

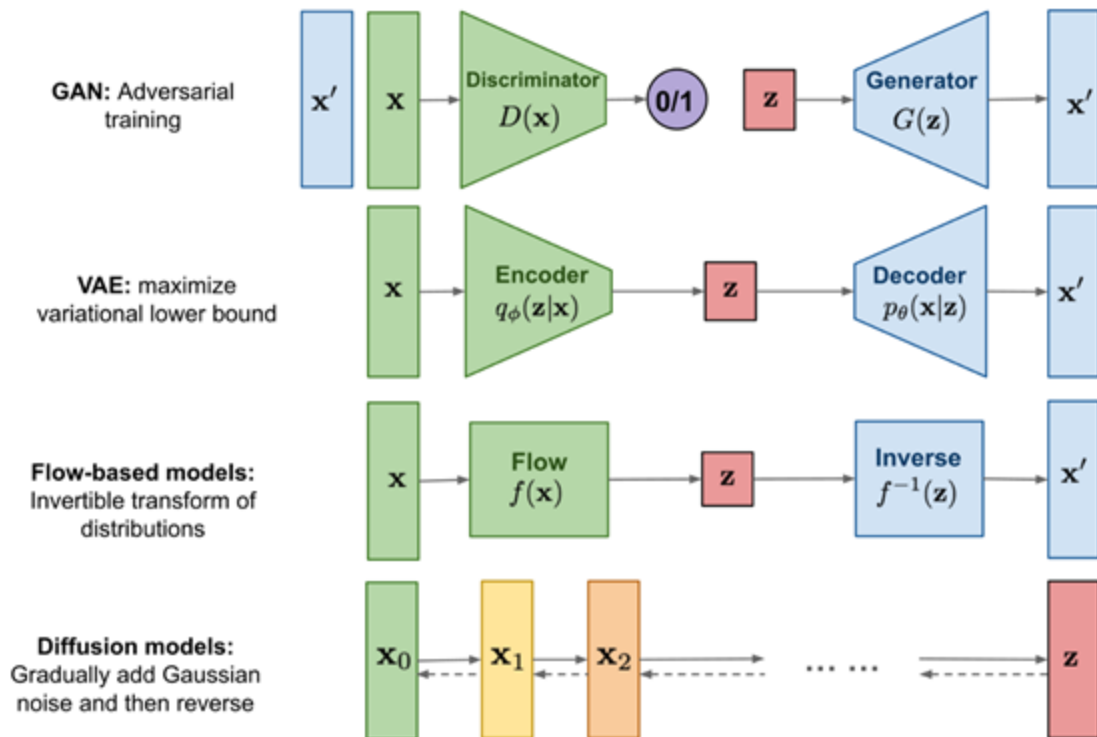
Diffusion Models for Audio Generation

Aibek Alanov,
PhD in Computer Science,
Head of Controllable GenAI at AIRI,
Head of BayesGroup at HSE University

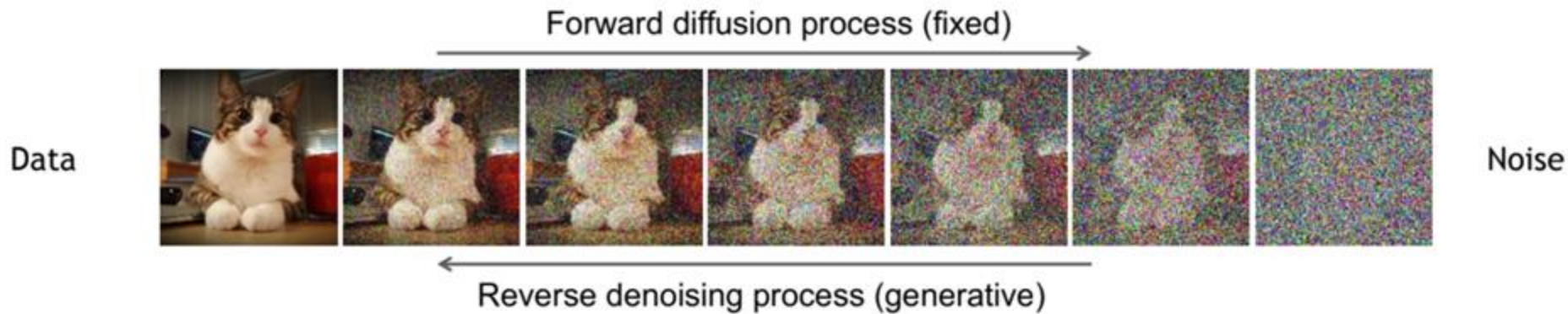
Agenda

- Introduction to Diffusion Models: A Multi-Perspective Approach:
 - Score Matching Perspective
 - Stochastic Differential Equations (SDE) Perspective
 - Latent Probabilistic Models Perspective
- Applications of Diffusion Models in Audio
 - Diffusion Models for Vocoding
 - Diffusion Models for Text-to-Speech (TTS)

