}_N = \sum_{i=1}^{N} \delta_{x_0^{(i)}}$.

ref: Ho et al., 2020[4]

Expected lower-bound

Let $\left(x_0^{(1)}, \cdots, x_0^{(N)}\right)$ be our dataset, represented by the empirical measure $\hat{\mu}_N = \sum_{i=1}^{N} \delta_{x_0^{(i)}}$.

Maximum likelihood estimator

$$\hat{\theta}(\hat{\mu}_N) \in \underset{\theta \in \mathbb{R}^p}{\operatorname{Argmax}} \left\{ \ell\left(\theta; \hat{\mu}_N\right) := \frac{1}{N} \int_{\mathbb{R}^d} \log p_0(x_0; \theta) \hat{\mu}_N(\mathrm{d}x_0) \right\}.$$

ref: Ho et al., 2020[4]

Let $\left(x_0^{(1)}, \cdots, x_0^{(N)}\right)$ be our dataset, represented by the empirical measure $\hat{\mu}_N = \sum_{i=1}^N \delta_{x_0^{(i)}}$.

Maximum likelihood estimator

$$\hat{\theta}(\hat{\mu}_N) \in \underset{\theta \in \mathbb{R}^p}{\text{Argmax}} \left\{ \ell\left(\theta; \hat{\mu}_N\right) := \frac{1}{N} \int_{\mathbb{R}^d} \log p_0(x_0; \theta) \hat{\mu}_N(\mathrm{d}x_0) \right\}.$$

For any $x_0 \in \mathbb{R}^d$,

$$\log p_0(x_0; \theta) = \log \left(\int_{\left(\mathbb{R}^d\right)^n} p_n(x_n) \prod_{k=1}^n p_{k\text{-}1|k}(x_{k\text{-}1} \mid x_k; \theta) \mathrm{d}x_{1:n} \right)$$

ref: Ho et al., 2020[4]

Let $\left(x_0^{(1)}, \cdots, x_0^{(N)}\right)$ be our dataset, represented by the empirical measure $\hat{\mu}_N = \sum_{i=1}^N \delta_{x_0^{(i)}}$.

Maximum likelihood estimator

$$\hat{\theta}(\hat{\mu}_N) \in \underset{\theta \in \mathbb{R}^p}{\operatorname{Argmax}} \left\{ \ell\left(\theta; \hat{\mu}_N\right) := \frac{1}{N} \int_{\mathbb{R}^d} \log p_0(x_0; \theta) \hat{\mu}_N(\mathrm{d}x_0) \right\}.$$

For any $x_0 \in \mathbb{R}^d$,

$$\begin{aligned}
\log p_0(x_0; \theta) &= \log \left(\int_{\left(\mathbb{R}^d\right)^n} p_n(x_n) \prod_{k=1}^{n} p_{k\text{-}1|k}(x_{k\text{-}1} \mid x_k; \theta) \mathrm{d}x_{1:n} \right) \\
&= \log \left(\int_{\left(\mathbb{R}^d\right)^n} p_{0|1}(x_0 \mid x_1; \theta) \frac{p_n(x_n)}{q_{n|0}(x_n \mid x_0)} q_{n|0}(x_n \mid x_0) \right. \\
&\quad \left. \times \prod_{k=2}^{n} \frac{p_{k\text{-}1|k}(x_{k\text{-}1} \mid x_k; \theta)}{q_{k\text{-}1|k,0}(x_{k\text{-}1} \mid x_k, x_0)} q_{k\text{-}1|k,0}(x_{k\text{-}1} \mid x_k, x_0) \mathrm{d}x_{1:n} \right)
\end{aligned}$$

ref: Ho et al., 2020[4]

Expected lower-bound

$$\log p_0(x_0;\theta) = \ell(\theta, \delta_{x_0})$$

$$\geq \int_{(\mathbb{R}^d)^n} \log \left(p_{0|1}(x_0 \mid x_1;\theta) \frac{p_n(x_n)}{q_{n|0}(x_n \mid x_0)} \prod_{k=2}^{n} \frac{p_{k\text{-}1|k}(x_{k\text{-}1} \mid x_k;\theta)}{q_{k\text{-}1|k,0}(x_{k\text{-}1} \mid x_k, x_0)} \right)$$

$$\times\, q_{n|0}(x_n \mid x_0) \prod_{k=2}^{n} q_{k\text{-}1|k,0}(x_{k\text{-}1} \mid x_k, x_0) \mathrm{d}x_{1:n}$$

$$=: \tilde{\ell}(\theta, \delta_{x_0})$$ (**ELBO**).

ref: Ho et al., 2020[4]

Expected lower-bound

$$\tilde{\ell}(\theta, \delta_{x_0}) = \int_{\mathbb{R}^d} \log\left(p_{0|1}(x_0 \mid x_1; \theta)\right) q_{1|0}(x_1 \mid x_0) \mathrm{d}x_1 \left(=: \ell_{0|1}(\theta, \delta_{x_0})\right)$$

ref: Ho et al., 2020[4]

Expected lower-bound

$$\tilde{\ell}(\theta, \delta_{x_0}) = \int_{\mathbb{R}^d} \log\left(p_{0|1}(x_0 \mid x_1; \theta)\right) q_{1|0}(x_1 \mid x_0)\mathrm{d}x_1 \, (=: \ell_{0|1}(\theta, \delta_{x_0}))$$

$$+ \int_{\mathbb{R}^d} \log\left(\frac{p_n(x_n)}{q_{n|0}(x_n \mid x_0)}\right) q_{n|0}(x_n \mid x_0)\mathrm{d}x_n \, (= \mathrm{cst})$$

ref: Ho et al., 2020[4]

