Deep Learning:

Denoising Diffusion Probabilistic Models

Ozan Özdenizci

Institute of Theoretical Computer Science

ozan.ozdenizci@igi.tugraz.at

Deep Learning VO - WS 23/24

Lecture 11 - January 8th, 2024

Today

- [ ] Deep Generative Models
  - [x] Variational Autoencoders
  - [x] Generative Adversarial Networks
  - [ ] Denoising Diffusion Probabilistic Models
    - [ ] Idea & Training Objective
    - [ ] Sampling
    - [ ] Further Extensions & Applications

Today's lecture is based on...

Deep Unsupervised Learning using Nonequilibrium Thermodynamics

Jascha Sohl-Dickstein JASCHA@STANFORD.EDU
Stanford University

Eric A. Weiss EWEISS@BERKELEY.EDU
University of California, Berkeley

Niru Maheswaranathan NIRUM@STANFORD.EDU
Stanford University

Surya Ganguli SGANGULI@STANFORD.EDU
Stanford University

International Conference on Machine Learning (ICML) 2015

Denoising Diffusion Probabilistic Models

Jonathan Ho
UC Berkeley
jonathanho@berkeley.edu

Ajay Jain
UC Berkeley
ajayj@berkeley.edu

Pieter Abbeel
UC Berkeley
pabbeel@cs.berkeley.edu

Advances in Neural Information Processing Systems (NeurIPS) 2020

Diffusion Models Beat GANs on Image Synthesis

Prafulla Dhariwal*
OpenAI
prafulla@openai.com

Alex Nichol*
OpenAI
alex@openai.com

Advances in Neural Information Processing Systems (NeurIPS) 2021

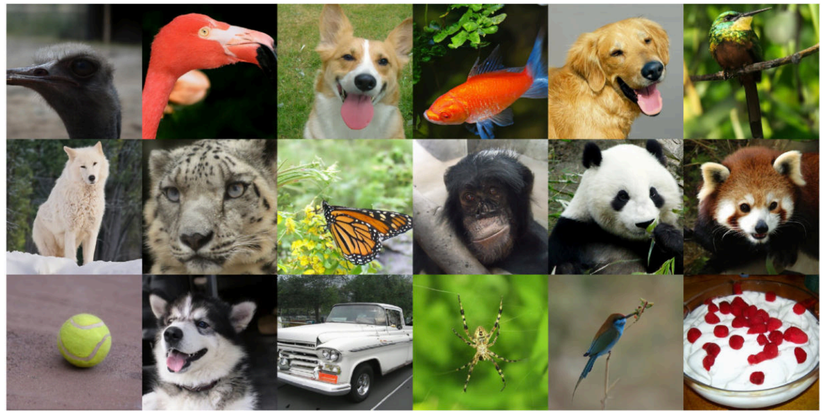
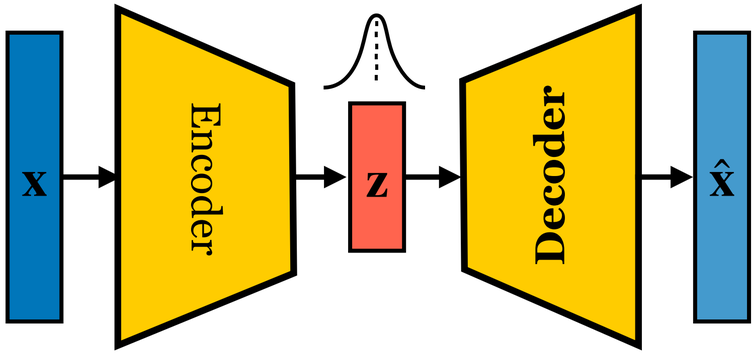


Figure 1: Selected samples from our best ImageNet 512×512 model (FID 3.85)

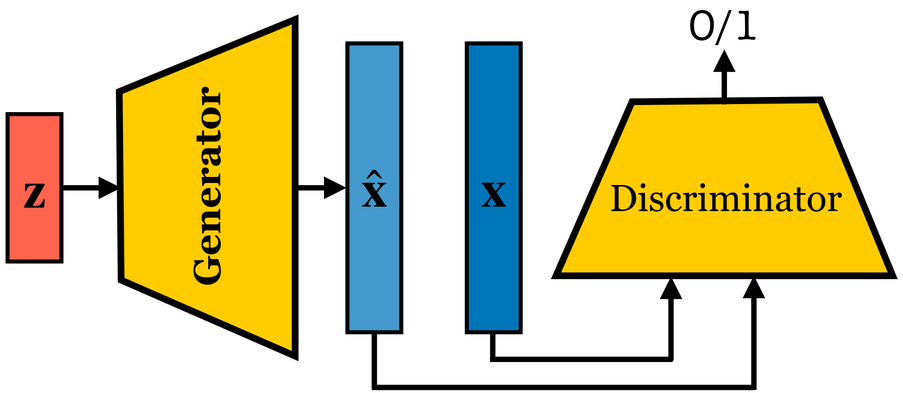
Recap: Deep Generative Models

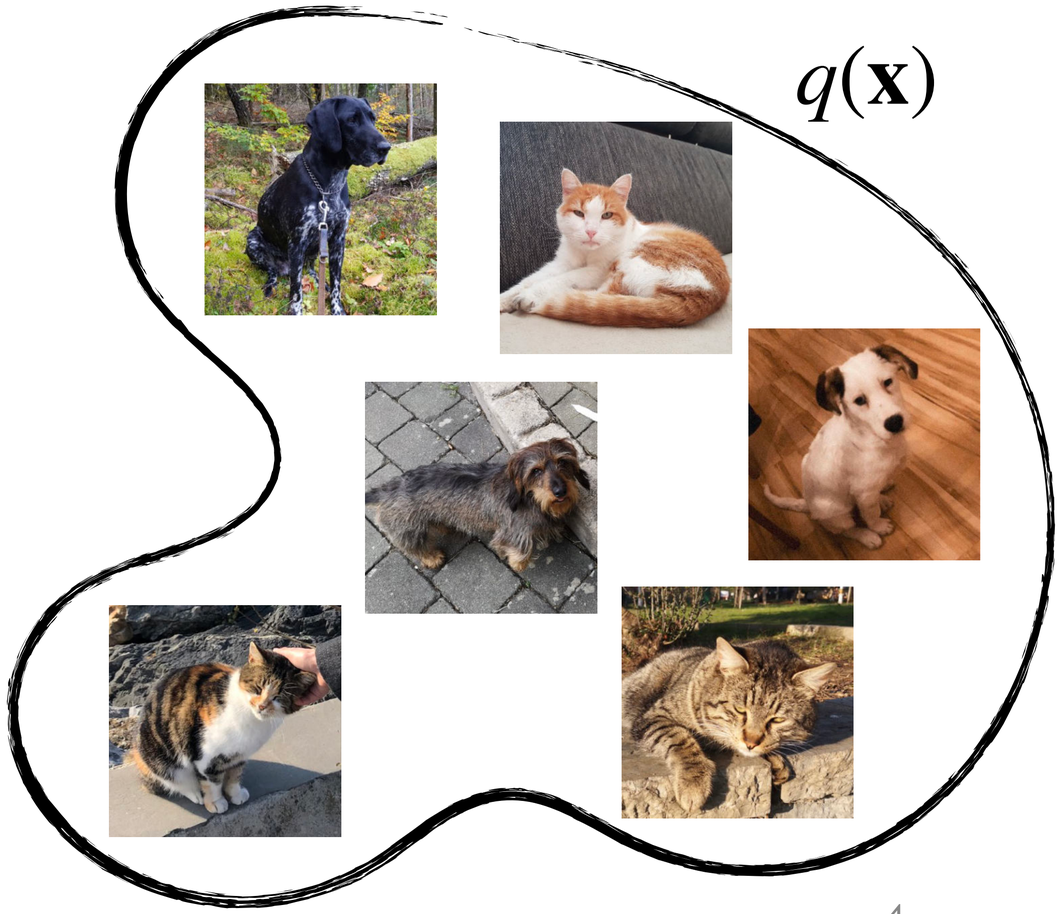
- Given a list of data points $X = \langle \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)} \rangle$ coming from a data distribution $q(\mathbf{x})$.
- **Generative model:** a model $p_\theta(\mathbf{x})$ for this distribution, from which we can **sample**.