Generative models

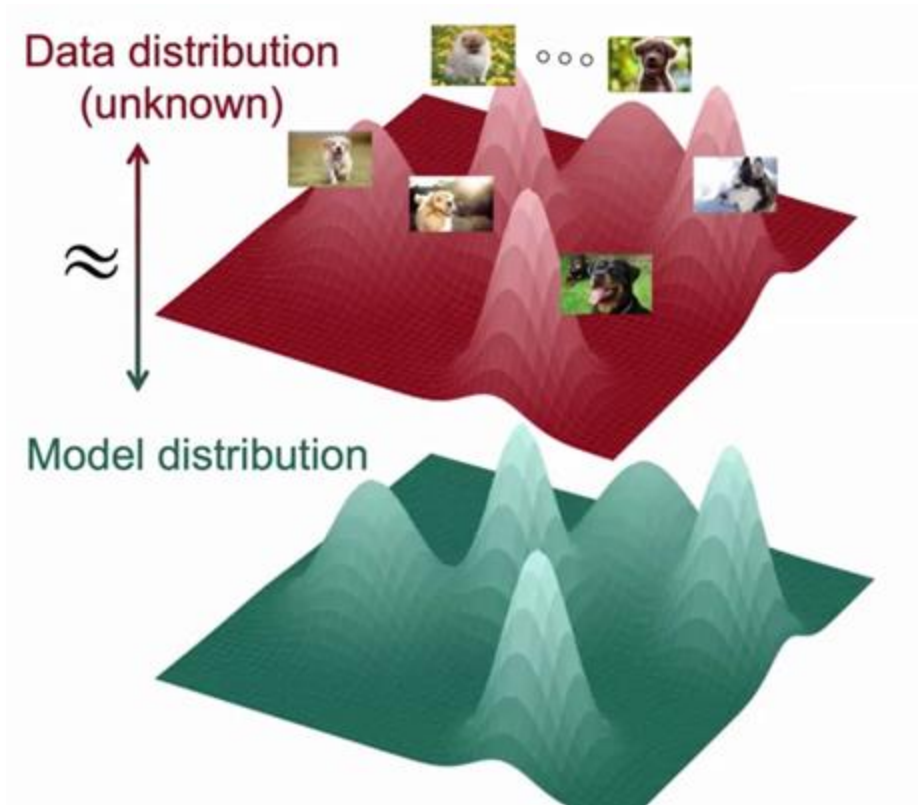


Diffusion models

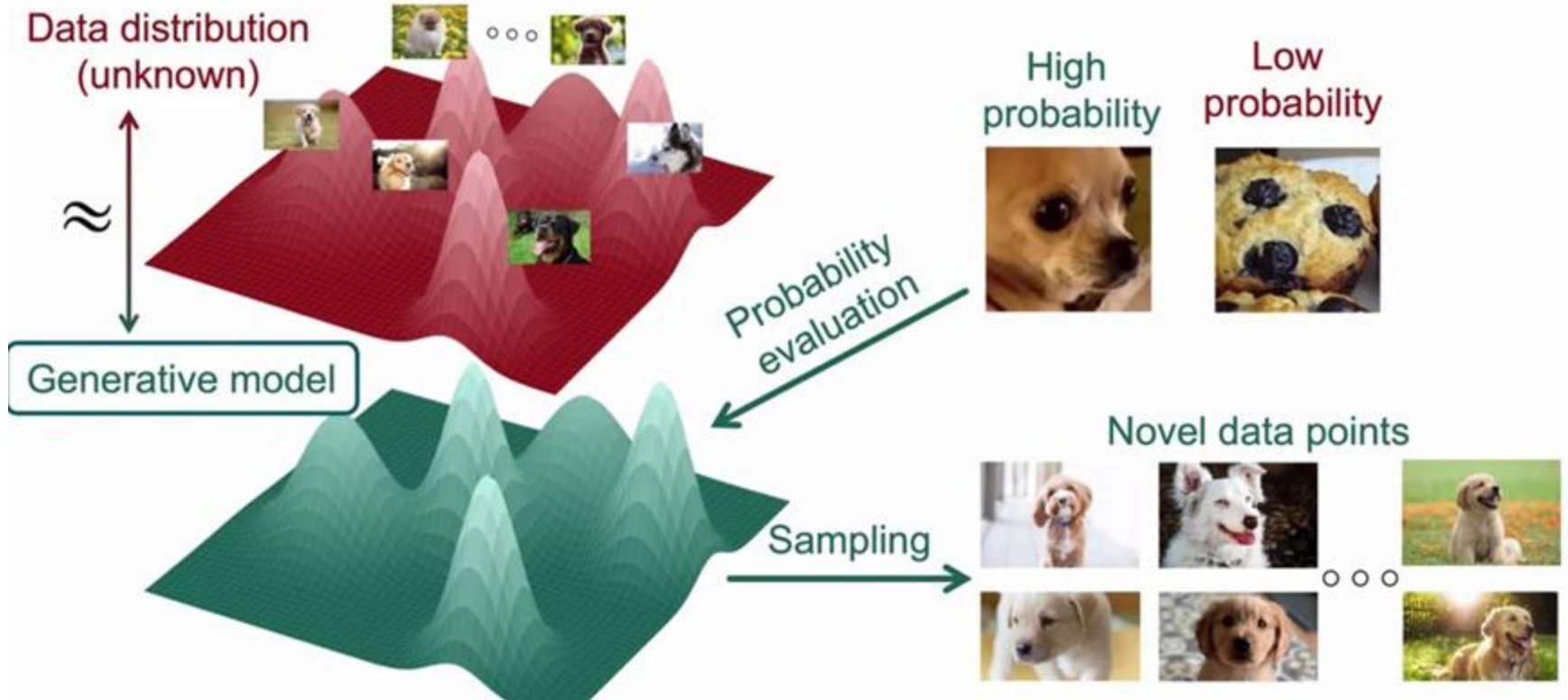


Score Matching Perspective