Expected lower-bound

$$
\begin{aligned}
\tilde{\ell}(\theta, \delta_{x_0}) &= \int_{\mathbb{R}^d} \log\left(p_{0|1}(x_0 \mid x_1; \theta)\right) q_{1|0}(x_1 \mid x_0)\mathrm{d}x_1 \,(=: \ell_{0|1}(\theta, \delta_{x_0}))\\
&+ \int_{\mathbb{R}^d} \log\left(\frac{p_n(x_n)}{q_{n|0}(x_n \mid x_0)}\right) q_{n|0}(x_n \mid x_0)\mathrm{d}x_n \,(= \mathrm{cst})\\
&+ \sum_{k=2}^{n} \int_{\mathbb{R}^d}\int_{\mathbb{R}^d} \log\left(\frac{p_{k\text{-}1|k}(x_{k\text{-}1} \mid x_k; \theta)}{q_{k\text{-}1|k,0}(x_{k\text{-}1} \mid x_k, x_0)}\right) q_{k\text{-}1|k,0}(x_{k\text{-}1} \mid x_k, x_0)\mathrm{d}x_{k\text{-}1}\\
&\times q_{k|0}(x_k \mid x_0)\mathrm{d}x_k\\
&\Bigg(= -\sum_{k=2}^{n} \int_{\mathbb{R}^d} \mathcal{D}_{KL}\left(\mathcal{N}\left(\gamma_k x_0 + \lambda_k x_k, \tilde{\beta}_k I_d\right) \| \mathcal{N}\left(\mu_k(x_k, \theta), \Sigma_k(x_k, \theta)\right)\right)\\
&\qquad \times q_{k|0}(x_k \mid x_0)\mathrm{d}x_k =: -\sum_{k=2}^{n} \mathcal{D}_k(\theta, \delta_{x_0})\Bigg).
\end{aligned}
$$

---

ref: Ho et al., 2020[4]

Score function

If $\Sigma_k(x_k, \theta) = \sigma_k^2 I_d$, then

$$\mathcal{D}_k(\theta, \delta_{x_0}) = \text{cst} + \frac{1}{2\sigma_k^2} \int_{\mathbb{R}^d} \|\gamma_k x_0 + \lambda_k x_k - \mu_k(x_k, \theta)\|^2 q_{k|0}(x_k \mid x_0) \mathrm{d}x_k .$$

ref: Ho et al., 2020[4]

If $\Sigma_k(x_k,\theta) = \sigma_k^2 I_d$, then

$$\mathcal{D}_k(\theta, \delta_{x_0}) = \text{cst} + \frac{1}{2\sigma_k^2}\int_{\mathbb{R}^d} \|\gamma_k x_0 + \lambda_k x_k - \mu_k(x_k,\theta)\|^2 q_{k|0}(x_k \mid x_0)\mathrm{d}x_k .$$

ε-functions

With the following reparameterization:

$$\mu_k(x_k,\theta) = \frac{1}{\sqrt{1-\beta_k}}\left(x_k - \frac{\beta_k}{\sqrt{1-\alpha_k}}\varepsilon_k(x_k,\theta)\right),$$

we obtain

$$\mathcal{D}_k(\theta, \delta_{x_0}) = \text{cst} + \nu_k \int_{\mathbb{R}^d} \left\|\varepsilon - \varepsilon_k(\sqrt{\alpha_k}x_0 + \sqrt{1-\alpha_k}\varepsilon, \theta)\right\|^2 \mathcal{N}(0, I_d)(\mathrm{d}\varepsilon),$$

with $\nu_k = \dfrac{\beta_k^2}{2\sigma_k^2(1-\beta_k)(1-\alpha_k)}$.

ref: Ho et al., 2020[4]

Score function

If $\Sigma_k(x_k, \theta) = \sigma_k^2 I_d$, then

$$\mathcal{D}_k(\theta, \delta_{x_0}) = \text{cst} + \frac{1}{2\sigma_k^2} \int_{\mathbb{R}^d} \|\gamma_k x_0 + \lambda_k x_k - \mu_k(x_k, \theta)\|^2 q_{k|0}(x_k \mid x_0) \mathrm{d}x_k .$$

ε-functions

With the following reparameterization:

$$\mu_k(x_k, \theta) = \frac{1}{\sqrt{1-\beta_k}} \left(x_k - \frac{\beta_k}{\sqrt{1-\alpha_k}} \varepsilon_k(x_k, \theta) \right),$$

we obtain

$$\mathcal{D}_k(\theta, \delta_{x_0}) = \text{cst} + \nu_k \int_{\mathbb{R}^d} \left\| \varepsilon - \varepsilon_k(\sqrt{\alpha_k} x_0 + \sqrt{1-\alpha_k} \varepsilon, \theta) \right\|^2 \mathcal{N}(0, I_d)(\mathrm{d}\varepsilon),$$

with $\nu_k = \dfrac{\beta_k^2}{2\sigma_k^2(1-\beta_k)(1-\alpha_k)}$.

$$\varepsilon_k(x_k, \theta) \approx -\sqrt{1-\alpha_k} \nabla_{x_k} \log q_k(x_k).$$

ref: Ho et al., 2020[4]

Maximum ELBO estimator

$$\hat{\theta}(\hat{\mu}_N) \in \underset{\theta \in \mathbb{R}^p}{\mathrm{Argmax}} \left\{ \tilde{\ell}(\theta, \hat{\mu}_N) = \ell_{0|1}(\theta, \hat{\mu}_N) - \sum_{k=2}^{n} \mathcal{D}_k(\theta, \hat{\mu}_N) \right\}.$$

ref: Ho et al., 2020[4]

Maximum ELBO estimator

$$\hat{\theta}(\hat{\mu}_N) \in \underset{\theta \in \mathbb{R}^p}{\mathrm{Argmax}} \left\{ \tilde{\ell}(\theta, \hat{\mu}_N) = \ell_{0|1}(\theta, \hat{\mu}_N) - \sum_{k=2}^{n} \mathcal{D}_k(\theta, \hat{\mu}_N) \right\}.$$

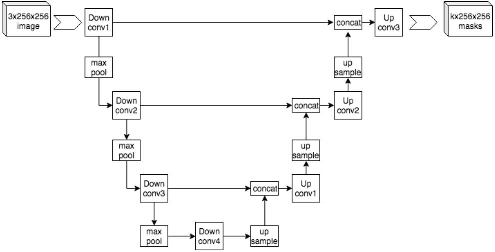


Figure 1: U-Net architecture for segmentation (Mehrdad Yazdani, Wikipedia), from which the architecture for the score functions $\varepsilon_k(x, \theta)$ is inspired.

ref: Ho et al., 2020[4]

Quantification of the performance of generative models

Let $\hat{\nu}_M = \sum_{m=1}^{M} \delta_{x_m}$ a measure representing a validation dataset. Our trained model is a probability distribution $\tilde{\mu}$. The performance of the model is quantified by $\text{dist}\left(\tilde{\mu}, \frac{1}{M}\hat{\nu}_M\right)$ subject to the condition that $\text{dist}\left(\frac{1}{M}\hat{\nu}_M, \mu\right) \approx 0$.
In practice, computing $\text{dist}(\mu_1, \mu_2)$ is intractable in most applications.