Variational Autoencoders

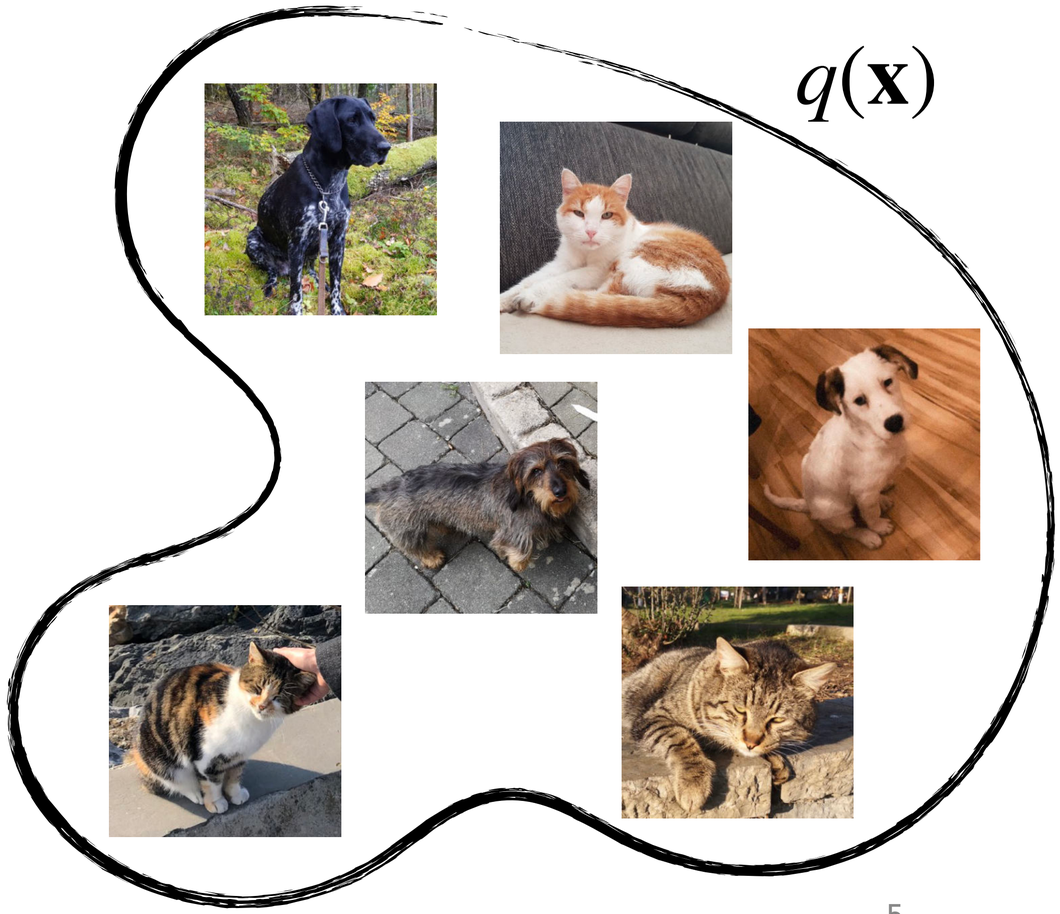


Generative Adversarial Nets

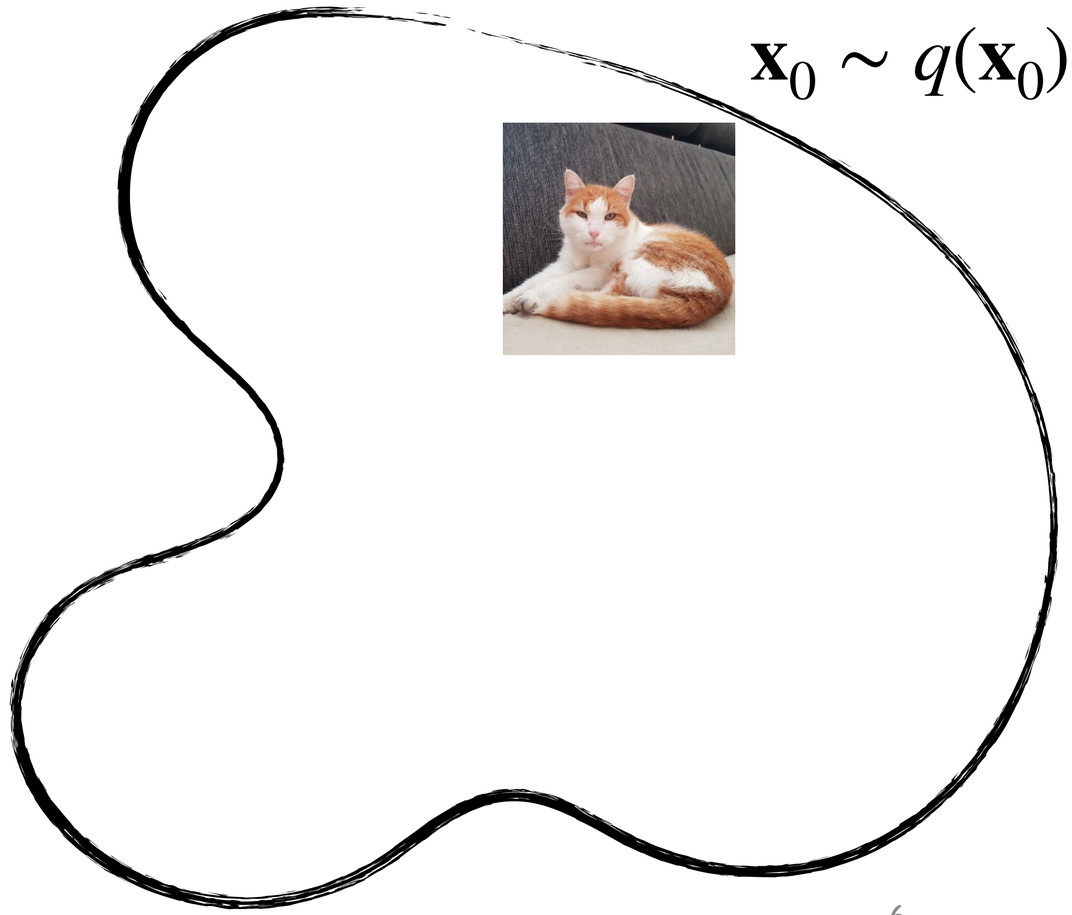




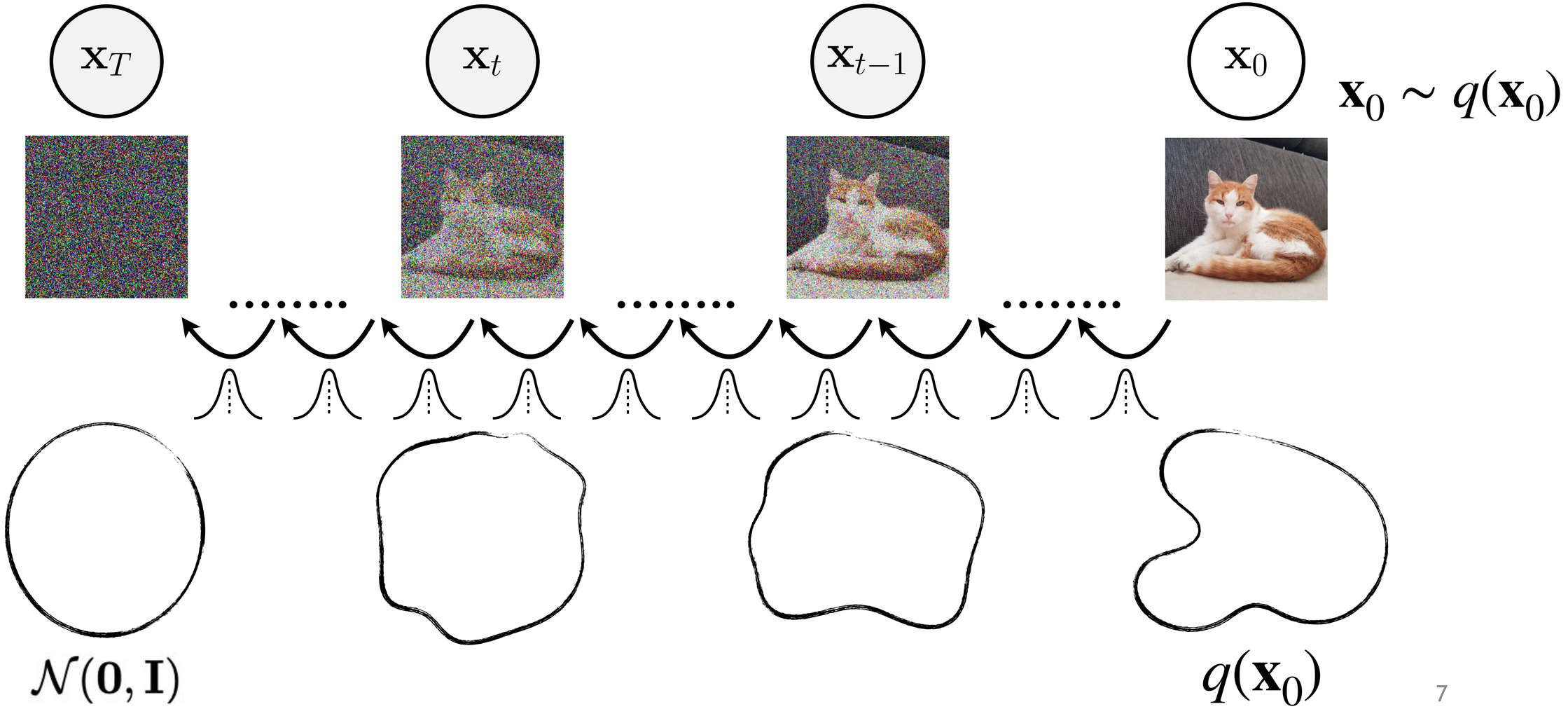
Denoising Diffusion Probabilistic Models (DDPMs)



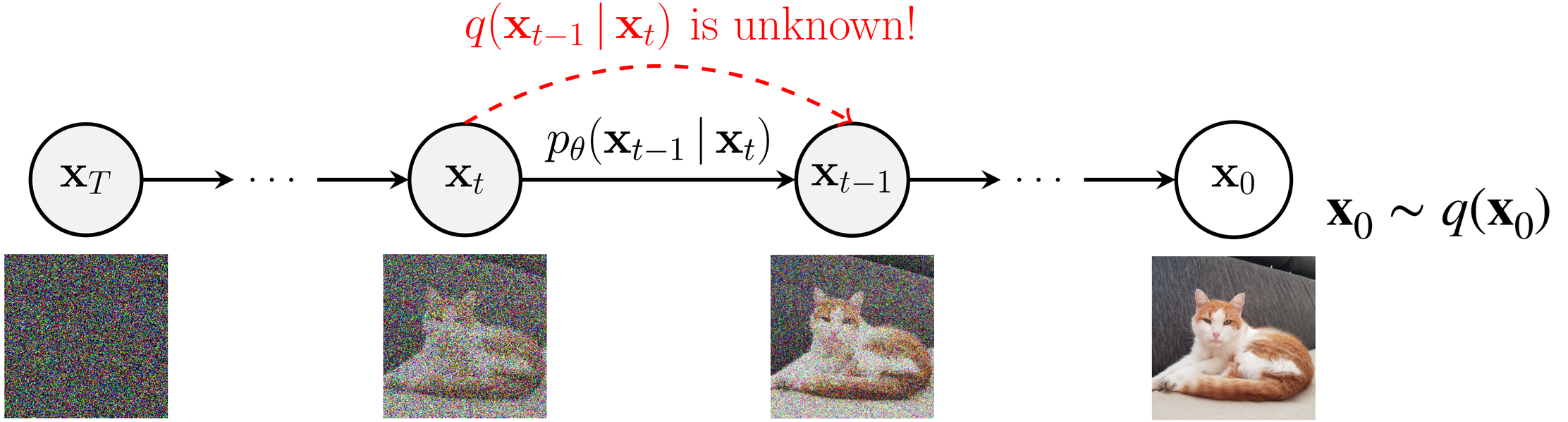
Denoising Diffusion Probabilistic Models (DDPMs)



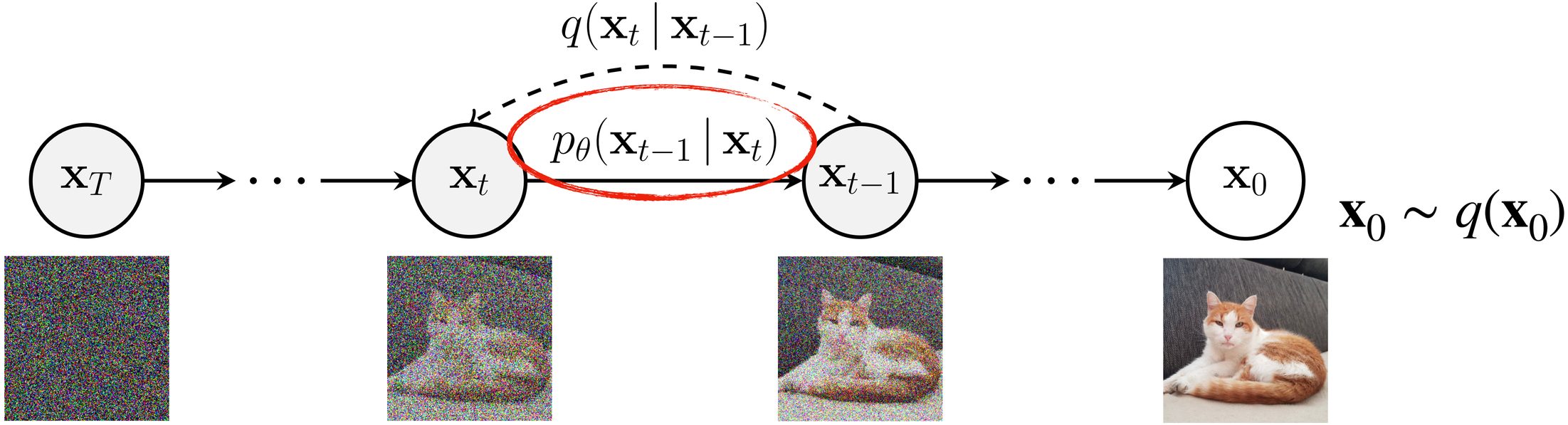
Denoising Diffusion Probabilistic Models (DDPMs)