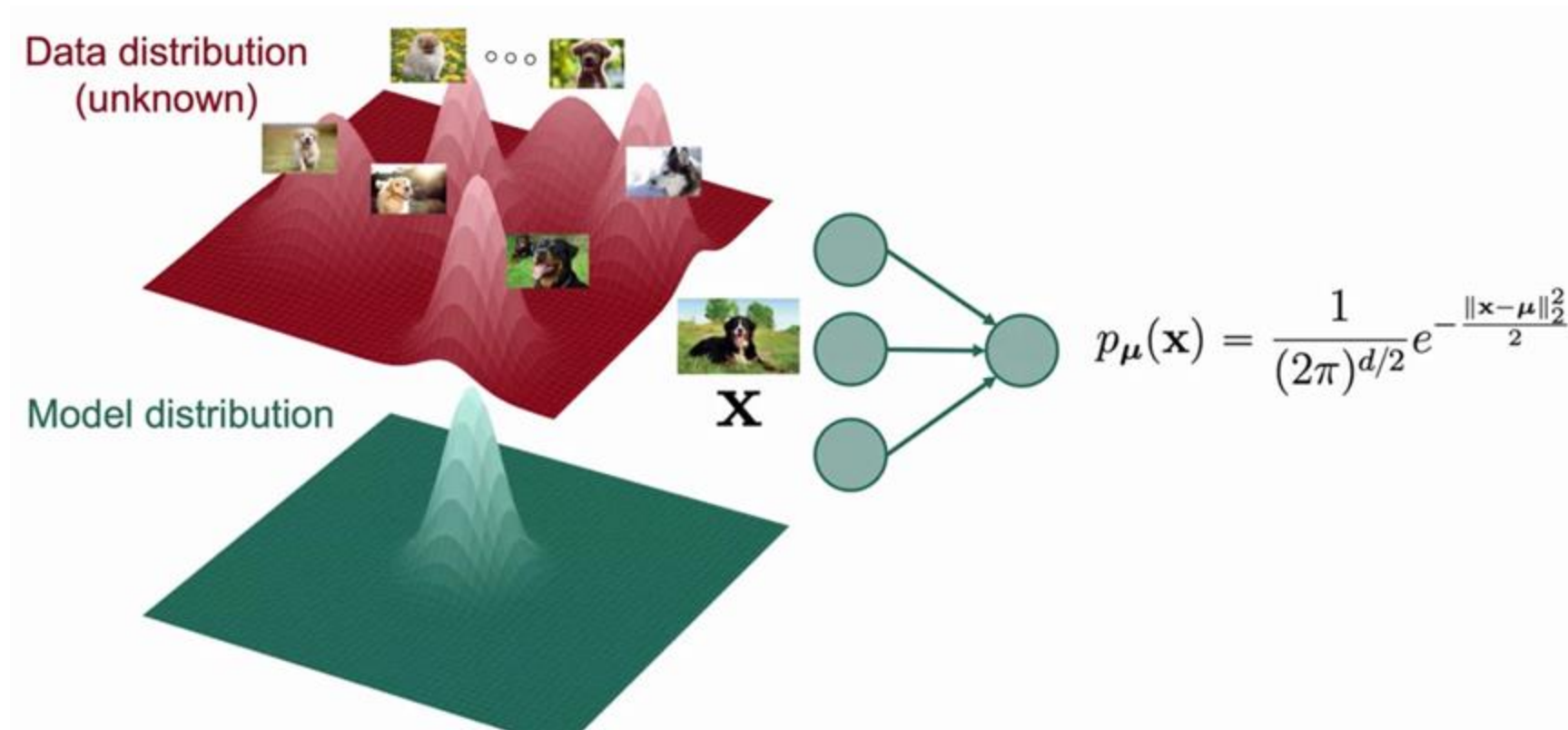
Estimating the probability distribution of data



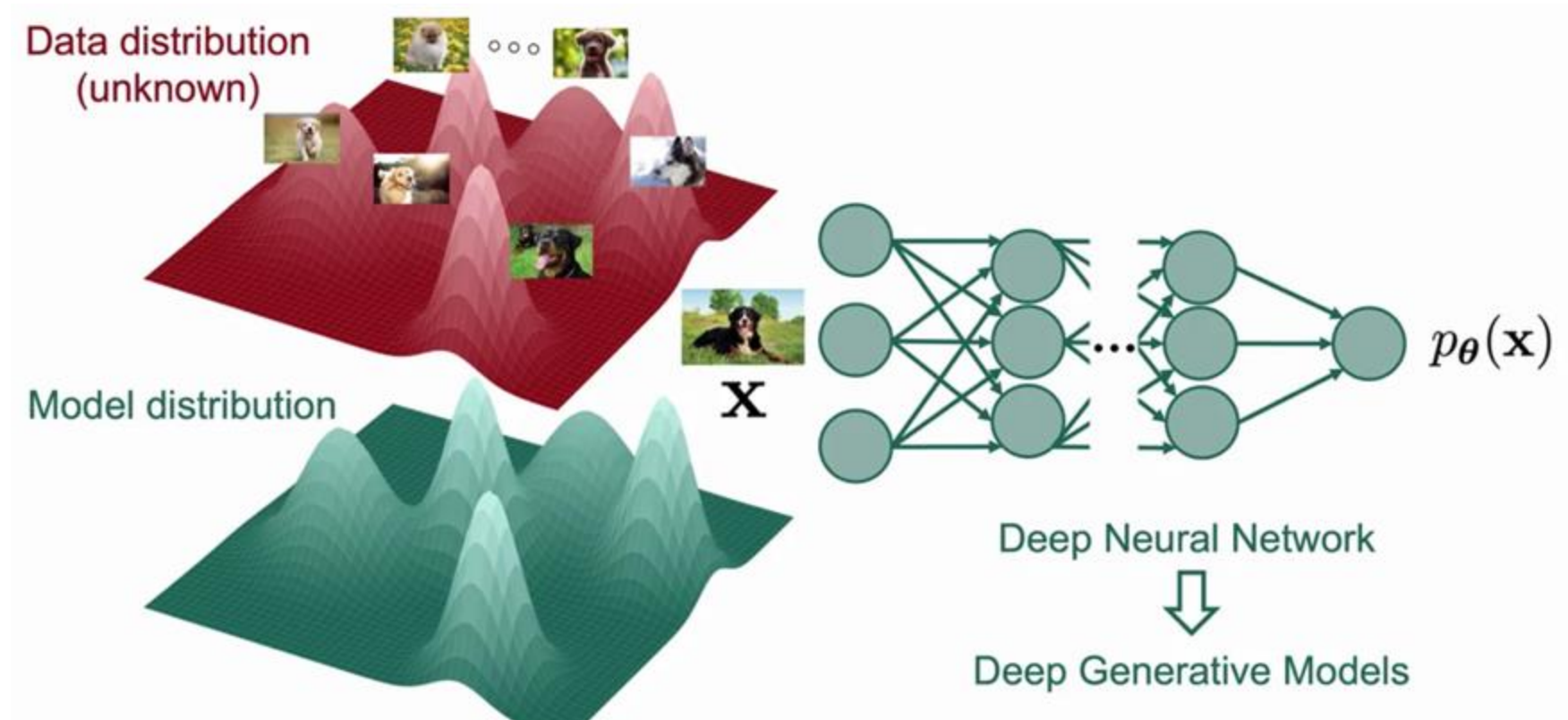
Estimating the probability distribution of data