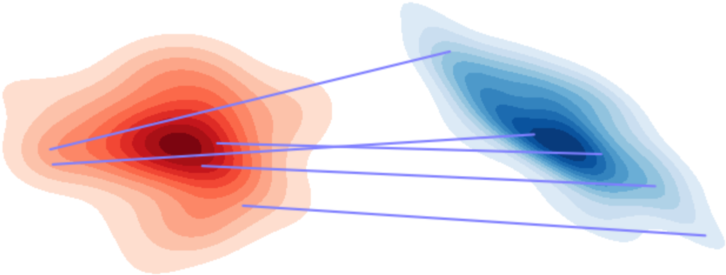


Figure 2: Kilian Fatras, Towards Data Science, 2020.

Inception-V3 classifier

Let $\mathcal{Y} = \{c_1, \ldots, c_K\}$ a set of image classes ($K \approx 20,000$ for ImageNet).

Let $x \in \mathbb{R}^d \mapsto \mu^\dagger(\mathrm{d}y \mid x) := \sum_{k=1}^{K} p_k^\dagger(x)\delta_{c_k}(\mathrm{d}y) \in \mathcal{P}(\mathcal{Y})$ be the Inception V3 classifier (chosen as reference).

Ref: Szegedy et al., 2015[9].

Inception-V3 classifier

Let $\mathcal{Y} = \{c_1, \ldots, c_K\}$ a set of image classes ($K \approx 20,000$ for ImageNet).

Let $x \in \mathbb{R}^d \mapsto \mu^\dagger(\mathrm{d}y \mid x) := \sum_{k=1}^{K} p_k^\dagger(x)\delta_{c_k}(\mathrm{d}y) \in \mathcal{P}(\mathcal{Y})$ be the Inception V3 classifier (chosen as reference).

Figure 3: Classification using VGG-net, older than Inception-V3.

Ref: Szegedy et al., 2015[9].

Inception score

Let $\tilde{\mu}(\mathrm{d}x) = p_0\left(x; \hat{\theta}\right) \mathrm{d}x \in \mathcal{P}\left(\mathbb{R}^d\right)$ be the learned generative model.

Inception score

$$\mathcal{S}_i\left(\tilde{\mu}; \mu^\dagger\right) := \exp\left[\int_{\mathbb{R}^d} \mathcal{D}_{KL}\left(\mu^\dagger(\mathrm{d}y \mid x) \| \mu^\dagger(\mathrm{d}y \mid \tilde{\mu})\right) \tilde{\mu}(\mathrm{d}x)\right],$$

where $\mu^\dagger(\mathrm{d}y \mid \tilde{\mu}) := \int_{\mathbb{R}^d} \mu^\dagger(\mathrm{d}y \mid x)\tilde{\mu}(\mathrm{d}x)$.

Ref: Salimans et al. (2016) [7].

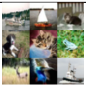
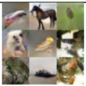
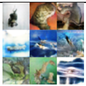
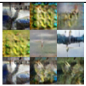
Inception score

Let $\tilde{\mu}(\mathrm{d}x) = p_0\left(x;\hat{\theta}\right)\mathrm{d}x \in \mathcal{P}\left(\mathbb{R}^d\right)$ be the learned generative model.

Inception score

$$\mathcal{S}_i\left(\tilde{\mu};\mu^\dagger\right) := \exp\left[\int_{\mathbb{R}^d} \mathcal{D}_{KL}\left(\mu^\dagger(\mathrm{d}y \mid x) \| \mu^\dagger(\mathrm{d}y \mid \tilde{\mu})\right)\tilde{\mu}(\mathrm{d}x)\right],$$

where $\mu^\dagger(\mathrm{d}y \mid \tilde{\mu}) := \int_{\mathbb{R}^d} \mu^\dagger(\mathrm{d}y \mid x)\tilde{\mu}(\mathrm{d}x)$.

Samples						
Model	Real data	Model 1	Model 2	Model 3	Model 4	Model 5
Score ± std.	11.24 ± .12	8.09 ± .07	7.54 ± .07	6.86 ± .06	6.83 ± .06	4.36 ± .04

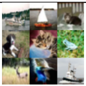
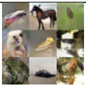
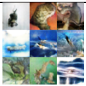
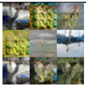
Ref: Salimans et al. (2016) [7].

Let $\tilde{\mu}(\mathrm{d}x) = p_0\left(x;\hat{\theta}\right)\mathrm{d}x \in \mathcal{P}\left(\mathbb{R}^d\right)$ be the learned generative model.

Inception score

$$\mathcal{S}_i\left(\tilde{\mu};\mu^\dagger\right) := \exp\left[\int_{\mathbb{R}^d} \mathcal{D}_{KL}\left(\mu^\dagger(\mathrm{d}y \mid x) \| \mu^\dagger(\mathrm{d}y \mid \tilde{\mu})\right)\tilde{\mu}(\mathrm{d}x)\right],$$

where $\mu^\dagger(\mathrm{d}y \mid \tilde{\mu}) := \int_{\mathbb{R}^d} \mu^\dagger(\mathrm{d}y \mid x)\tilde{\mu}(\mathrm{d}x)$.

Samples						
Model	Real data	Model 1	Model 2	Model 3	Model 4	Model 5
Score ± std.	11.24 ± .12	8.09 ± .07	7.54 ± .07	6.86 ± .06	6.83 ± .06	4.36 ± .04

Trade-off between diversity and fidelity of the generated samples.