Denoising Diffusion Probabilistic Models (DDPMs)



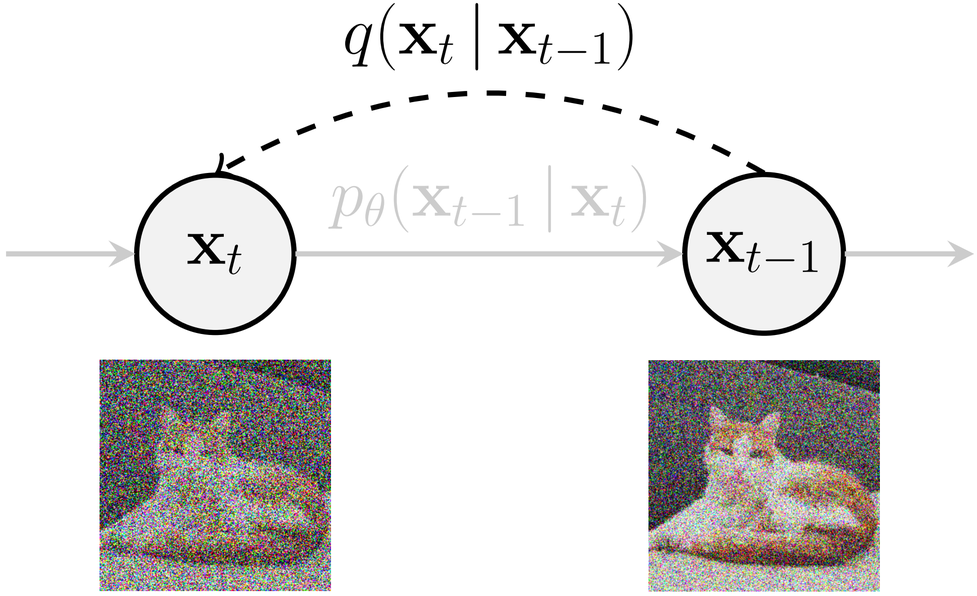
Denoising Diffusion Probabilistic Models (DDPMs)



Central idea: Training a deep neural network that iteratively denoises an image through a *reverse process* that corresponds to a *diffusion process* which destroys the data structure by adding noise.

Fixed Forward (Diffusion) Process

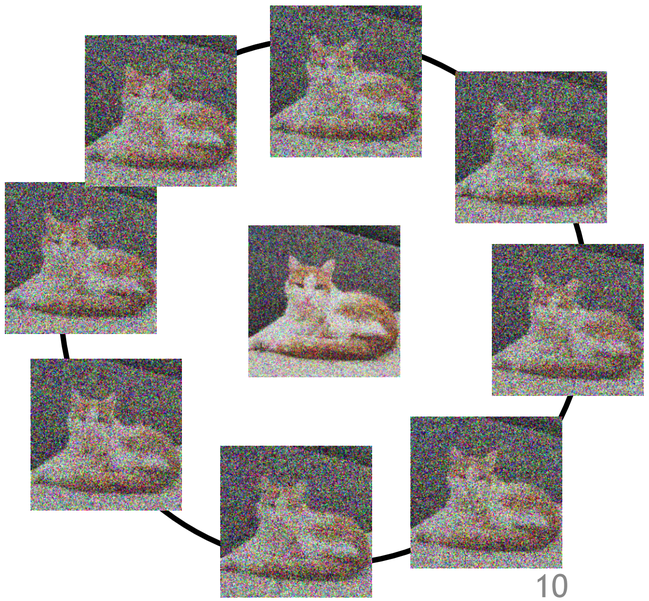
forward process



- The ***forward*** process (i.e., ***diffusion*** process) gradually adds Gaussian noise according to a known variance schedule $\beta_1 < \beta_2 < \ldots < \beta_T$.

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}, \beta_t\,\mathbf{I})$$

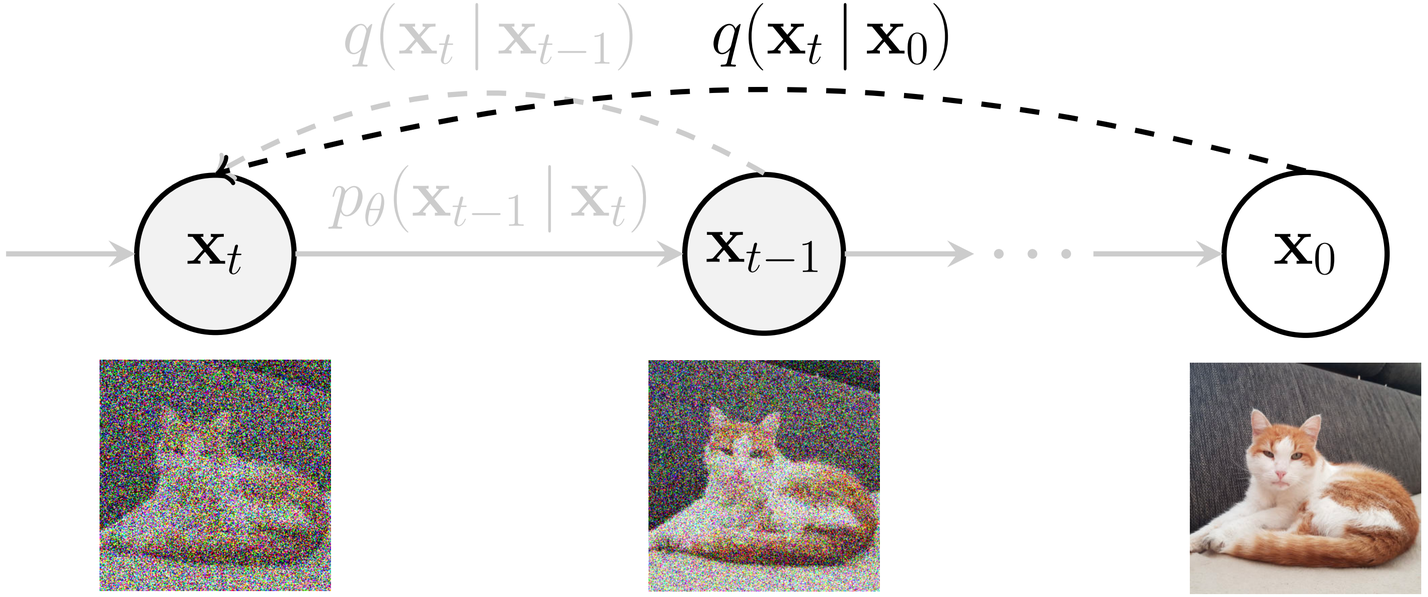
$q(\mathbf{x}_{1:T} \mid \mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t \mid \mathbf{x}_{t-1})$ ➜ joint distribution



$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}, \beta_t\,\mathbf{I})$$

Fixed Forward (Diffusion) Process

forward process



- We can also directly jump to any time-step using:

$$\alpha_t = 1 - \beta_t \quad \text{and} \quad \bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$$

$$q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0, (1-\bar{\alpha}_t)\,\mathbf{I})$$

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon} \quad \text{where} \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

the noise schedule is designed such that:

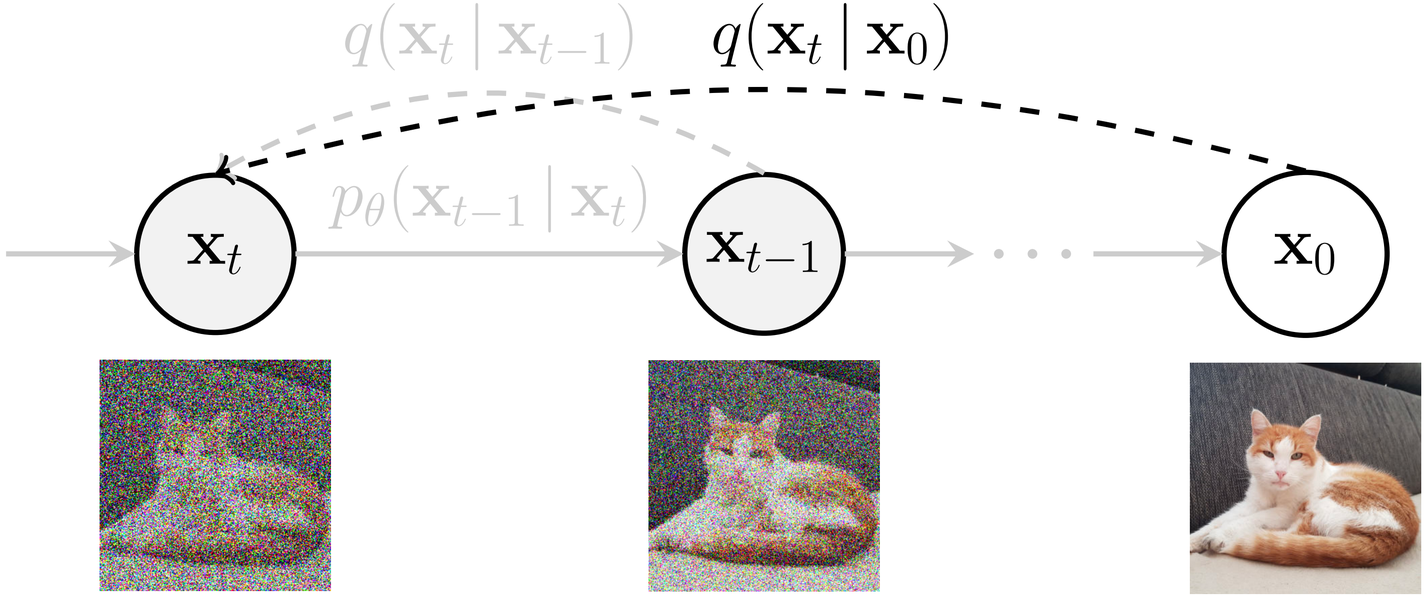
$$\bar{\alpha}_T \to 0$$

$$\bar{\alpha}_1 > \cdots > \bar{\alpha}_T$$

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}, \beta_t\,\mathbf{I})$$

Fixed Forward (Diffusion) Process

forward process



- We can also directly jump to any time-step using:

$$\alpha_t = 1 - \beta_t \quad \text{and} \quad \bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$$

$$q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0, (1-\bar{\alpha}_t)\,\mathbf{I})$$