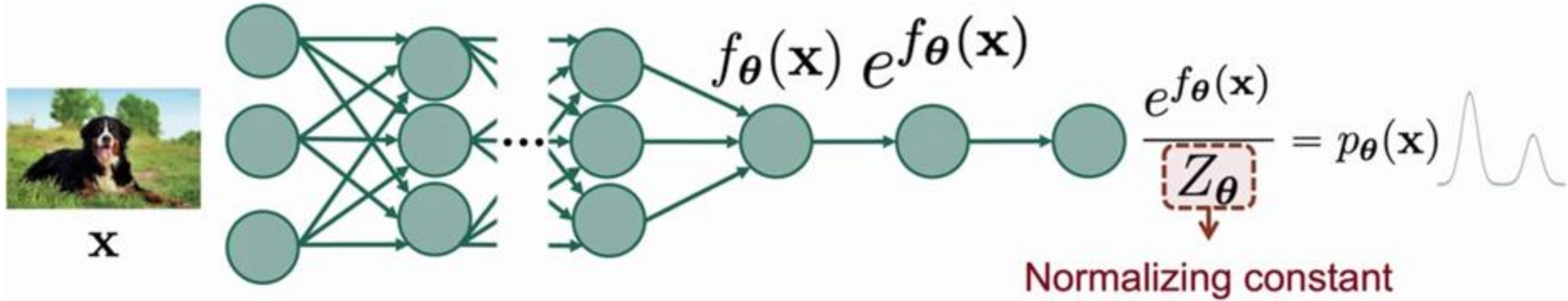
Estimating the probability distribution of data



Estimating the probability distribution of data



Estimating the probability distribution of data



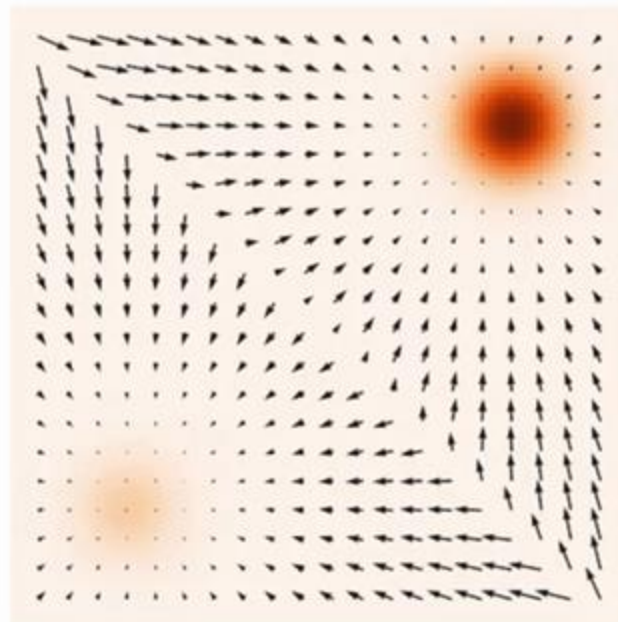
$$Z_{\theta} = \int e^{f_{\theta}(\mathbf{x})} d\mathbf{x}$$

Estimating the score function of data

$p(\mathbf{x})$
Probability density function



$\nabla_{\mathbf{x}} \log p(\mathbf{x})$
(Stein) score function



Score vs. density function

Estimating the score function of data



Training data

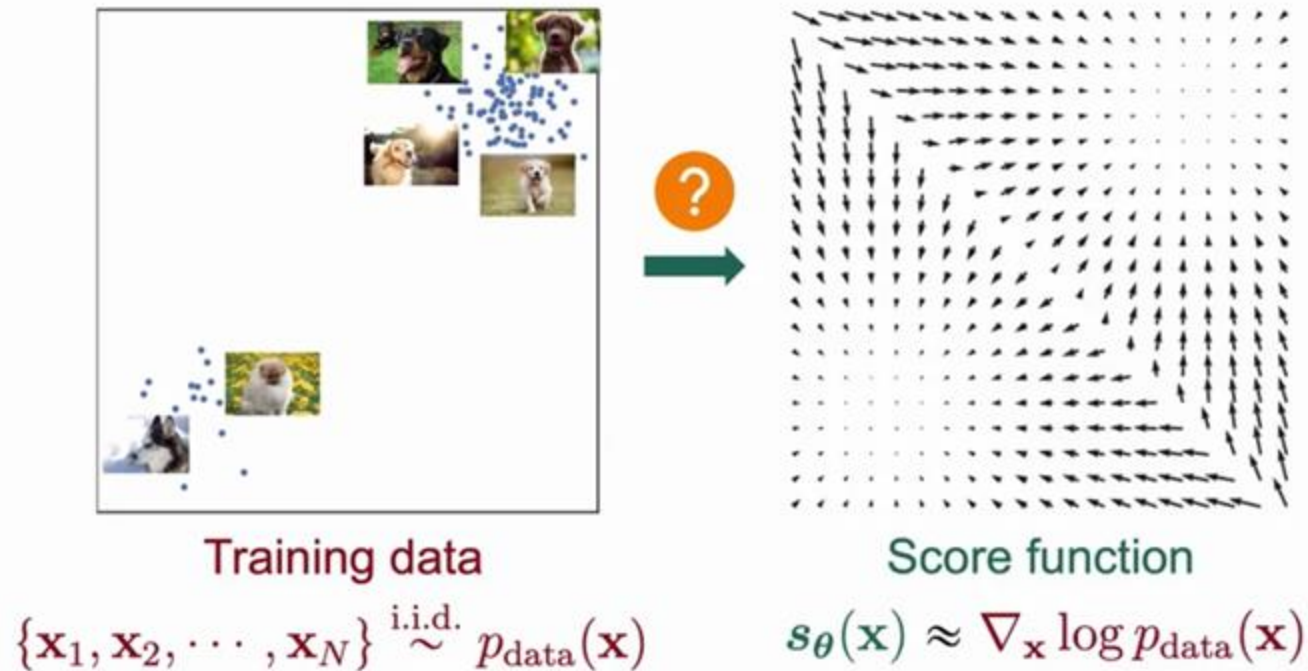
$$\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\} \stackrel{\text{i.i.d.}}{\sim} p_{\text{data}}(\mathbf{x})$$