Ref: Salimans et al. (2016) [7].

Fréchet Inception *distance*

Let $\varphi : \mathbb{R}^d \to \mathbb{R}^K$ be the final embedding layer of the Inception V3 classifier, such that for all $k \in [\![1, K]\!]$,

$$p_k^\dagger(x) = \frac{\exp\left(\varphi(x)_k\right)}{\sum_{\ell=1}^K \exp\left(\varphi(x)_\ell\right)}.$$

Ref: Szegedy et al. (2015)[9], Heusel et al. (2017)[3]

Fréchet Inception *distance*

Let $\varphi : \mathbb{R}^d \to \mathbb{R}^K$ be the final embedding layer of the Inception V3 classifier, such that for all $k \in [\![1, K]\!]$,

$$p_k^\dagger(x) = \frac{\exp\left(\varphi(x)_k\right)}{\sum_{\ell=1}^{K} \exp\left(\varphi(x)_\ell\right)}.$$

Let $\tilde{\mu}_M = \sum_{m=1}^{M} \delta_{\tilde{x}_m}$ be independent samples from the model.

Ref: Szegedy et al. (2015)[9], Heusel et al. (2017)[3]

Fréchet Inception *distance*

Let $\varphi : \mathbb{R}^d \to \mathbb{R}^K$ be the final embedding layer of the Inception V3 classifier, such that for all $k \in [\![1, K]\!]$,

$$p_k^\dagger(x) = \frac{\exp\left(\varphi(x)_k\right)}{\sum_{\ell=1}^K \exp\left(\varphi(x)_\ell\right)}.$$

Let $\tilde{\mu}_M = \sum_{m=1}^M \delta_{\tilde{x}_m}$ be independent samples from the model.

Frechet Inception *distance*

$$\mathcal{D}_f\left(\frac{1}{M}\tilde{\mu}_M, \frac{1}{M}\nu_M\right)^2 := \left\|\text{mean}\left(\varphi_\# \frac{1}{M}\tilde{\mu}_M\right) - \text{mean}\left(\varphi_\# \frac{1}{M}\nu_M\right)\right\|^2$$
$$+ \operatorname{Tr}\left(\text{cov}\left(\varphi_\# \frac{1}{M}\tilde{\mu}_M\right) + \text{cov}\left(\varphi_\# \frac{1}{M}\nu_M\right)\right.$$
$$\left. -2\left(\text{cov}\left(\varphi_\# \frac{1}{M}\tilde{\mu}_M\right)\text{cov}\left(\varphi_\# \frac{1}{M}\nu_M\right)\right)^{1/2}\right).$$

Ref: Szegedy et al. (2015)[9], Heusel et al. (2017)[3]

Figure 4: **FID = 33.0** for Model 1, generating images of Welsh Corgis, trained on ImageNet.

Ref: Dhariwal and Nichol (2021) [2].

Figure 5: **FID = 12.0** for Model 2, generating images of Welsh Corgis, trained on ImageNet.

Ref: Dhariwal and Nichol (2021) [2].

Fréchet Inception *distance*

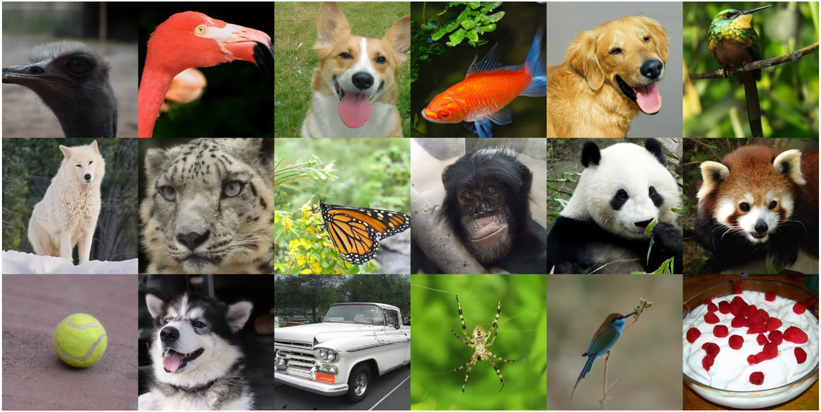


Figure 6: **FID = 3.85** for Model 3 trained on the whole ImageNet dataset.

Ref: Dhariwal and Nichol (2021) [2].

Outline

Unconditional denoising diffusion probabilistic models

Classifier guidance

Classifier-free guidance

Latent diffusion models

Universal guidance

Conditional denoising

Finite-class mixture distribution

Let $\mathcal{Y} = \{c_1, \ldots, c_K\}$ be a finite set.
We assume that the target distribution can be decomposed into a finite mixture:

$$\mu(\mathrm{d}x) = \sum_{c \in \mathcal{Y}} \pi_c \mu_c(\mathrm{d}x \mid c).$$

Ref: Dhariwal and Nichol, 2021[2].

Conditional denoising

Finite-class mixture distribution

Let $\mathcal{Y} = \{c_1, \ldots, c_K\}$ be a finite set.
We assume that the target distribution can be decomposed into a finite mixture:

$$\mu(\mathrm{d}x) = \sum_{c \in \mathcal{Y}} \pi_c \mu_c(\mathrm{d}x \mid c).$$

We assume that we have a diffusion model $(p_{k\text{-}1|k})_{1 \leq k \leq n}$, and a classifier model $(p_{y|\ell})_{0 \leq \ell \leq n}$ trained on noisy data.

Ref: Dhariwal and Nichol, 2021[2].

Finite-class mixture distribution

Let $\mathcal{Y} = \{c_1, \ldots, c_K\}$ be a finite set.
We assume that the target distribution can be decomposed into a finite mixture:

$$\mu(\mathrm{d}x) = \sum_{c \in \mathcal{Y}} \pi_c \mu_c(\mathrm{d}x \mid c).$$

We assume that we have a diffusion model $(p_{k\text{-}1|k})_{1 \le k \le n}$, and a classifier model $(p_{y|\ell})_{0 \le \ell \le n}$ trained on noisy data.

