Diffusion Model

Computer Vision - Project

Clément, Grégoire, Nathan

January 9, 2025

Denoising Diffusion Probabilistic Models

General Idea

Consider the set of hand-written digits D. Can you give a probability distribution q such that $x \sim q(x)$?

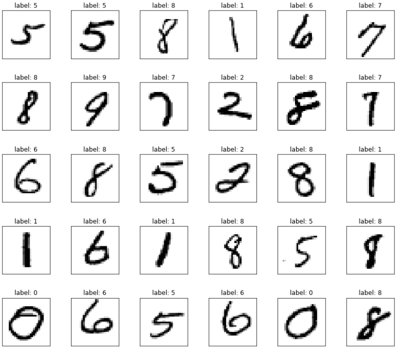
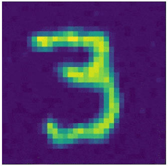


Figure 1: Source: ludwig.ai

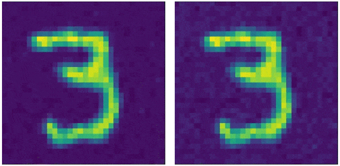
General Idea

Consider the set of hand-written digits D. It is hard to find q such that $x \sim q(x)$, we need a clever way to sample hand-written digits. Consider the following process:



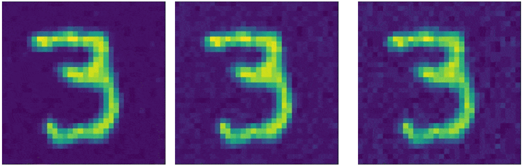
General Idea

Consider the set of hand-written digits D. It is hard to find q such that $x \sim q(x)$, we need a clever way to sample hand-written digits. Consider the following process:



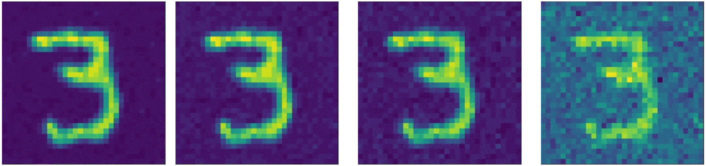
General Idea

Consider the set of hand-written digits D. It is hard to find q such that $x \sim q(x)$, we need a clever way to sample hand-written digits. Consider the following process:



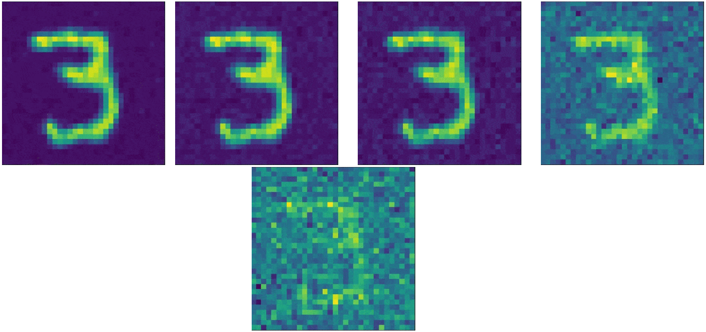
General Idea

Consider the set of hand-written digits D. It is hard to find q such that $x \sim q(x)$, we need a clever way to sample hand-written digits. Consider the following process:



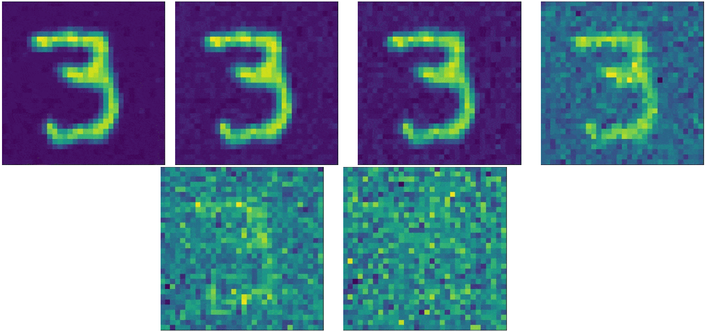
General Idea

Consider the set of hand-written digits D. It is hard to find q such that $x \sim q(x)$, we need a clever way to sample hand-written digits. Consider the following process:



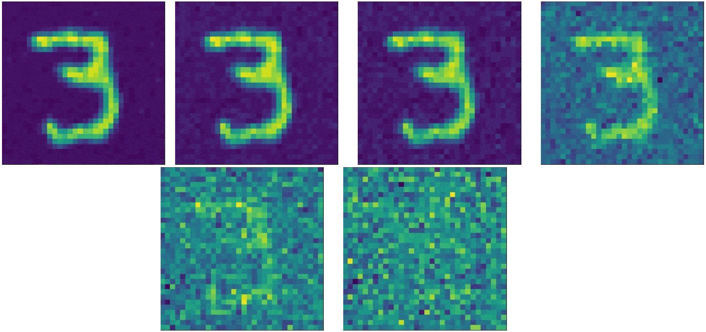
General Idea

Consider the set of hand-written digits D. It is hard to find q such that $x \sim q(x)$, we need a clever way to sample hand-written digits. Consider the following process:



General Idea

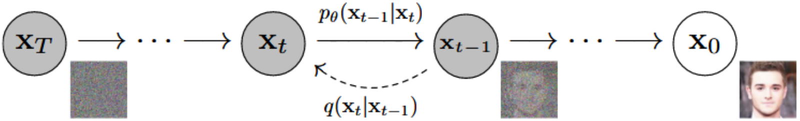
Consider the set of hand-written digits D. It is hard to find q such that $x \sim q(x)$, we need a clever way to sample hand-written digits. Consider the following process:



Formally: $q(x_{t+1} \mid x_t) := \mathcal{N}(x_{t+1}; \sqrt{1-\beta_t}x_t, \beta_t I)$ for some schedule $(\beta_t)_t$. Can we learn to reverse this process ?

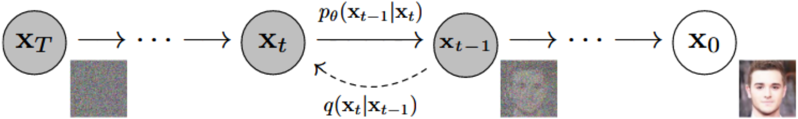
What we want to learn

Given a noisy image x_t, we train a model to predict x_{t-1}.



What we want to learn

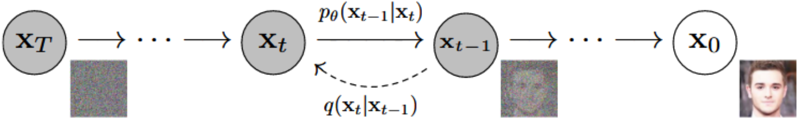
Given a noisy image x_t, we train a model to predict x_{t-1}.



- Given a data image x_0, we sample $(x_t)_{1:T}$ according to $q(x_{1:T} \mid x_0) := \prod_{t=1}^{T} q(x_t \mid x_{t-1})$,

What we want to learn

Given a noisy image x_t, we train a model to predict x_{t-1}.



- Given a data image x_0, we sample $(x_t)_{1:T}$ according to $q(x_{1:T} \mid x_0) := \prod_{t=1}^{T} q(x_t \mid x_{t-1})$,
- Given a noisy image x_t and t, we sample according to $p_\theta(x_{t-1} \mid x_t) := \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$.

Decreasing training data generation cost

Given a data image x_0, compute x_t takes t sampling on q. But a simple trick, allows to do only one.

Decreasing training data generation cost

Given a data image x_0, compute x_t takes t sampling on q. But a simple trick, allows to do only one.

Remember that $q(x_{t+1} \mid x_t) := \mathcal{N}(x_{t+1}; \sqrt{1-\beta_t} x_t, \beta_t I)$. Let $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$.

Decreasing training data generation cost

Given a data image x_0, compute x_t takes t sampling on q. But a simple trick, allows to do only one.

Remember that $q(x_{t+1} \mid x_t) := \mathcal{N}(x_{t+1}; \sqrt{1-\beta_t} x_t, \beta_t I)$. Let $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$.

$$x_t = \sqrt{\alpha_t} x_{t-1} + \sqrt{1-\alpha_t} \epsilon_{t-1}$$

Decreasing training data generation cost

Given a data image x_0, compute x_t takes t sampling on q. But a simple trick, allows to do only one.

Remember that $q(x_{t+1} \mid x_t) := \mathcal{N}(x_{t+1}; \sqrt{1-\beta_t} x_t, \beta_t I)$. Let $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$.

$$\begin{aligned} x_t &= \sqrt{\alpha_t} x_{t-1} + \sqrt{1-\alpha_t} \epsilon_{t-1} \\ &= \sqrt{\alpha_t}\sqrt{\alpha_{t-1}} x_{t-2} + \sqrt{\alpha_t}\sqrt{1-\alpha_t} \epsilon_{t-1} + \sqrt{1-\alpha_t} \epsilon_{t-1} \end{aligned}$$

Decreasing training data generation cost

Given a data image x_0, compute x_t takes t sampling on q. But a simple trick, allows to do only one.

Remember that $q(x_{t+1} \mid x_t) := \mathcal{N}(x_{t+1}; \sqrt{1-\beta_t} x_t, \beta_t I)$. Let $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$.

$$
\begin{aligned}
x_t &= \sqrt{\alpha_t} x_{t-1} + \sqrt{1-\alpha_t} \epsilon_{t-1} \\
&= \sqrt{\alpha_t}\sqrt{\alpha_{t-1}} x_{t-2} + \sqrt{\alpha_t}\sqrt{1-\alpha_t} \epsilon_{t-1} + \sqrt{1-\alpha_t} \epsilon_{t-1}
\end{aligned}
$$

Let $G_1 \sim \mathcal{N}(0, \sigma_1^2 I)$, $G_2 \sim \mathcal{N}(0, \sigma_2^2 I)$, the sum of them gives $g_2 \sim \mathcal{N}(0, (\sigma_1^2 + \sigma_2^2) I)$.