Probability density function

$$p_{\theta}(\mathbf{x}) \approx p_{\text{data}}(\mathbf{x})$$

Estimating the score function of data



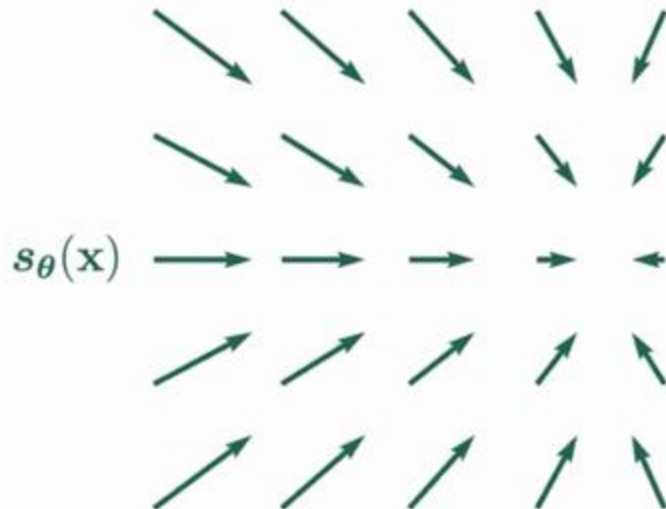
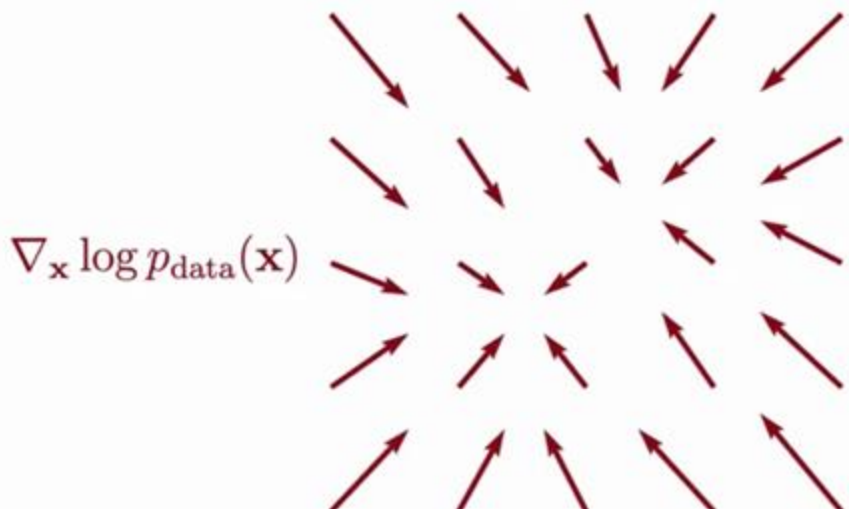
Estimating the score function of data

Given: $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\} \stackrel{\text{i.i.d.}}{\sim} p_{\text{data}}(\mathbf{x})$

Goal: $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$

Score Model: $s_{\theta}(\mathbf{x}) : \mathbb{R}^d \rightarrow \mathbb{R}^d \approx \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$

Objective: How to compare two vector fields of scores?



Estimating the score function of data

Given: $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\} \stackrel{\text{i.i.d.}}{\sim} p_{\text{data}}(\mathbf{x})$

Goal: $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$

Score Model: $\mathbf{s}_{\theta}(\mathbf{x}) : \mathbb{R}^d \rightarrow \mathbb{R}^d \approx \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$

Objective: How to compare two vector fields of scores?

$$\frac{1}{2} \mathbb{E}_{p_{\text{data}}(\mathbf{x})} [\|\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - \mathbf{s}_{\theta}(\mathbf{x})\|_2^2]$$

(Fisher divergence)

Estimating the score function of data

$$\begin{aligned} & \frac{1}{2} \mathbb{E}_{p_{\text{data}}} [(\nabla_x \log p_{\text{data}}(x) - \nabla_x \log p_{\theta}(x))^2] \\ &= \frac{1}{2} \int p_{\text{data}}(x) (\nabla_x \log p_{\text{data}}(x) - \nabla_x \log p_{\theta}(x))^2 dx \\ &= \underbrace{\frac{1}{2} \int p_{\text{data}}(x) (\nabla_x \log p_{\text{data}}(x))^2 dx}_{\text{const}} + \frac{1}{2} \int p_{\text{data}}(x) (\nabla_x \log p_{\theta}(x))^2 dx \\ &\quad - \int p_{\text{data}}(x) \nabla_x \log p_{\theta}(x) \nabla_x \log p_{\text{data}}(x) dx. \end{aligned}$$

Estimating the score function of data

$$\begin{aligned}& - \int p_{\text{data}}(x) \nabla_x \log p_{\theta}(x) \nabla_x \log p_{\text{data}}(x) dx \\&= - \int \nabla_x \log p_{\theta}(x) \nabla_x p_{\text{data}}(x) dx \\&= - p_{\text{data}}(x) \nabla_x \log p_{\theta}(x) \Big|_{-\infty}^{\infty} + \int p_{\text{data}}(x) \nabla_x^2 \log p_{\theta}(x) dx \\&\stackrel{(i)}{=} \mathbb{E}_{p_{\text{data}}} [\nabla_x^2 \log p_{\theta}(x)],\end{aligned}$$

Estimating the score function of data

Given: $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\} \stackrel{\text{i.i.d.}}{\sim} p_{\text{data}}(\mathbf{x})$

Goal: $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$

Score Model: $\mathbf{s}_{\theta}(\mathbf{x}) : \mathbb{R}^d \rightarrow \mathbb{R}^d \approx \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$

Objective: How to compare two vector fields of scores?

$$\frac{1}{2} \mathbb{E}_{p_{\text{data}}(\mathbf{x})} [\|\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - \mathbf{s}_{\theta}(\mathbf{x})\|_2^2]$$