Condtional denoising diffusion model

Let $c \in \mathcal{Y}$, and $s > 0$,

$$x_n \mid c \sim \tilde{p}_n(x_n \mid c) \propto p_{y|n}(c \mid x_n)^s \mathcal{N}(0, I_d)(\mathrm{d}x_n),$$

and for $k = n, \ldots, 1$,

$$x_{k\text{-}1} \mid x_k, c \sim \tilde{p}_{k\text{-}1|k}(x_{k\text{-}1} \mid x_k, c) \propto p_{y|k\text{-}1}(c \mid x_{k\text{-}1})^s p_{k\text{-}1|k}(x_{k\text{-}1} \mid x_k) \mathrm{d}x_{k\text{-}1}.$$

In general, distributions $\tilde{p}_{k\text{-}1|k}$ cannot be sampled.

Ref: Dhariwal and Nichol, 2021[2].

Perturbed Gaussian transition

Denoising kernel for conditional sampling

With $p_{k\text{-}1|k}(x_{k\text{-}1}|x_k)\mathrm{d}x_{k\text{-}1} = \mathcal{N}\left(\mu_k(x_k), \Sigma_k(x_k)\right)(\mathrm{d}x_{k\text{-}1})$,

$$\tilde{p}_{k\text{-}1|k}(x_{k\text{-}1} \mid x_k, c) \approx \mathcal{N}\left(\mu_k(x_k) + s\Sigma_k(x_k)\nabla_{x_{k\text{-}1}} \log p_{y|k\text{-}1}(c \mid \mu_k(x_k)), \Sigma_k(x_k)\right)(\mathrm{d}x_{k\text{-}1}).$$

Indeed, we have

$$\tilde{p}_{k\text{-}1|k}(x_{k\text{-}1} \mid x_k, c) \propto \exp\left(-\frac{1}{2}\left(x_{k\text{-}1} - \mu_k(x_k)\right)^{\mathsf{T}} \Sigma_k(x_k)^{-1}\left(x_{k\text{-}1} - \mu_k(x_k)\right) + s\log p_{y|k\text{-}1}(c \mid x_{k\text{-}1})\right),$$

Ref: Dhariwal and Nichol, 2021[2].

Perturbed Gaussian transition

Denoising kernel for conditional sampling

With $p_{k\text{-}1|k}(x_{k\text{-}1}|x_k)\mathrm{d}x_{k\text{-}1} = \mathcal{N}\left(\mu_k(x_k), \Sigma_k(x_k)\right)(\mathrm{d}x_{k\text{-}1})$,

$$\tilde{p}_{k\text{-}1|k}(x_{k\text{-}1} \mid x_k, c)$$
$$\approx \mathcal{N}\left(\mu_k(x_k) + s\Sigma_k(x_k)\nabla_{x_{k\text{-}1}} \log p_{y|k\text{-}1}(c \mid \mu_k(x_k)), \Sigma_k(x_k)\right)(\mathrm{d}x_{k\text{-}1}).$$

Indeed, we have

$$\tilde{p}_{k\text{-}1|k}(x_{k\text{-}1} \mid x_k, c) \propto \exp\left(-\frac{1}{2}\left(x_{k\text{-}1} - \mu_k(x_k)\right)^\intercal \Sigma_k(x_k)^{-1}\left(x_{k\text{-}1} - \mu_k(x_k)\right)\right.$$
$$\left. + s \log p_{y|k\text{-}1}(c \mid x_{k\text{-}1})\right),$$

$$\log p_{y|k\text{-}1}(c \mid x_{k\text{-}1}) = \log p_{y|k\text{-}1}(c \mid \mu_k(x_k))$$
$$+ \int_0^1 \nabla_{x_{k\text{-}1}} \log p_{y|k\text{-}1}(c \mid \mu_k(x_k) + \alpha(x_{k\text{-}1} - \mu_k(x_k)))^\intercal (x_{k\text{-}1} - \mu_k(x_k))\mathrm{d}\alpha$$

Ref: Dhariwal and Nichol, 2021[2].

Denoising kernel for conditional sampling

With $p_{k\text{-}1|k}(x_{k\text{-}1}|x_k)\mathrm{d}x_{k\text{-}1} = \mathcal{N}(\mu_k(x_k), \Sigma_k(x_k))(\mathrm{d}x_{k\text{-}1})$,

$$\tilde{p}_{k\text{-}1|k}(x_{k\text{-}1} \mid x_k, c) \approx \mathcal{N}\left(\mu_k(x_k) + s\Sigma_k(x_k)\nabla_{x_{k\text{-}1}} \log p_{y|k\text{-}1}(c \mid \mu_k(x_k)), \Sigma_k(x_k)\right)(\mathrm{d}x_{k\text{-}1}).$$

Indeed, we have

$$\tilde{p}_{k\text{-}1|k}(x_{k\text{-}1} \mid x_k, c) \propto \exp\left(-\frac{1}{2}(x_{k\text{-}1} - \mu_k(x_k))^\intercal \Sigma_k(x_k)^{-1}(x_{k\text{-}1} - \mu_k(x_k)) + s \log p_{y|k\text{-}1}(c \mid x_{k\text{-}1})\right),$$

$$\log p_{y|k\text{-}1}(c \mid x_{k\text{-}1}) = \log p_{y|k\text{-}1}(c \mid \mu_k(x_k)) + \int_0^1 \nabla_{x_{k\text{-}1}} \log p_{y|k\text{-}1}(c \mid \mu_k(x_k) + \alpha(x_{k\text{-}1} - \mu_k(x_k)))^\intercal (x_{k\text{-}1} - \mu_k(x_k))\mathrm{d}\alpha$$
$$=: \log p_{y|k\text{-}1}(c \mid \mu_k(x_k)) + g_{k\text{-}1}(x_{k\text{-}1})^\intercal (x_{k\text{-}1} - \mu_k(x_k)),$$

Ref: Dhariwal and Nichol, 2021[2].