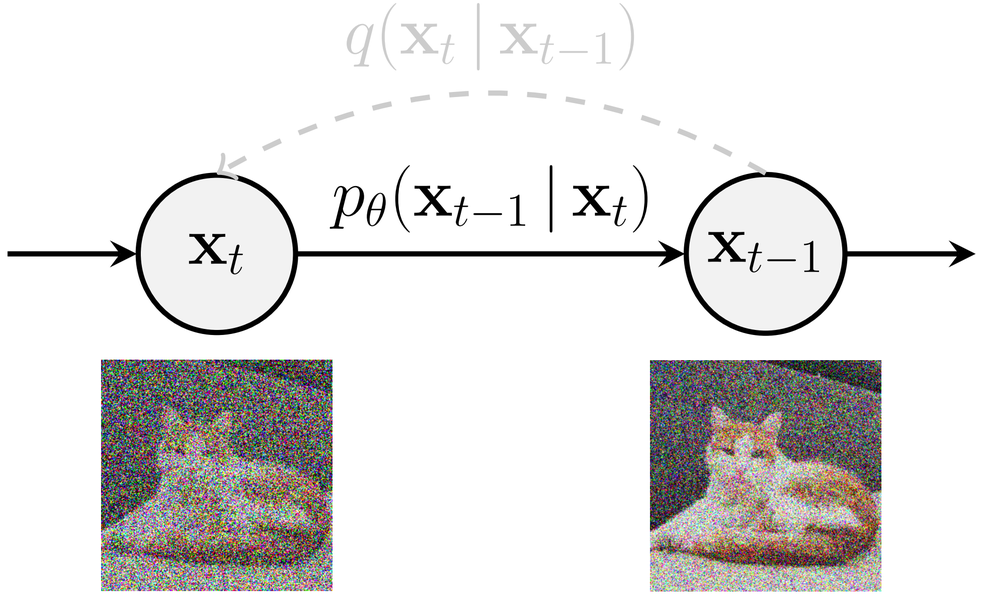
$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon} \quad \text{where} \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

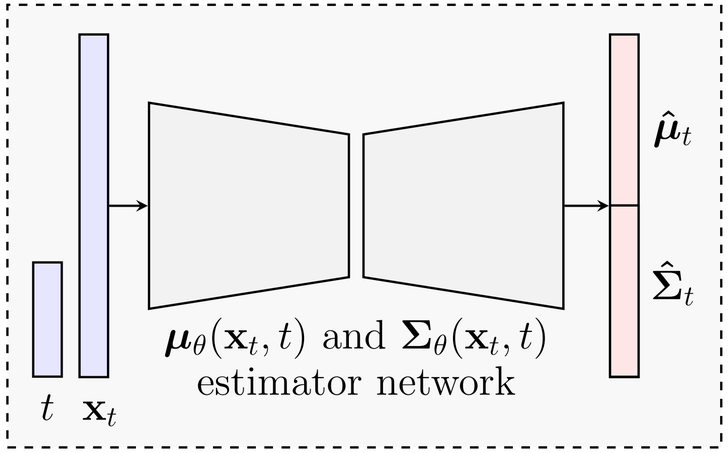
consecutively merging Gaussians:

$$\begin{aligned}
\mathbf{x}_t &= \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1} \\
&= \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\bar{\boldsymbol{\epsilon}}_{t-2} \\
&= \dots \\
&= \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}
\end{aligned}$$

Generative Reverse (Denoising) Process

reverse process





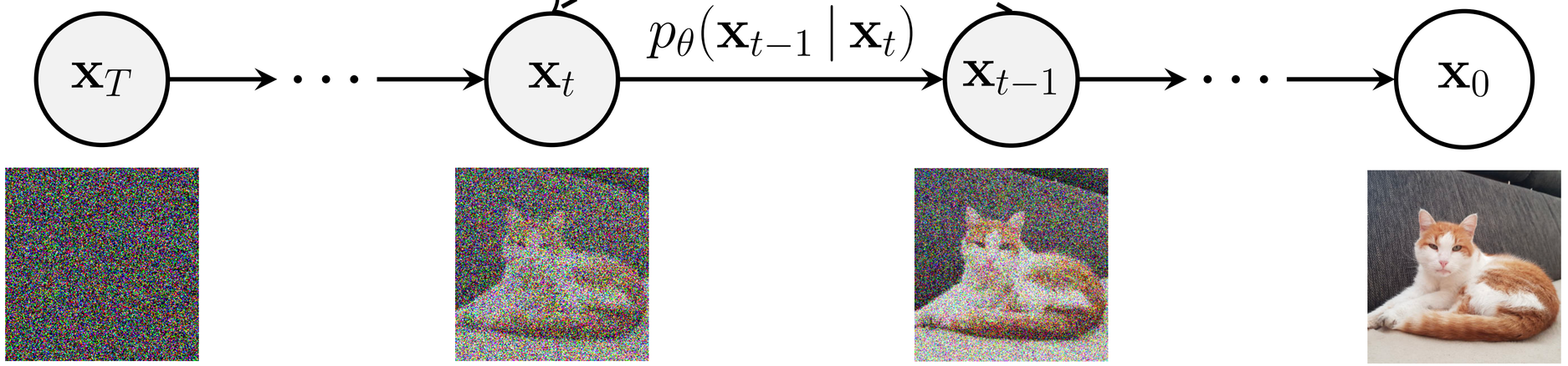
- The ***reverse*** process is the joint distribution, with learned Gaussian transitions starting from noise.

$$p_\theta(\mathbf{x}_{0:T}) = \underbrace{p_\theta(\mathbf{x}_T)}_{\text{known}} \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$$

$$p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I}) \qquad p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t))$$

Denoising Diffusion Probabilistic Models (DDPMs)

Summary: Gradually add Gaussian noise and then reverse.



$$\mathbf{x}_0 \sim q(\mathbf{x}_0)$$

forward process

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}, \beta_t\,\mathbf{I})$$

$$q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0, (1-\bar{\alpha}_t)\,\mathbf{I})$$

reverse process

$$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t))$$

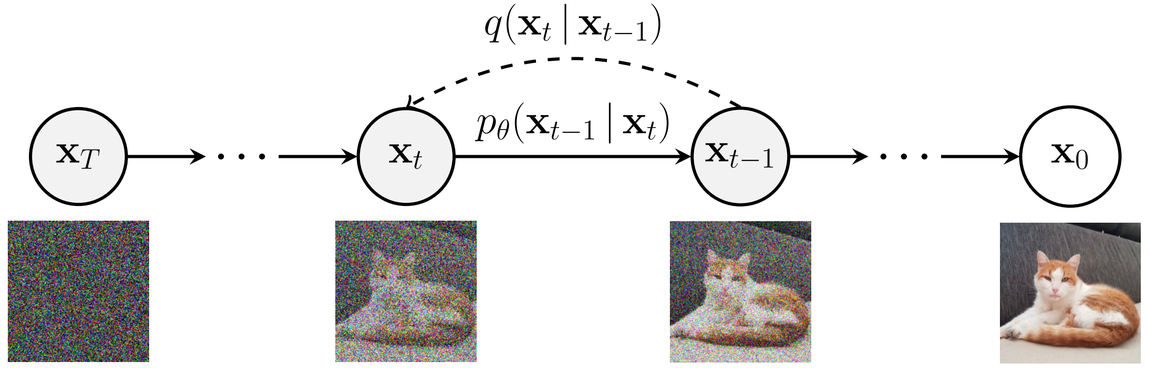
Training: Denoising Diffusion Probabilistic Models

Maximize: $\mathbb{E}_{q(\mathbf{x}_0)}[\log p_\theta(\mathbf{x}_0)]$

$$\mathbb{E}_{q(\mathbf{x}_0)}[\log p_\theta(\mathbf{x}_0)] \geq \mathbb{E}_q \left[\log p(\mathbf{x}_T) + \sum_{t=1}^{T} \log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)}{q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})} \right]$$

Ł, where Ł is the ELBO

Training is similar to VAEs, by maximizing an evidence lower-bound (ELBO) on the log-likelihood.



reverse process: $p_\theta(\mathbf{x}_{0:T}) = p_\theta(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$

$$p_\theta(\mathbf{x}_0) = \int p_\theta(\mathbf{x}_{0:T}) d\,\mathbf{x}_{1:T}$$

$$\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0) = \log \left[\int d\boldsymbol{x}_{1:T} q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0) \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_{0:T})}{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \right]$$

$$\geq \int d\boldsymbol{x}_{1:T} q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0) \log \left(p(\boldsymbol{x}_T) \prod_{t=1}^{T} \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)}{q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})} \right)$$

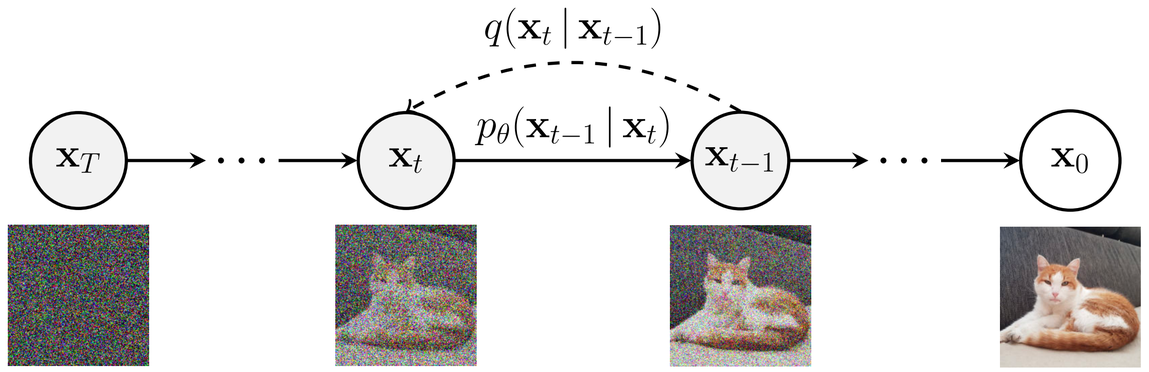
For details to the derivation: Kevin P. Murphy, “Probabilistic Machine Learning: Advanced Topics”, MIT Press, 2023.

Training: Denoising Diffusion Probabilistic Models

Maximize: $\mathbb{E}_{q(\mathbf{x}_0)}[\log p_\theta(\mathbf{x}_0)]$

$$\mathbb{E}_{q(\mathbf{x}_0)}[\log p_\theta(\mathbf{x}_0)] \geq \mathbb{E}_q \left[\log p(\mathbf{x}_T) + \sum_{t=1}^{T} \log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)}{q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})} \right]$$

Ł, where Ł is the ELBO



Training is similar to VAEs, by maximizing an evidence lower-bound (ELBO) on the log-likelihood.

plug in the following: $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{x}_0) = \frac{q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) q(\boldsymbol{x}_t|\boldsymbol{x}_0)}{q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_0)}$

$\cdots$

$$\text{Ł} = -\mathbb{E}_{q(\boldsymbol{x}_0)} \left[\underbrace{D_{\mathbb{KL}}\left(q(\boldsymbol{x}_T|\boldsymbol{x}_0) \| \, p(\mathbf{x}_T)\right)}_{L_T} + \sum_{t=2}^{T} \underbrace{D_{\mathbb{KL}}\left(q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) \| p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)\right)}_{L_{t-1}} \underbrace{- \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1)}_{L_0} \right]$$

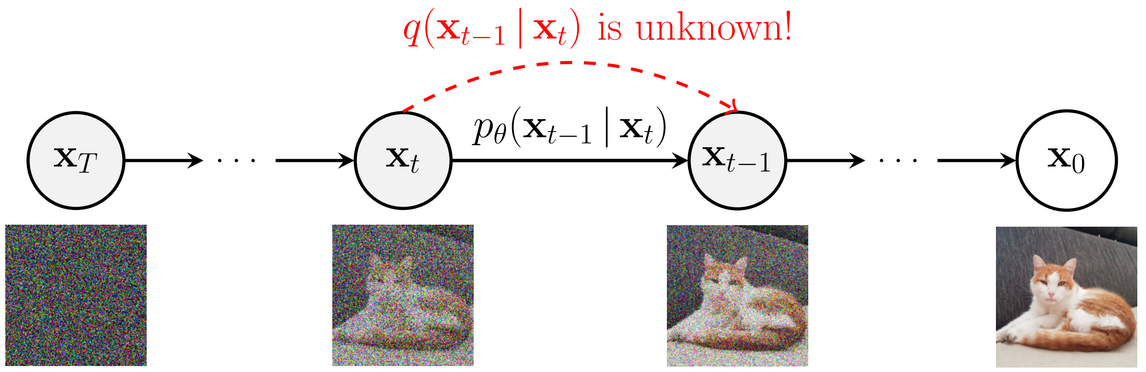
For details to the derivation: Kevin P. Murphy, “Probabilistic Machine Learning: Advanced Topics”, MIT Press, 2023.