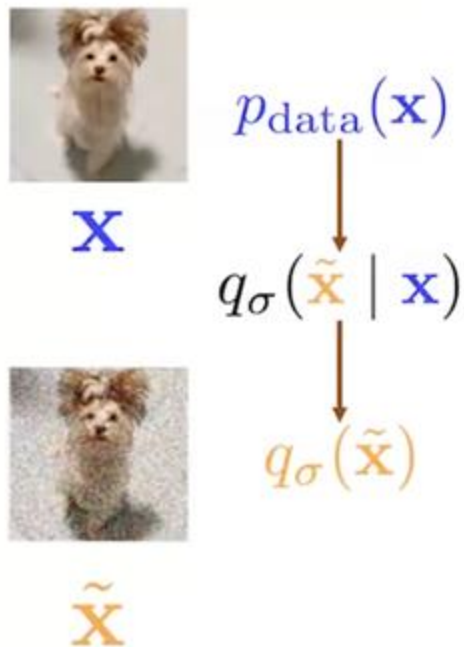
(Fisher divergence)

Score Matching [Hyvärinen 2005]:

$$\mathbb{E}_{p_{\text{data}}(\mathbf{x})} \left[\frac{1}{2} \|\mathbf{s}_{\theta}(\mathbf{x})\|_2^2 + \text{trace} \left(\underbrace{\nabla_{\mathbf{x}} \mathbf{s}_{\theta}(\mathbf{x})}_{\text{Jacobian of } \mathbf{s}_{\theta}(\mathbf{x})} \right) \right]$$

Integration by parts
(Gauss's theorem)

Denoising score matching



- Matching the score of a noise-perturbed distribution

$$\frac{1}{2} \mathbb{E}_{q_{\sigma}(\tilde{\mathbf{x}})} [\|\nabla_{\tilde{\mathbf{x}}} \log q_{\sigma}(\tilde{\mathbf{x}}) - \mathbf{s}_{\theta}(\tilde{\mathbf{x}})\|_2^2]$$

- Denoising score matching (Vincent 2011)

$$\frac{1}{2} \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{q_{\sigma}(\tilde{\mathbf{x}}|\mathbf{x})} [\|\nabla_{\tilde{\mathbf{x}}} \log q_{\sigma}(\tilde{\mathbf{x}} | \mathbf{x}) - \mathbf{s}_{\theta}(\tilde{\mathbf{x}})\|_2^2]$$

↓
Scalable!

Denoising score matching

$$q_{\theta}(\tilde{x}) = \int q_{\theta}(\tilde{x} | x) p_{\text{data}}(x) dx$$

$$E_{q_{\theta}(\tilde{x})} \| \nabla_{\tilde{x}} \log q_{\theta}(\tilde{x}) - s_{\theta}(\tilde{x}) \|_2^2$$

Denoising score matching

$$\begin{aligned} & \int q_{\theta}(\tilde{x}) \left(\nabla_{\tilde{x}} \log q_{\theta}(\tilde{x}) - s_{\theta}(\tilde{x}) \right)^2 d\tilde{x} = \\ &= \int q_{\theta}(\tilde{x}) s_{\theta}^2(\tilde{x}) d\tilde{x} - 2 \int q_{\theta}(\tilde{x}) s_{\theta}(\tilde{x}) \nabla_{\tilde{x}} \log q_{\theta}(\tilde{x}) d\tilde{x} + \\ &+ \underbrace{\int q_{\theta}(\tilde{x}) \left(\nabla_{\tilde{x}} \log q_{\theta}(\tilde{x}) \right)^2 d\tilde{x}}_{\text{const}} \end{aligned}$$

Denoising score matching

$$\begin{aligned}\int q_{\theta}(\hat{x}) s_{\theta}^2(\hat{x}) d\hat{x} &= \int \int p_{\text{data}}(x) q_{\theta}(\hat{x}|x) dx s_{\theta}^2(\hat{x}) d\hat{x} \\&= \int p_{\text{data}}(x) \left(\int s_{\theta}^2(\hat{x}) q_{\theta}(\hat{x}|x) d\hat{x} \right) dx = \\&= E_{p_{\text{data}}(x)} E_{q_{\theta}(\hat{x}|x)} s_{\theta}^2(\hat{x})\end{aligned}$$

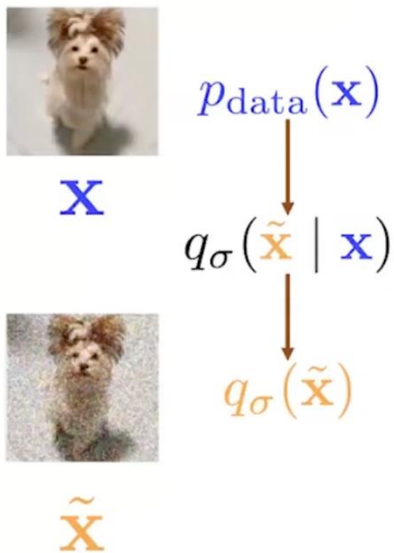
Denoising score matching

$$\begin{aligned}
 \int q_{\sigma}(\tilde{x}) s_{\theta}(\tilde{x}) \nabla_{\tilde{x}} \log q_{\sigma}(\tilde{x}) d\tilde{x} &= \int \nabla_{\tilde{x}} \log q_{\sigma}(\tilde{x}) = \frac{\nabla q_{\sigma}(\tilde{x})}{q_{\sigma}(\tilde{x})} d\tilde{x} = \\
 &= \int s_{\theta}(\tilde{x}) \nabla_{\tilde{x}} q_{\sigma}(\tilde{x}) d\tilde{x} = \int s_{\theta}(\tilde{x}) \nabla_{\tilde{x}} \left(\int p_{data}(x) q_{\sigma}(\tilde{x}|x) dx \right) d\tilde{x} = \\
 &= \int s_{\theta}(\tilde{x}) \int p_{data}(x) \nabla_{\tilde{x}} q_{\sigma}(\tilde{x}|x) dx d\tilde{x} = \\
 &= \int p_{data}(x) \int s_{\theta}(\tilde{x}) \nabla_{\tilde{x}} q_{\sigma}(\tilde{x}|x) d\tilde{x} dx =
 \end{aligned}$$