Decreasing training data generation cost

Given a data image x_0, compute x_t takes t sampling on q. But a simple trick, allows to do only one.

Remember that $q(x_{t+1} \mid x_t) := \mathcal{N}(x_{t+1}; \sqrt{1-\beta_t} x_t, \beta_t I)$. Let $\alpha_t = 1 - \beta_t$ and $\bar{\alpha_t} = \prod_{i=1}^{t} \alpha_i$.

$$\begin{aligned} x_t &= \sqrt{\alpha_t} x_{t-1} + \sqrt{1-\alpha_t} \epsilon_{t-1} \\ &= \sqrt{\alpha_t}\sqrt{\alpha_{t-1}} x_{t-2} + \sqrt{\alpha_t}\sqrt{1-\alpha_t} \epsilon_{t-1} + \sqrt{1-\alpha_t} \epsilon_{t-1} \\ &= \sqrt{\alpha_t \alpha_{t-1}} x_{t-2} + \sqrt{\alpha_t(1-\alpha_{t-1}) + 1 - \alpha_t} \bar{\epsilon_t} \end{aligned} \tag{1}$$

Let $G_1 \sim \mathcal{N}(0, \sigma_1^2 I)$, $G_2 \sim \mathcal{N}(0, \sigma_2^2 I)$, the sum of them gives $g_2 \sim \mathcal{N}(0, (\sigma_1^2 + \sigma_2^2) I)$.

Decreasing training data generation cost

Given a data image x_0, compute x_t takes t sampling on q. But a simple trick, allows to do only one.

Remember that $q(x_{t+1} \mid x_t) := \mathcal{N}(x_{t+1}; \sqrt{1-\beta_t} x_t, \beta_t I)$. Let $\alpha_t = 1 - \beta_t$ and $\bar{\alpha_t} = \prod_{i=1}^{t} \alpha_i$.

$$
\begin{aligned}
x_t &= \sqrt{\alpha_t} x_{t-1} + \sqrt{1-\alpha_t} \epsilon_{t-1} \\
&= \sqrt{\alpha_t}\sqrt{\alpha_{t-1}} x_{t-2} + \sqrt{\alpha_t}\sqrt{1-\alpha_t}\epsilon_{t-1} + \sqrt{1-\alpha_t}\epsilon_{t-1} \\
&= \sqrt{\alpha_t \alpha_{t-1}} x_{t-2} + \sqrt{\alpha_t(1-\alpha_{t-1}) + 1 - \alpha_t}\bar{\epsilon_t} \\
&= \sqrt{\alpha_t \alpha_{t-1}} x_{t-2} + \sqrt{1 - \alpha_t \alpha_{t-1}}\bar{\epsilon_t}
\end{aligned} \tag{1}
$$

Let $G_1 \sim \mathcal{N}(0, \sigma_1^2 I)$, $G_2 \sim \mathcal{N}(0, \sigma_2^2 I)$, the sum of them gives $g_2 \sim \mathcal{N}(0, (\sigma_1^2 + \sigma_2^2) I)$.

Decreasing training data generation cost

Given a data image x_0, compute x_t takes t sampling on q. But a simple trick, allows to do only one.

Remember that $q(x_{t+1} \mid x_t) := \mathcal{N}(x_{t+1}; \sqrt{1-\beta_t}x_t, \beta_t I)$. Let $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$.

We have $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$.

Training

For now, our model is learning μ and Σ, i.e. we sample according to

$$p_\theta(x_{t-1} \mid x_t) := \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$$

They've found that fixing Σ_θ to a constant gives the same result. So,

$$p_\theta(x_{t-1} \mid x_t) := \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \sigma_t I)$$

Training

$$p_\theta(x_{t-1} \mid x_t) := \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \sigma_t I)$$

The probability for our model to generate x_0 is $p_\theta(x_0) := \int p_\theta(x_{0:T}) dx_{1:T}$.

Training

$$p_\theta(x_{t-1} \mid x_t) := \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \sigma_t I)$$

The probability for our model to generate x_0 is $p_\theta(x_0) := \int p_\theta(x_{0:T}) dx_{1:T}$. Using negative log likelihood and computatios, we want to minimize:

$$E_q \left[\frac{1}{2\sigma_t^2} \|\tilde{\mu}_t(x_t, x_0) - \mu_\theta(x_t, t)\|^2 \right]$$

where $\tilde{\mu}$ is the optimal mean that depends on x_0 which we don't know.

Training

$$p_\theta(x_{t-1} \mid x_t) := \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \sigma_t I)$$

The probability for our model to generate x_0 is $p_\theta(x_0) := \int p_\theta(x_{0:T}) dx_{1:T}$. Using negative log likelihood and computatios, we want to minimize:

$$E_q \left[\frac{1}{2\sigma_t^2} \|\tilde{\mu}_t(x_t, x_0) - \mu_\theta(x_t, t)\|^2 \right]$$

where $\tilde{\mu}$ is the optimal mean that depends on x_0. Using $x_t(x_0, \epsilon) = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon$ we have a loss we can train on.