Training: Denoising Diffusion Probabilistic Models

Maximize: $\mathbb{E}_{q(\mathbf{x}_0)}[\log p_\theta(\mathbf{x}_0)]$

$$\mathbb{E}_{q(\mathbf{x}_0)}[\log p_\theta(\mathbf{x}_0)] \geq \mathbb{E}_q \left[\log p(\mathbf{x}_T) + \sum_{t=1}^{T} \log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)}{q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})} \right]$$

Ł, where Ł is the ELBO



Training is similar to VAEs, by maximizing an evidence lower-bound (ELBO) on the log-likelihood.

plug in the following: $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{x}_0) = \frac{q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) q(\boldsymbol{x}_t|\boldsymbol{x}_0)}{q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_0)}$

$\cdots$

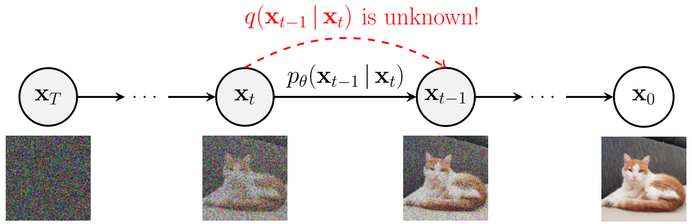
$$\text{Ł} = -\mathbb{E}_{q(\boldsymbol{x}_0)} \left[\underbrace{D_{\mathbb{KL}}\left(q(\boldsymbol{x}_T|\boldsymbol{x}_0) \| \, p(\mathbf{x}_T)\right)}_{L_T} + \sum_{t=2}^{T} \underbrace{D_{\mathbb{KL}}\left(q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) \| p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)\right)}_{L_{t-1}} \underbrace{- \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1)}_{L_0} \right]$$

We can compute an approximation to the overall objective by sampling random time points and optimizing a single L_t each time.

For details to the derivation: Kevin P. Murphy, "Probabilistic Machine Learning: Advanced Topics", MIT Press, 2023.

Training: Denoising Diffusion Probabilistic Models

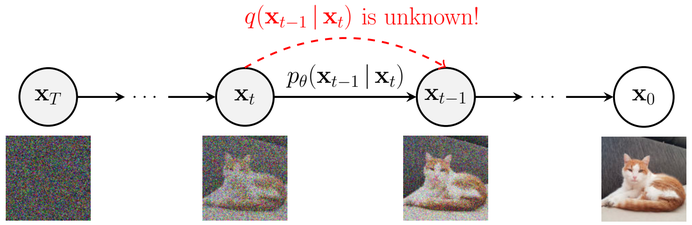
$$\text{Ł} = -\mathbb{E}_{q(\boldsymbol{x}_0)} \left[\underbrace{D_{\mathbb{KL}}\left(q(\boldsymbol{x}_T|\boldsymbol{x}_0) \| \, p(\mathbf{x}_T)\right)}_{L_T} + \sum_{t=2}^{T} \underbrace{D_{\mathbb{KL}}\left(q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) \| p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)\right)}_{L_{t-1}} - \underbrace{\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1)}_{L_0} \right]$$



Minimizing the KL divergence between the modeled reverse process transition probability and the true (unknown) reverse process conditioned on the input.

Training: Denoising Diffusion Probabilistic Models

$$Ł = -\mathbb{E}_{q(\boldsymbol{x}_0)} \left[\underbrace{D_{\mathbb{KL}}\left(q(\boldsymbol{x}_T|\boldsymbol{x}_0) \| p(\mathbf{x}_T)\right)}_{L_T} + \sum_{t=2}^{T} \underbrace{D_{\mathbb{KL}}\left(q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) \| p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)\right)}_{L_{t-1}} - \underbrace{\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1)}_{L_0} \right]$$



Minimizing the KL divergence between the modeled reverse process transition probability and the true (unknown) reverse process conditioned on the input.

This is tractable when conditioned on $\boldsymbol{x}_0$:

$$q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) \propto q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \cancel{\boldsymbol{x}_0}) q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_0)$$

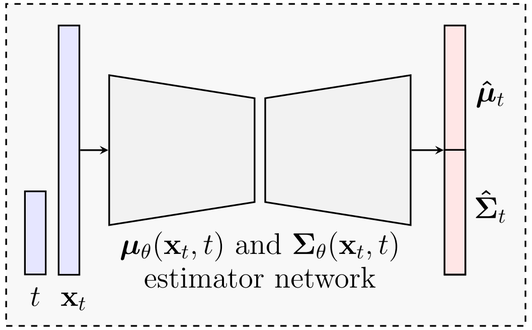
$$= \mathcal{N}(\boldsymbol{x}_{t-1}| \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t, \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t \mathbf{I})$$

Using our forward process properties:

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}, \beta_t \mathbf{I})$$

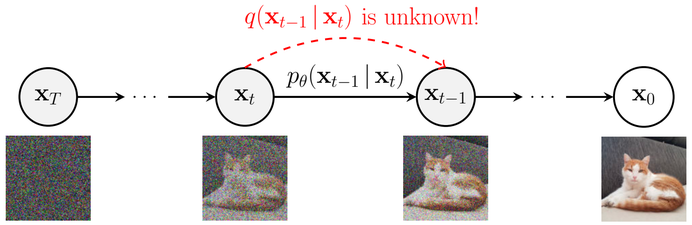
$$q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0, (1-\bar{\alpha}_t)\,\mathbf{I})$$

Optimize $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t))$ to follow $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$ via KL-divergence between Gaussians.



Training: Denoising Diffusion Probabilistic Models

$$\text{Ł} = -\mathbb{E}_{q(\boldsymbol{x}_0)} \left[\underbrace{D_{\mathbb{KL}}\left(q(\boldsymbol{x}_T|\boldsymbol{x}_0) \| \, p(\mathbf{x}_T)\right)}_{L_T} + \sum_{t=2}^{T} \underbrace{D_{\mathbb{KL}}\left(q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) \| p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)\right)}_{L_{t-1}} \underbrace{- \log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1)}_{L_0} \right]$$



Minimizing the KL divergence between the modeled reverse process transition probability and the true (unknown) reverse process conditioned on the input.

This is tractable when conditioned on $\boldsymbol{x}_0$:

$$q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) \propto q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \cancel{\boldsymbol{x}_0}) q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_0)$$

$$= \mathcal{N}(\boldsymbol{x}_{t-1}| \, \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t \, , \, \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t \mathbf{I})$$

Using our forward process properties:

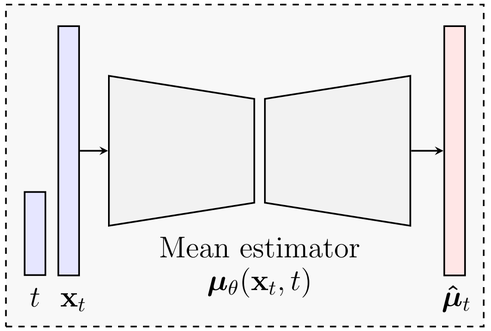
$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}, \beta_t \mathbf{I})$$

$$q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})$$

Optimize $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t))$ to follow $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$ via KL-divergence between Gaussians.

Simplify (1): We often set $\Sigma_\theta(\mathbf{x}_t, t) = \sigma_t^2 \mathbf{I}$, as untrained time-dependent constants, and optimize a network to only predict the mean:

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



Training: Denoising Diffusion Probabilistic Models

Optimize $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \sigma_t^2 \mathbf{I})$ to follow $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$ via KL-divergence between Gaussians.

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

Simplify (2): We can re-parameterize the objective to predict the sampled noise ϵ for time t, instead of predicting the mean.

$$q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_{t-1}| \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t, \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t \mathbf{I})$$

Using forward process properties:

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

Training: Denoising Diffusion Probabilistic Models

Optimize $p_\theta(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \sigma_t^2\mathbf{I})$ to follow $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$ via KL-divergence between Gaussians.

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

Simplify (2): We can re-parameterize the objective to predict the sampled noise ϵ for time t, instead of predicting the mean.

$$q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_{t-1}|\ \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t\,,\ \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t\mathbf{I})$$

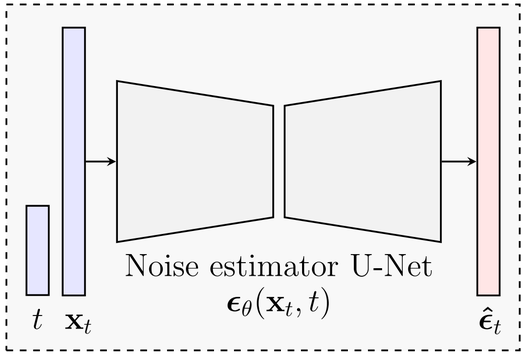
$$= \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}\right)$$

$\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$

Optimize a noise estimator $\epsilon_\theta(\mathbf{x}_t, t)$ to predict the noise added to $\mathbf{x}_0$ at time t:

$$L_{\text{simple}}(\theta) := \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2\right]$$

such that one can predict $\mathbf{x}_0$ via: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x}_t, t)\right) + \sigma_t\boldsymbol{z}$ $\quad \boldsymbol{z} \sim \mathcal{N}(0, \mathbf{I})$