Denoising score matching

$$\begin{aligned} &= \int p_{data}(x) \int q_{\sigma}(\hat{x}|x) s_{\theta}(\hat{x}) \nabla_{\hat{x}} \log q_{\sigma}(\hat{x}|x) d\hat{x} dx = \\ &= E_{p_{data}(x)} E_{q_{\sigma}(\hat{x}|x)} \left[s_{\theta}(\hat{x}) \nabla_{\hat{x}} \log q_{\sigma}(\hat{x}|x) \right] \end{aligned}$$

Denoising score matching



- Matching the score of a noise-perturbed distribution

$$\frac{1}{2} \mathbb{E}_{q_{\sigma}(\tilde{\mathbf{x}})} [\|\nabla_{\tilde{\mathbf{x}}} \log q_{\sigma}(\tilde{\mathbf{x}}) - \mathbf{s}_{\theta}(\tilde{\mathbf{x}})\|_2^2]$$

- Denoising score matching (Vincent 2011)

$$\frac{1}{2} \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{q_{\sigma}(\tilde{\mathbf{x}}|\mathbf{x})} [\|\nabla_{\tilde{\mathbf{x}}} \log q_{\sigma}(\tilde{\mathbf{x}} | \mathbf{x}) - \mathbf{s}_{\theta}(\tilde{\mathbf{x}})\|_2^2]$$

Scalable!

Denoising score matching

$$\begin{aligned}\nabla_{\tilde{\mathbf{x}}} \log q_{\sigma}(\tilde{\mathbf{x}}|\mathbf{x}) &= \nabla_{\tilde{\mathbf{x}}} \log \mathcal{N}(\mathbf{x}, \sigma^2 \cdot \mathbf{I}) \\&= \nabla_{\tilde{\mathbf{x}}} \log \frac{\exp(-\frac{1}{2}(\tilde{\mathbf{x}} - \mathbf{x})^T \cdot (\sigma^2 \cdot \mathbf{I})^{-1} \cdot (\tilde{\mathbf{x}} - \mathbf{x}))}{\sqrt{(2\pi)^d |\sigma^2 \cdot \mathbf{I}|}} \\&= \nabla_{\tilde{\mathbf{x}}} \log \exp(-\frac{1}{2}(\tilde{\mathbf{x}} - \mathbf{x})^T \cdot (\sigma^2 \cdot \mathbf{I})^{-1} \cdot (\tilde{\mathbf{x}} - \mathbf{x})) \\&\quad - \underbrace{\nabla_{\tilde{\mathbf{x}}} \log \sqrt{(2\pi)^d |\sigma^2 \cdot \mathbf{I}|}}_{=0} \\&= -\frac{1}{2\sigma^2} \nabla_{\tilde{\mathbf{x}}} (\tilde{\mathbf{x}} - \mathbf{x})^T \cdot \mathbf{I} \cdot (\tilde{\mathbf{x}} - \mathbf{x}) \\&= -\frac{1}{2\sigma^2} \nabla_{\tilde{\mathbf{x}}} (\tilde{\mathbf{x}} - \mathbf{x})^2 \\&= -\frac{(\tilde{\mathbf{x}} - \mathbf{x})}{\sigma^2} = \frac{(\mathbf{x} - \tilde{\mathbf{x}})}{\sigma^2}.\end{aligned}$$

Denoising score matching

- Sample a minibatch of datapoints $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\} \sim p_{\text{data}}(\mathbf{x})$
- Sample a minibatch of perturbed datapoints $\{\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \dots, \tilde{\mathbf{x}}_n\} \sim q_{\sigma}(\tilde{\mathbf{x}})$
 $\tilde{\mathbf{x}}_i \sim q_{\sigma}(\tilde{\mathbf{x}}_i \mid \mathbf{x}_i)$

- Estimate the denoising score matching loss with empirical means

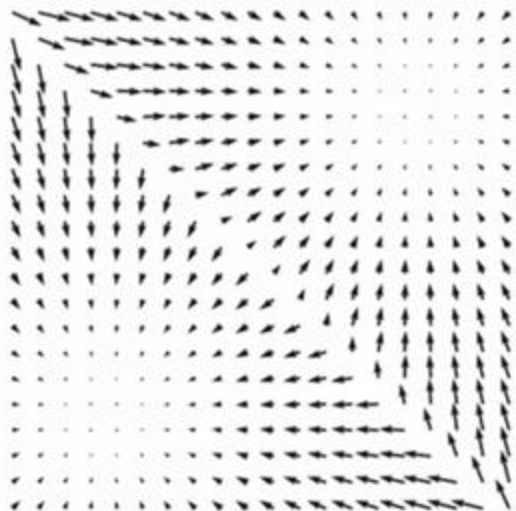
$$\frac{1}{2n} \sum_{i=1}^n [\|s_{\theta}(\tilde{\mathbf{x}}_i) - \nabla_{\tilde{\mathbf{x}}} \log q_{\sigma}(\tilde{\mathbf{x}}_i \mid \mathbf{x}_i)\|_2^2]$$

- If Gaussian perturbation

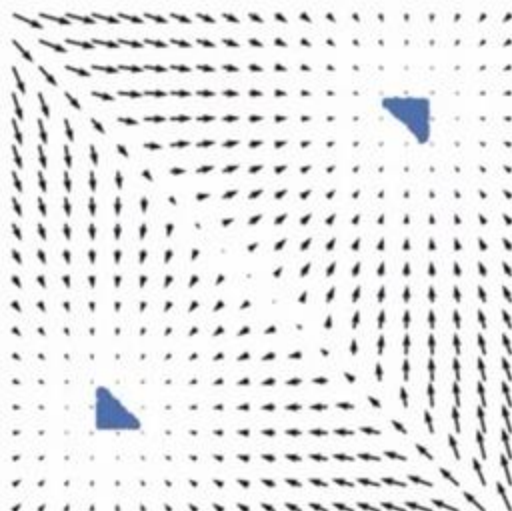
$$\frac{1}{2n} \sum_{i=1}^n \left[\left\| s_{\theta}(\tilde{\mathbf{x}}_i) + \frac{\tilde{\mathbf{x}}_i - \mathbf{x}_i}{\sigma^2} \right\|_2^2 \right]$$

- Stochastic gradient descent
- Need to choose a very small σ !