Perturbed Gaussian transition

$$\tilde{p}_{k\text{-}1|k}(x_{k\text{-}1} \mid x_k, c) \propto \exp\Bigg(-\frac{1}{2}\left(x_{k\text{-}1} - \mu_k(x_k)\right)^{\mathsf{T}} \Sigma_k(x_k)^{-1} \left(x_{k\text{-}1} - \mu_k(x_k)\right)$$
$$+ s g_k(x_{k\text{-}1})^{\mathsf{T}}(x_{k\text{-}1} - \mu_k(x_k)) \Bigg).$$

Ref: Dhariwal and Nichol, 2021[2].

$$\tilde{p}_{k\text{-}1|k}(x_{k\text{-}1} \mid x_k, c) \propto \exp\Bigg(-\frac{1}{2} \left(x_{k\text{-}1} - \mu_k(x_k)\right)^\mathsf{T} \Sigma_k(x_k)^{-1} \left(x_{k\text{-}1} - \mu_k(x_k)\right) + s g_k(x_{k\text{-}1})^\mathsf{T}(x_{k\text{-}1} - \mu_k(x_k)) \Bigg).$$

Class-conditional mean and score for the sampling

Conditional backward mean:

$$\tilde{\mu}_k(x_k, c) := \mu_k(x_k) + s\Sigma_k(x_k)\nabla_{x_{k\text{-}1}} \log p_{y|k\text{-}1}(c \mid \mu_k(x_k)).$$

Conditional score:

$$\tilde{\varepsilon}_k(x_k, c) := \varepsilon_k(x_k) - s\frac{\sigma_k^2}{\beta_k}\sqrt{(1-\alpha_k)(1-\beta_k)}\nabla_{x_{k\text{-}1}} \log p_{y|k\text{-}1}(c \mid \mu_k(x_k)).$$

Ref: Dhariwal and Nichol, 2021[2].

Sensitivity with respect to s

Figure 7: s ranging from ≈ 0 to 5.5.

Ref: Dhariwal and Nichol, 2021[2].

Outline

Unconditional denoising diffusion probabilistic models

Classifier guidance

Classifier-free guidance

Latent diffusion models

Universal guidance

Empirical distribution of the data: $\hat{\mu}_N = \sum_{i=1}^{N} \delta_{\left(x_0^{(i)}, y^{(i)}\right)} \in \mathcal{M}_+(\mathcal{X} \times \mathcal{Y})$.

Ref: Ho and Salimans (2022) [5].

Classifier-free guidance

Empirical distribution of the data: $\hat{\mu}_N = \sum_{i=1}^{N} \delta_{\left(x_0^{(i)}, y^{(i)}\right)} \in \mathcal{M}_+(\mathcal{X} \times \mathcal{Y})$.

Parameterization of the score functions: $\varepsilon_k : \mathcal{X} \times (\mathcal{Y} \cup \{\varnothing\}) \times \Theta \to \mathbb{R}^d$.

Ref: Ho and Salimans (2022) [5].

Classifier-free guidance

Empirical distribution of the data: $\hat{\mu}_N = \sum_{i=1}^{N} \delta_{\left(x_0^{(i)}, y^{(i)}\right)} \in \mathcal{M}_+(\mathcal{X} \times \mathcal{Y})$.

Parameterization of the score functions: $\varepsilon_k : \mathcal{X} \times (\mathcal{Y} \cup \{\varnothing\}) \times \Theta \to \mathbb{R}^d$.
Projector on the empty class:

$$\pi_\varnothing : \sum_{k=1}^{m} a_k \delta_{(x_k, y_k)} \in \mathrm{Span}\left\{\delta_{(x,y)},\ (x,y) \in \mathcal{X} \times \mathcal{Y}\right\} \mapsto \sum_{k=1}^{m} a_k \delta_{(x_k, \varnothing)}.$$

Ref: Ho and Salimans (2022) [5].

Empirical distribution of the data: $\hat{\mu}_N = \sum_{i=1}^N \delta_{\left(x_0^{(i)}, y^{(i)}\right)} \in \mathcal{M}_+(\mathcal{X} \times \mathcal{Y})$.

Parameterization of the score functions: $\varepsilon_k : \mathcal{X} \times (\mathcal{Y} \cup \{\varnothing\}) \times \Theta \to \mathbb{R}^d$.

Projector on the empty class:

$$\pi_\varnothing : \sum_{k=1}^m a_k \delta_{(x_k, y_k)} \in \operatorname{Span}\left\{\delta_{(x,y)},\ (x,y) \in \mathcal{X} \times \mathcal{Y}\right\} \mapsto \sum_{k=1}^m a_k \delta_{(x_k, \varnothing)}.$$

Training loss for learning both conditional and unconditional diffusion model

Let $p_\varnothing \in (0,1)$.

$$\tilde{\ell}(\theta, \hat{\mu}_N, p_\varnothing) = -\sum_{k=1}^n \left[(1-p_\varnothing)\tilde{\mathcal{D}}_k(\theta, \hat{\mu}_N) + p_\varnothing \tilde{\mathcal{D}}_k(\theta, \pi_\varnothing \hat{\mu}_N)\right].$$

with

$$\begin{aligned}\tilde{\mathcal{D}}_k(\theta, \mu) =& \nu_k \int_{\mathcal{X} \times \mathcal{Y} \cup \{\varnothing\}} \int_{\mathbb{R}^d} \left\| \varepsilon - \varepsilon_k(\sqrt{\alpha_k} x_0 + \sqrt{1-\alpha_k}\varepsilon, y, \theta) \right\|^2 \\ & \times \mathcal{N}(0, I_d)(\mathrm{d}\varepsilon)\mu(\mathrm{d}x_0, \mathrm{d}y).\end{aligned}$$

Ref: Ho and Salimans (2022) [5].

Conditional sampling without classifier

Let $c \in \mathcal{Y}$ and $s > 0$,

$$x_n \sim \mathcal{N}(0, I_d)(\mathrm{d}x_n),$$

and for $k = n, \ldots, 1$,

$$x_{k\text{-}1} \mid x_k, c \sim \mathcal{N}\left(\frac{1}{\sqrt{1-\beta_k}}\left(x_k - \frac{\beta_k}{\sqrt{1-\alpha_k}}\tilde{\varepsilon}_k(x_k, c, s)\right), \sigma_k^2 I_d\right),$$

with $\tilde{\varepsilon}_k(x_k, c, s) := \varepsilon_k(x_k, c, \hat{\theta}) + s\left(\varepsilon_k(x_k, c, \hat{\theta}) - \varepsilon_k(x_k, \varnothing, \hat{\theta})\right).$

Ref: Ho and Salimans (2022) [5].