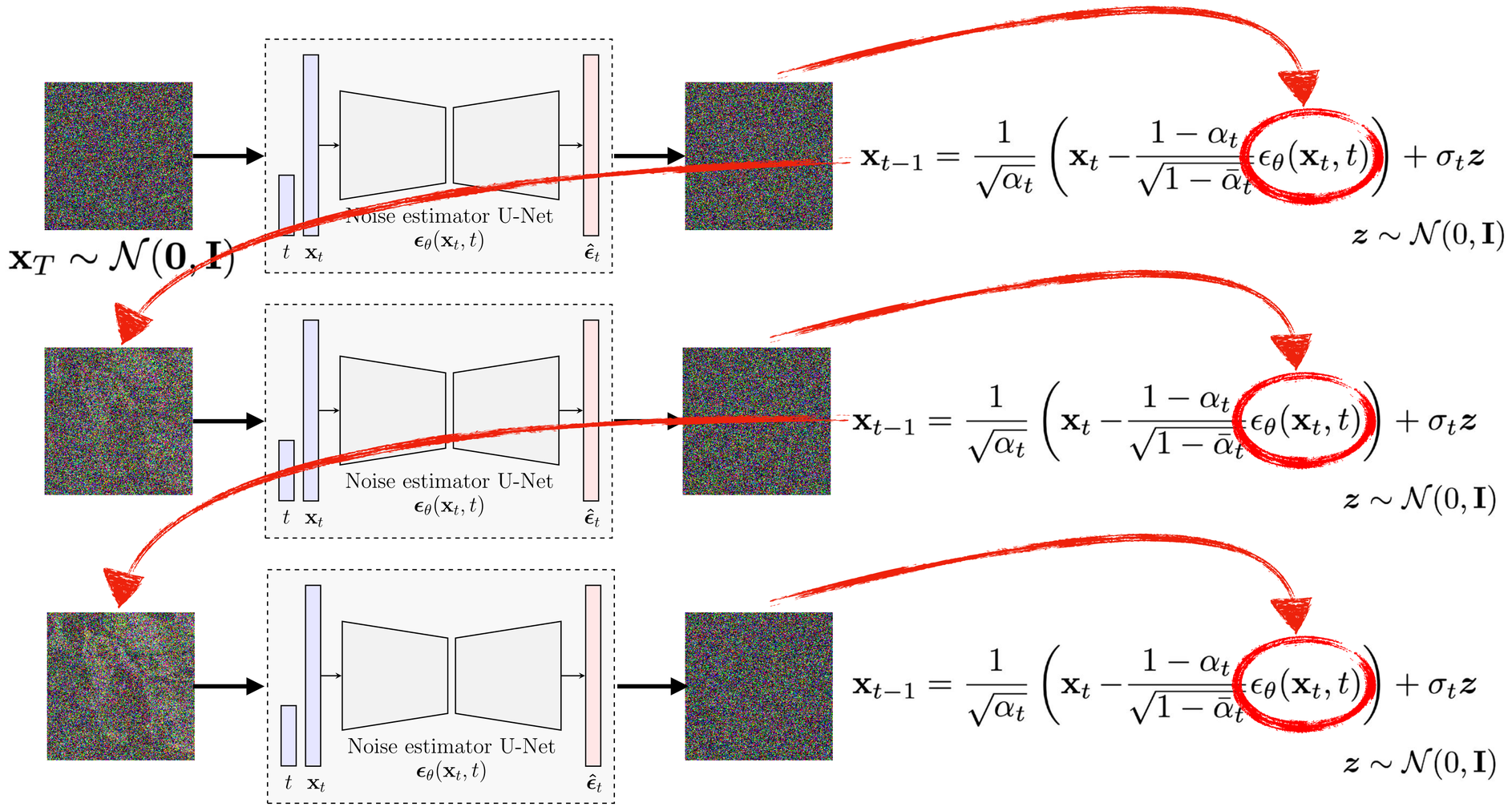
Today

- [ ] Deep Generative Models
  - [x] Variational Autoencoders
  - [x] Generative Adversarial Networks
  - [ ] Denoising Diffusion Probabilistic Models
    - [x] Idea & Training Objective
    - [ ] Sampling
    - [ ] Further Extensions & Applications

Sampling: Denoising Diffusion Probabilistic Models

$\beta_1 < \beta_2 < \ldots < \beta_T$

$T = 1000$



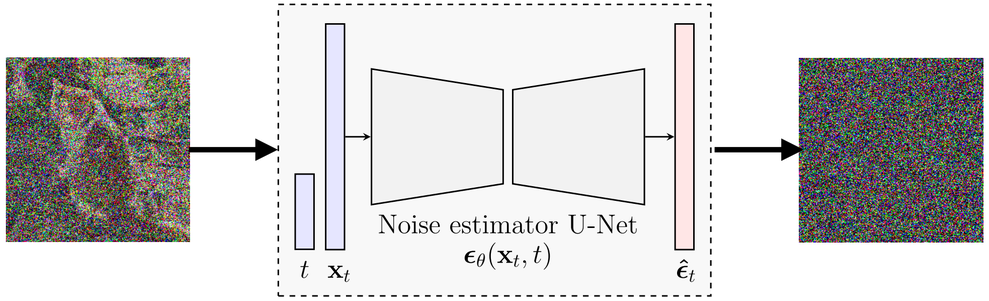
.....

Sampling: Denoising Diffusion Probabilistic Models

$\beta_1 < \beta_2 < \ldots < \beta_T$

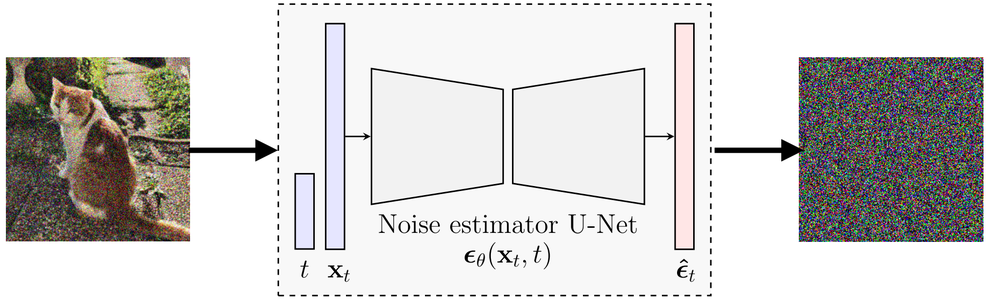
$T = 1000$

$t = 100$



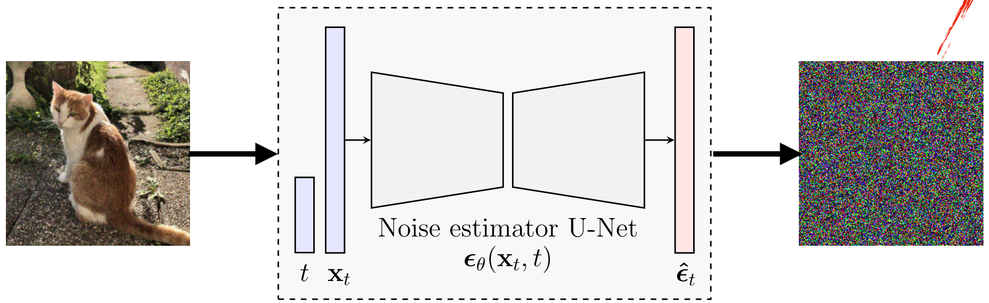
.....

$t = 50$



.....

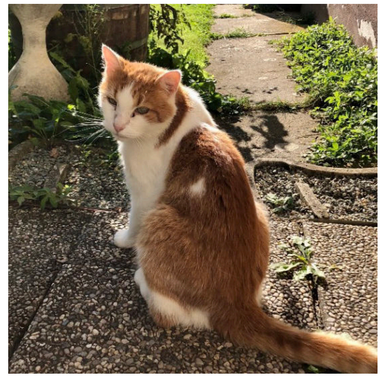
$t = 1$



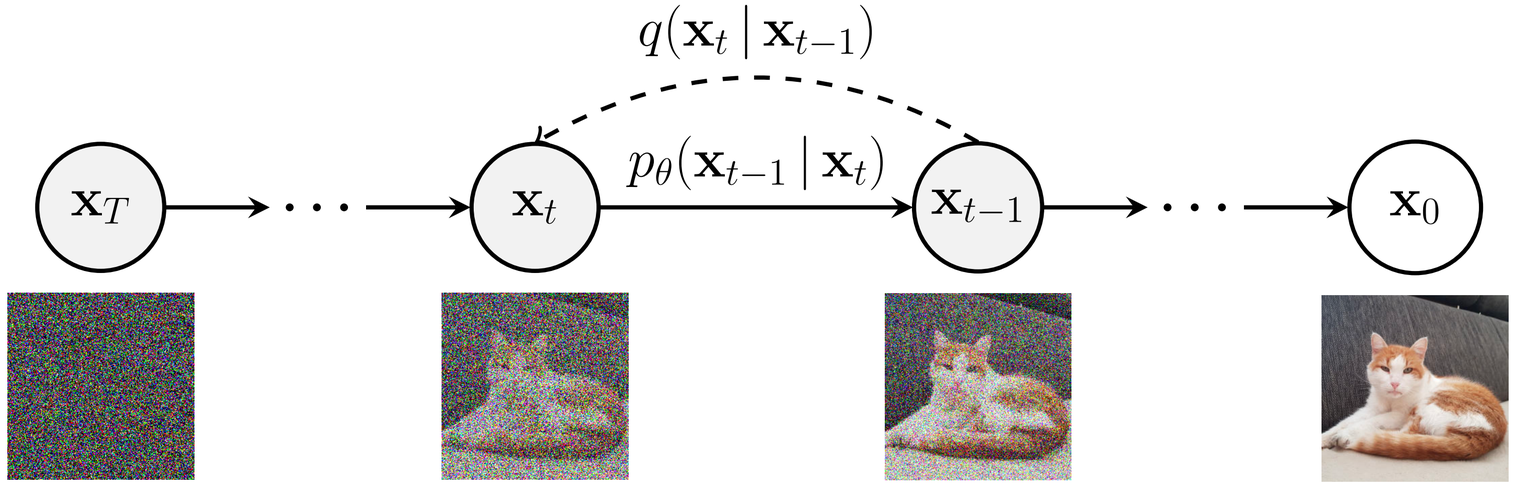
$t = 0$

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

$\mathbf{x}_0 \sim p_\theta(\mathbf{x}_0)$



Algorithm: Denoising Diffusion Probabilistic Models

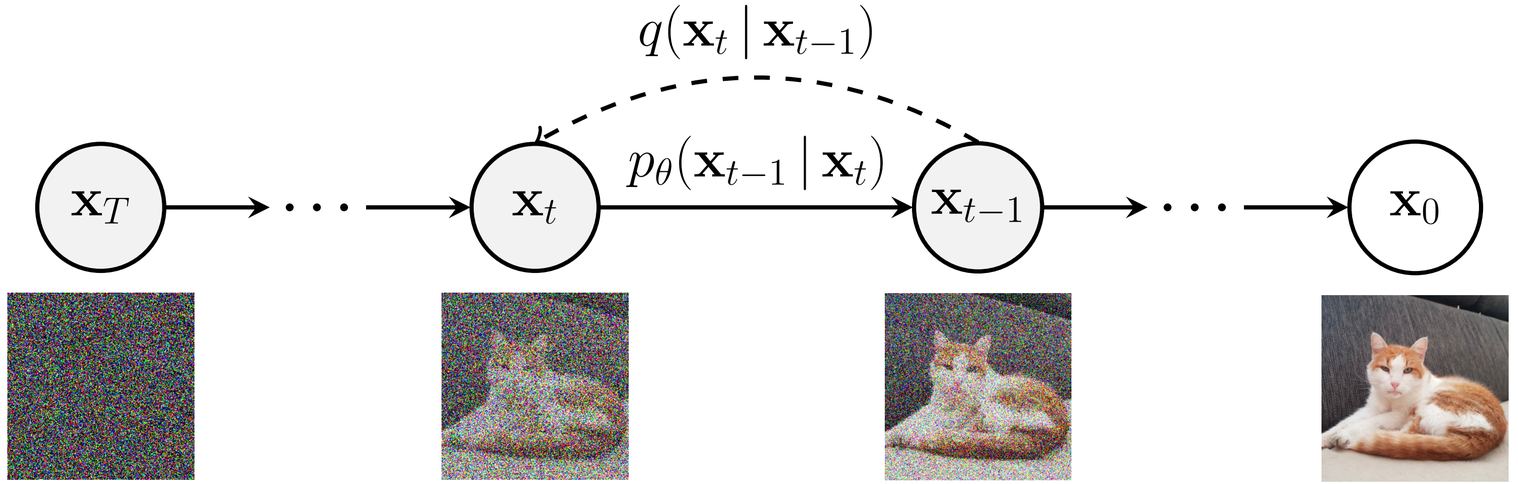


Algorithm 1 Training

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

Optimizing model parameters that predicts the "added noise" that was sampled, using $\boldsymbol{t}$ and $\mathbf{x}_t$