Conditional sampling without classifier

Let $c \in \mathcal{Y}$ and $s > 0$,

$$x_n \sim \mathcal{N}(0, I_d)(\mathrm{d}x_n),$$

and for $k = n, \dots, 1$,

$$x_{k\text{-}1} \mid x_k, c \sim \mathcal{N}\left(\frac{1}{\sqrt{1-\beta_k}}\left(x_k - \frac{\beta_k}{\sqrt{1-\alpha_k}}\tilde{\varepsilon}_k(x_k, c, s)\right), \sigma_k^2 I_d\right),$$

with $\tilde{\varepsilon}_k(x_k, c, s) := \varepsilon_k(x_k, c, \hat{\theta}) + s\left(\varepsilon_k(x_k, c, \hat{\theta}) - \varepsilon_k(x_k, \varnothing, \hat{\theta})\right).$

Interpretation:

$$\varepsilon_k(x_k, c, \hat{\theta}) - \varepsilon_k(x_k, \varnothing, \hat{\theta}) \approx -\nabla_{x_{k\text{-}1}} \log \tilde{p}_{y|k\text{-}1}(c \mid x_k),$$

with $\tilde{p}_{y|k\text{-}1}(c \mid x_{k\text{-}1}) = \dfrac{\tilde{p}_{k\text{-}1}(x_{k\text{-}1} \mid c)}{\tilde{p}_{k\text{-}1}(x_{k\text{-}1} \mid \varnothing)}.$

Ref: Ho and Salimans (2022) [5].

Conditional sampling without classifier

Let $c \in \mathcal{Y}$ and $s > 0$,

$$x_n \sim \mathcal{N}(0, I_d)(\mathrm{d}x_n),$$

and for $k = n, \ldots, 1$,

$$x_{k\text{-}1} \mid x_k, c \sim \mathcal{N}\left(\frac{1}{\sqrt{1-\beta_k}}\left(x_k - \frac{\beta_k}{\sqrt{1-\alpha_k}}\tilde{\varepsilon}_k(x_k, c, s)\right), \sigma_k^2 I_d\right),$$

with $\tilde{\varepsilon}_k(x_k, c, s) := \varepsilon_k(x_k, c, \hat{\theta}) + s\left(\varepsilon_k(x_k, c, \hat{\theta}) - \varepsilon_k(x_k, \varnothing, \hat{\theta})\right)$.

Interpretation:

$$\varepsilon_k(x_k, c, \hat{\theta}) - \varepsilon_k(x_k, \varnothing, \hat{\theta}) \approx -\nabla_{x_{k\text{-}1}} \log \tilde{p}_{y|k\text{-}1}(c \mid x_k),$$

with $\tilde{p}_{y|k\text{-}1}(c \mid x_{k\text{-}1}) = \dfrac{\tilde{p}_{k\text{-}1}(x_{k\text{-}1} \mid c)}{\tilde{p}_{k\text{-}1}(x_{k\text{-}1} \mid \varnothing)}$.

Remark: **$\mathcal{Y}$ does not have to be finite !** We can learn infinite mixture:

$$\mu(\mathrm{d}x) = \int_{\mathcal{Y}} \mu(\mathrm{d}x \mid y)\pi(\mathrm{d}y).$$

Ref: Ho and Salimans (2022) [5].

Selection of $(s, p_\varnothing)$

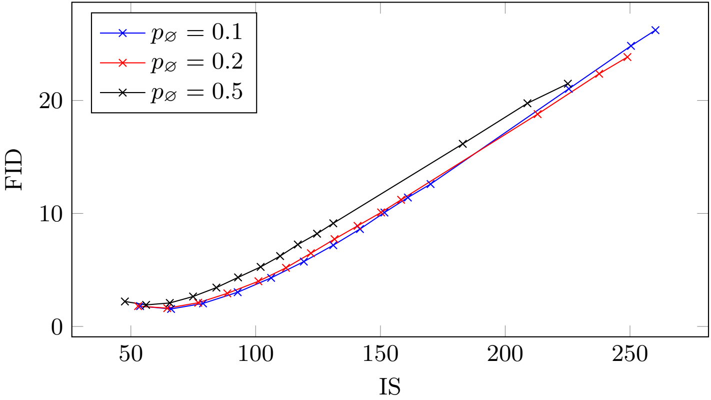


Figure 8: Curves $s \in [0,4] \mapsto (\mathrm{IS}(s, p_\varnothing), \mathrm{FID}(s, p_\varnothing))$ on Image-Net 64×64.

Ref: Ho and Salimans (2022) [5].

Outline

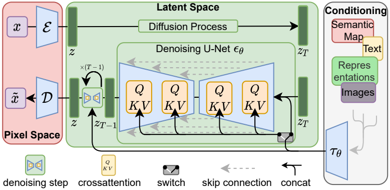
Latent diffusion models

Auto-encoder

$$x \in \mathbb{R}^{H\times W\times 3} \longrightarrow E(x,\theta) \in \mathbb{R}^{h\times w\times c} \text{ encoder},$$
$$\text{decoder } D(z,\theta) \in \mathbb{R}^{H\times W\times 3} \longleftarrow z \in \mathbb{R}^{h\times w\times c} =: \mathcal{Z}.$$

The auto-encoder is trained so that $D(E(x,\theta),\theta) \approx x$ and $E_{\#}(\mu,\theta) \approx \mu_z$ a target distribution, via the minimization of the loss $\mathcal{D}_{ae}(\theta, \hat{\mu}_N)$.

Choice of $\mu_z(\mathrm{d}z) = p_0(z;\theta)\mathrm{d}z = \int_{\mathcal{Y}} p_0(\theta, \tau(y,\theta),\theta)\mathrm{d}y$ a conditional diffusion distribution, with τ an encoder of the conditioner.



Ref: Rombach et al. (2021) [6].