Algorithm: Denoising Diffusion Probabilistic Models



Algorithm 1 Training

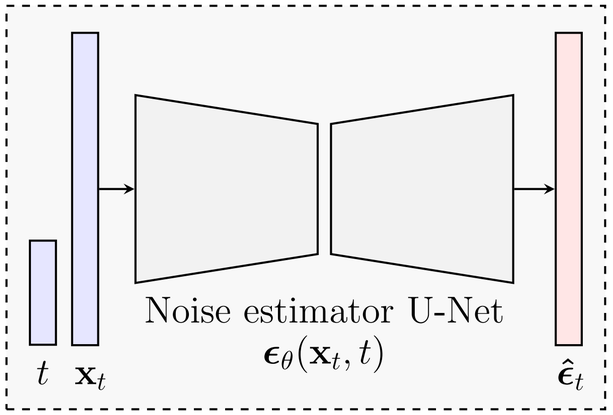
1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0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

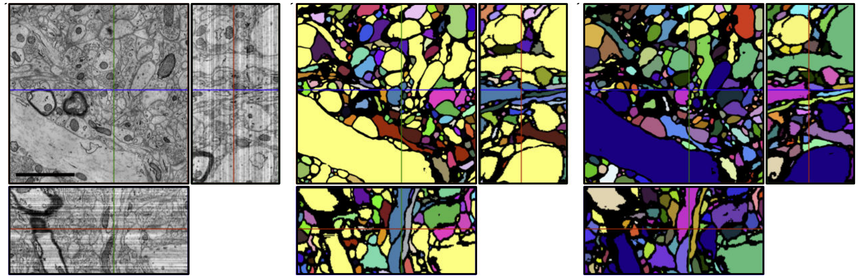
Algorithm 2 Sampling

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** $\mathbf{x}_0$

Sequentially remove the estimated "added noise" starting from an image sampled from a Gaussian noise distribution.

Image-to-Image Network Architecture: U-Net





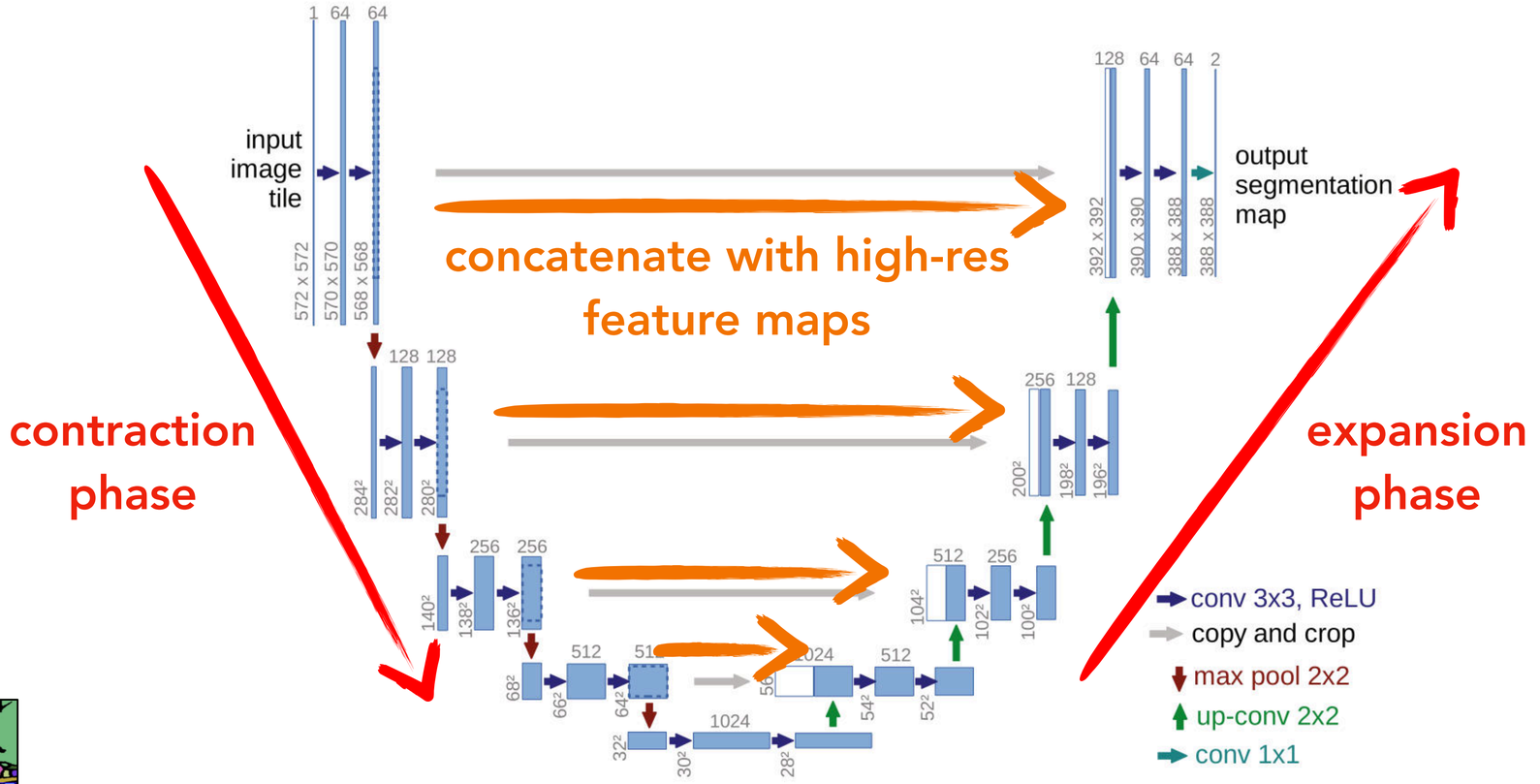


Fig. 1. U-net architecture (example for 32x32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations.

Olaf Ronneberger et al., "*U-Net: Convolutional Networks for Biomedical Image Segmentation*", International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI), 2015.

Today

- [ ] Deep Generative Models
  - [x] Variational Autoencoders
  - [x] Generative Adversarial Networks
  - [ ] Denoising Diffusion Probabilistic Models
    - [x] Idea & Training Objective
    - [x] Sampling
    - [ ] Further Extensions & Applications

Applications: Conditional Image Diffusion

Palette: Image-to-Image Diffusion Models

Chitwan Saharia
Google Research, Brain Team
Toronto, ON, Canada
sahariac@google.com

William Chan
Google Research, Brain Team
Toronto, ON, Canada
williamchan@google.com

Huiwen Chang
Google Research
New York, NY, USA
huiwenchang@google.com

Chris A. Lee
Google Research
Mountain View, CA, USA
chrisalee@google.com

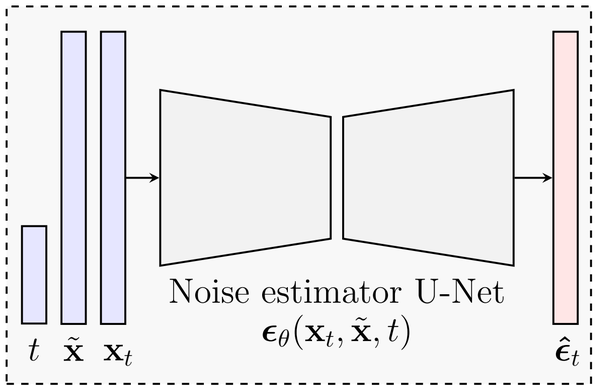
Jonathan Ho
Google Research, Brain Team
New York, NY, USA
jonathanho@google.com

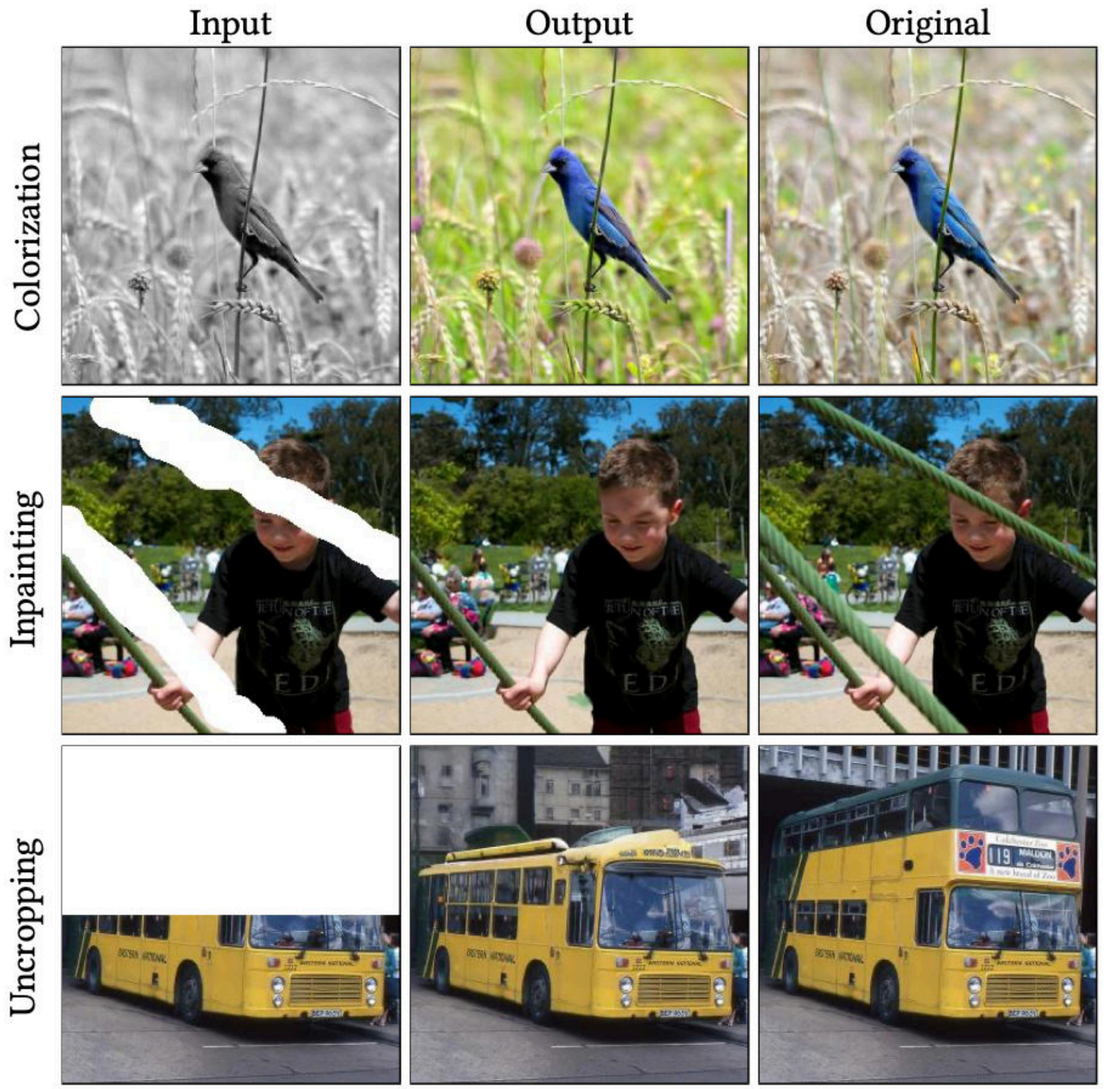
Tim Salimans
Google Research, Brain Team
Amsterdam, Netherlands
salimans@google.com