Training loss for the latent diffusion model

Let $\hat{\mu}_N = \sum_{i=1}^{N} \delta_{(x_i, y_i)}$ be the training set.

$$\tilde{\ell}_{\ell b}(\theta, \hat{\mu}_N) = -\mathcal{D}_{ae}(\theta, \hat{\mu}_N) - \sum_{k=1}^{n} \mathcal{D}_k^z(\theta, \hat{\mu}_N),$$

$$\mathcal{D}_k^z(\theta, \hat{\mu}_N) := \nu_k \int_{\mathcal{X} \times \mathcal{Y}} \int_{\mathcal{Z}} \left\| \varepsilon - \varepsilon_k(\sqrt{\alpha_k} E(x_0, \theta) + \sqrt{1 - \alpha_k} \varepsilon, \tau(y, \theta), \theta) \right\|^2 \times \mathcal{N}(0, I_d)(\mathrm{d}\varepsilon) \hat{\mu}_N(\mathrm{d}x_0, \mathrm{d}y).$$

Ref: Rombach et al. (2021) [6].

Figure 9: $y =$"*A painting of the last supper by Picasso.*"

Ref: Rombach et al. (2021) [6].

Layout to image generation

Outline

Unconditional denoising diffusion probabilistic models

Classifier guidance

Classifier-free guidance

Latent diffusion models

Universal guidance

Let us assume that we have a pre-trained diffusion model with score functions $(\varepsilon_k)_{1\leq k\leq n}$. We want to sample an image x subject to the condition $f(x) \approx c$, where c is a prompt and f is a guidance function. The condition can be rewritten equivalently as

$$\mathcal{L}(c, f(x)) \approx 0 \text{ for some loss } \mathcal{L}.$$

Score function for the conditional sampling

$$\tilde{\varepsilon}_k(x_k, c) := \varepsilon_k(x_k) + s_k \nabla_{x_k} \mathcal{L}\left(c, f\left(\hat{x}_0^k(x_k)\right)\right),$$

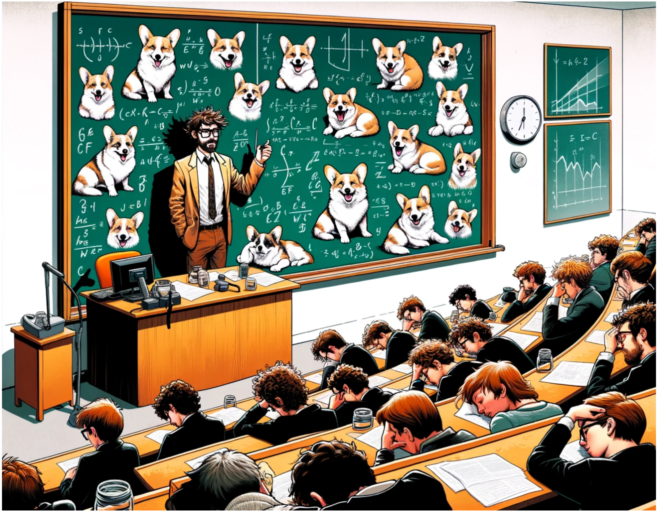
$$\text{with } \hat{x}_0^k(x_k) = \frac{x_k - \sqrt{1-\alpha_k}\varepsilon_k(x_k)}{\sqrt{\alpha_k}}.$$

Ref: Bansal et al. (2023) [1].

Example: image generation from text + style source

Ref: Bansal et al. (2023) [1].

Discussion



Illustrate a scene in the style reminiscent of André Franquin featuring a post-doc researcher presenting in front of a sleeping audience. The researcher, characterized by a clumsy demeanor, stands flustered in front of a blackboard. This researcher's clothing is slightly disheveled, indicating their preoccupation with work over appearance. In a humorous twist, instead of scientific equations or complex data, the blackboard is filled with pictures of Welsh corgis, adding a playful and absurd touch to the scene. The audience, depicted with exaggeratedly humorous sleeping poses, reflects Franquin's knack for dynamic expressions and character designs. Some are slouched over their desks, others have their heads thrown back mid-snore, and one might even have a bubble popping from their

Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Soumyadip Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. “Universal Guidance for Diffusion Models”. In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (2023), pp. 843–852. URL: https://api.semanticscholar.org/CorpusID:256846836 (cit. on pp. 74, 75).

Prafulla Dhariwal and Alex Nichol. “Diffusion Models Beat GANs on Image Synthesis”. In: *ArXiv* abs/2105.05233 (2021). URL: https://api.semanticscholar.org/CorpusID:234357997 (cit. on pp. 46–48, 50–58).

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. “GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium”. In: *Neural Information Processing Systems*. 2017. URL: https://api.semanticscholar.org/CorpusID:326772 (cit. on pp. 43–45).

Jonathan Ho, Ajay Jain, and P. Abbeel. “Denoising Diffusion Probabilistic Models”. In: *ArXiv* abs/2006.11239 (2020). URL: https://api.semanticscholar.org/CorpusID:219955663 (cit. on pp. 10–36).

Jonathan Ho and Tim Salimans. “Classifier-Free Diffusion Guidance”. In: *ArXiv* abs/2207.12598 (2022). URL: https://api.semanticscholar.org/CorpusID:249145348 (cit. on pp. 60–67).

Robin Rombach, A. Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. “High-Resolution Image Synthesis with Latent Diffusion Models”. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2021), pp. 10674–10685. URL: https://api.semanticscholar.org/CorpusID:245335280 (cit. on pp. 69–71).

Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. “Improved Techniques for Training GANs”. In: *ArXiv* abs/1606.03498 (2016). URL: https://api.semanticscholar.org/CorpusID:1687220 (cit. on pp. 40–42).

Yang Song, Jascha Narain Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. "Score-Based Generative Modeling through Stochastic Differential Equations". In: *ArXiv* abs/2011.13456 (2020). URL: https://api.semanticscholar.org/CorpusID:227209335 (cit. on pp. 2–7).

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. "Rethinking the Inception Architecture for Computer Vision". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 2818–2826. URL: https://api.semanticscholar.org/CorpusID:206593880 (cit. on pp. 38, 39, 43–45).