David Fleet
Google Research, Brain Team
Toronto, ON, Canada
davidfleet@google.com

Mohammad Norouzi
Google Research, Brain Team
Toronto, ON, Canada
mnorouzi@google.com

Training a conditional diffusion model by concatenating a conditioning image at the input.





Applications: Conditional Image Diffusion

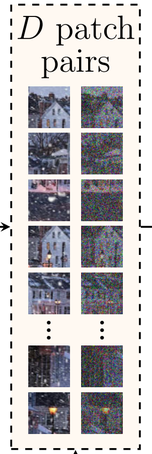
Restoring Vision in Adverse Weather Conditions with Patch-Based Denoising Diffusion Models

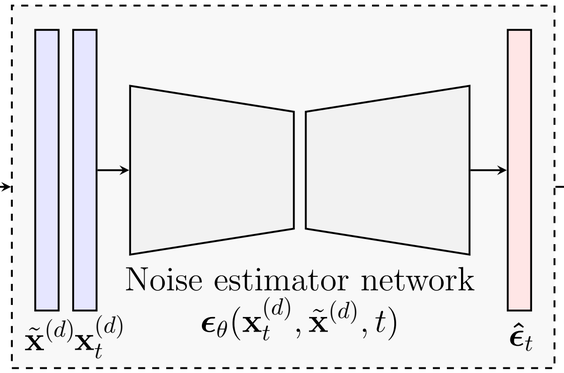
Ozan Özdenizci and Robert Legenstein

Abstract—Image restoration under adverse weather conditions has been of significant interest for various computer vision applications. Recent successful methods rely on the current progress in deep neural network architectural designs (e.g., with vision transformers). Motivated by the recent progress achieved with state-of-the-art conditional generative models, we present a novel patch-based image restoration algorithm based on denoising diffusion probabilistic models. Our patch-based diffusion modeling approach enables size-agnostic image restoration by using a guided denoising process with smoothed noise estimates across overlapping patches during inference. We empirically evaluate our model on benchmark datasets for image desnowing, combined deraining and dehazing, and raindrop removal. We demonstrate our approach to achieve state-of-the-art performances on both weather-specific and multi-weather image restoration, and qualitatively show strong generalization to real-world test images.

Index Terms—denoising diffusion models, patch-based image restoration, deraining, desnowing, dehazing, raindrop removal.

Weather-degraded observation: $\tilde{\mathbf{X}}$





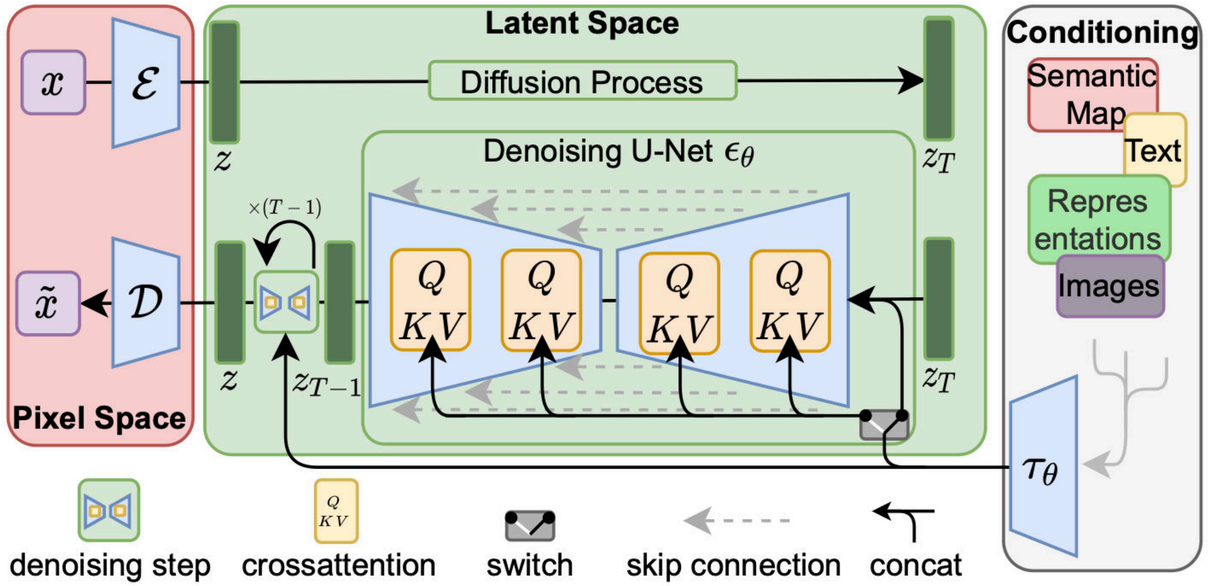
merge patches during sampling

$\mathbf{X}_t$

Applications: Text-to-Image Diffusion Models

High-Resolution Image Synthesis with Latent Diffusion Models

Robin Rombach[1] * Andreas Blattmann[1] * Dominik Lorenz[1] Patrick Esser[R] Björn Ommer[1]
[1]Ludwig Maximilian University of Munich & IWR, Heidelberg University, Germany [R]Runway ML
https://github.com/CompVis/latent-diffusion



"a teddy bear in a suit riding a bike on the streets of Graz" by Stable Diffusion

Robin Rombach et al., "*High-resolution image synthesis with latent diffusion models*", IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022.

More recent applications…

- Improvements in higher resolution image synthesis
- Generating and editing 3D shapes
- Video diffusion models
- Text-to-music diffusion models (e.g., Riffusion, Moûsai, …)
- …

SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis

Dustin Podell **Zion English** **Kyle Lacey** **Andreas Blattmann** **Tim Dockhorn**

Jonas Müller **Joe Penna** **Robin Rombach**

Stability AI, Applied Research

3D Shape Generation and Completion through Point-Voxel Diffusion

Linqi Zhou
Stanford University

Yilun Du
MIT

Jiajun Wu
Stanford University

Stable Video Diffusion: Scaling Latent Video Diffusion Models to Large Datasets

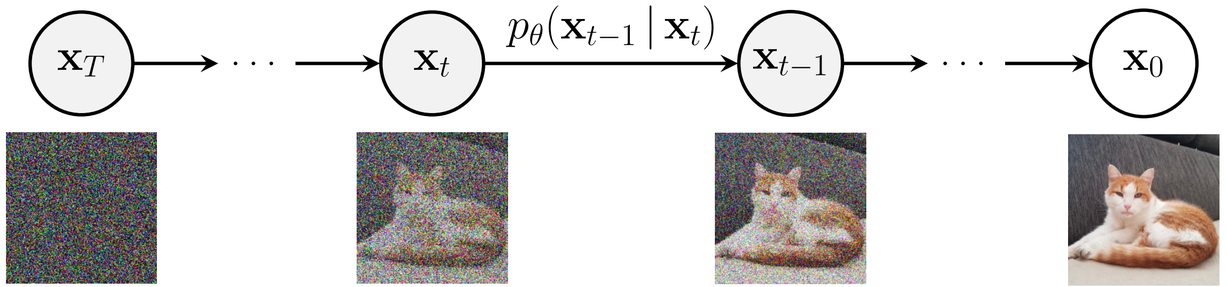
Andreas Blattmann* Tim Dockhorn* Sumith Kulal* Daniel Mendelevitch
Maciej Kilian Dominik Lorenz Yam Levi Zion English Vikram Voleti
Adam Letts Varun Jampani Robin Rombach
Stability AI

"A robot dj is playing the turntables, in heavy raining futuristic tokyo, rooftop, sci-fi, fantasy"

Summary

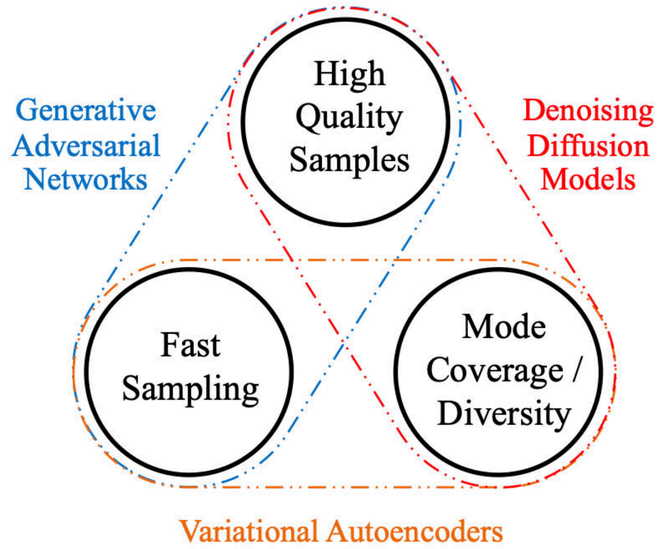
- Iteratively **denoising** an image with a neural network by inverting a **diffusion** process which destroys the data structure by gradually adding noise.
- Denoising diffusion models are easy to train: $\mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon}-\boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0+\sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon},t)\right\|^2\right]$
- Image-to-image (U-Net) or vector-to-vector (latent-space) network architectures can be used.
- Sampling from a diffusion model is relatively slow.

  (Note: recent accelerated sampling techniques exist.)



- State-of-the-art: More powerful than existing generative models.

The Generative Learning Trilemma



[Xiao et al., ICLR 2022]

Today

- ☑ Deep Generative Models
  - ☑ Variational Autoencoders
  - ☑ Generative Adversarial Networks
  - ☑ Denoising Diffusion Probabilistic Models
    - ☑ Idea & Training Objective
    - ☑ Sampling
    - ☑ Further Extensions & Applications



"audience raising hands for questions during a lecture" by Stable Diffusion

Questions?