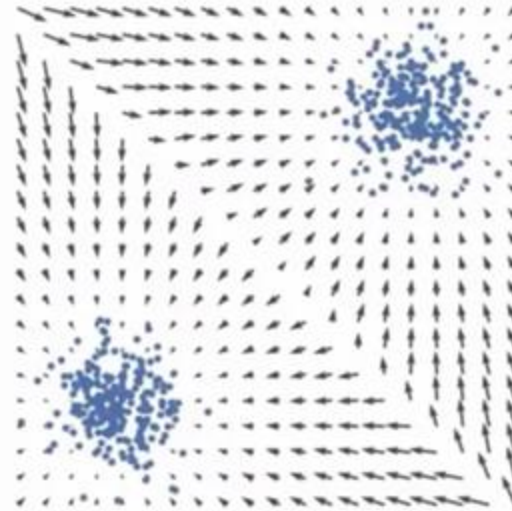
Sampling from score function



Score function



Follow the scores

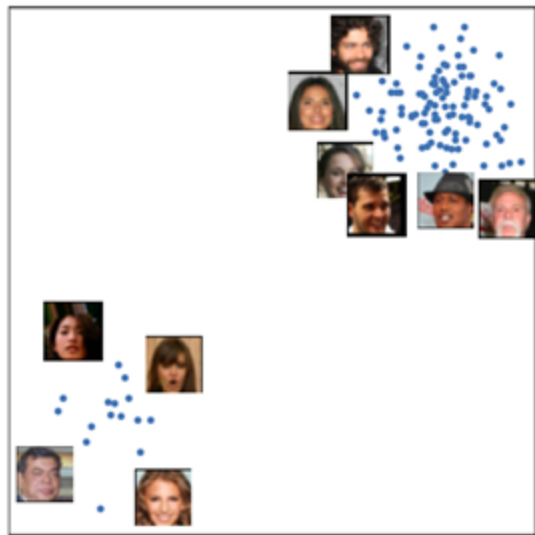


Follow the noisy scores

$$\mathbf{x}_{i+1} \leftarrow \mathbf{x}_i + \epsilon \nabla_{\mathbf{x}} \log p(\mathbf{x}) \quad \mathbf{x}_{i+1} \leftarrow \mathbf{x}_i + \epsilon \nabla_{\mathbf{x}} \log p(\mathbf{x}) + \sqrt{2\epsilon} \mathbf{z}_i$$

Langevin dynamics

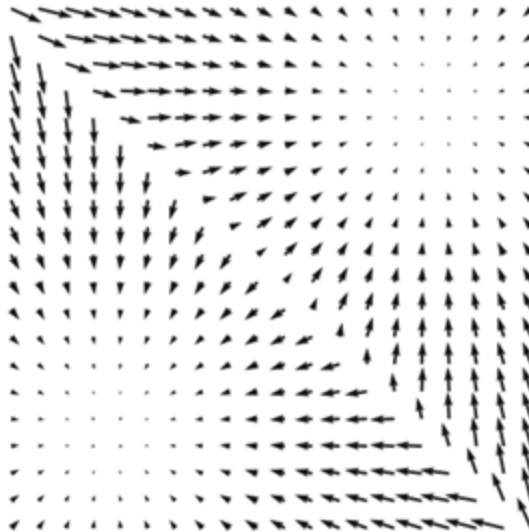
Sampling from score function



Data samples

$$\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\} \stackrel{\text{i.i.d.}}{\sim} p(\mathbf{x})$$

score
matching



Scores

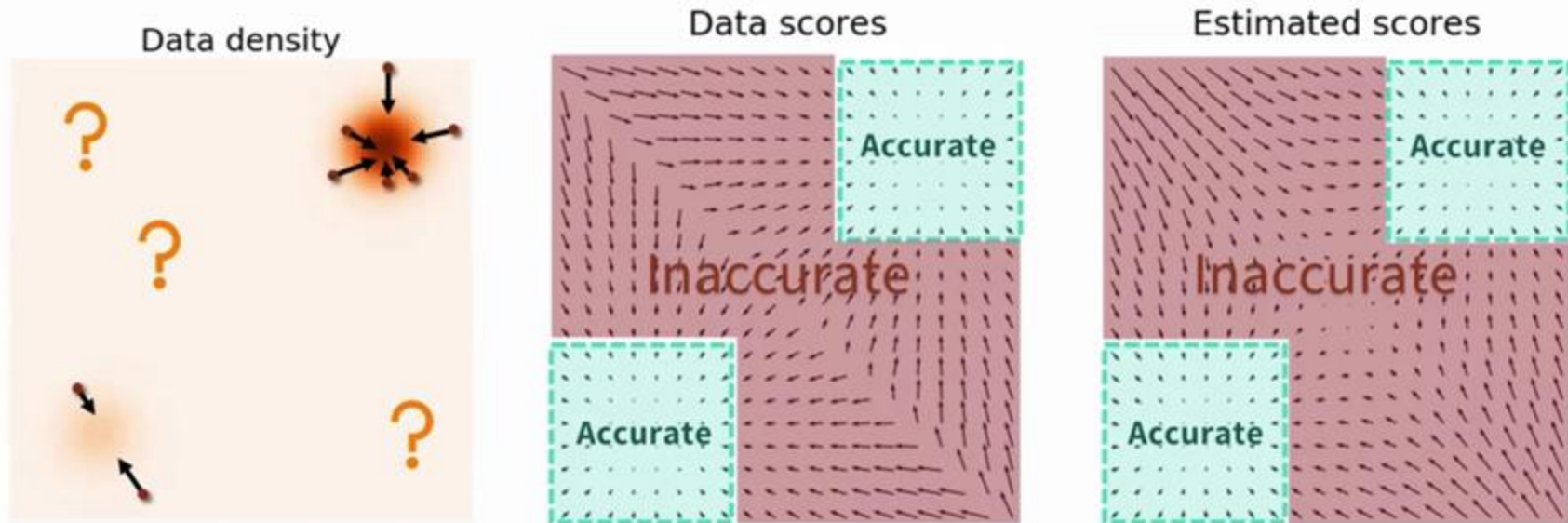
$$\mathbf{s}_\theta(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log p(\mathbf{x})$$

Langevin
dynamics



New samples

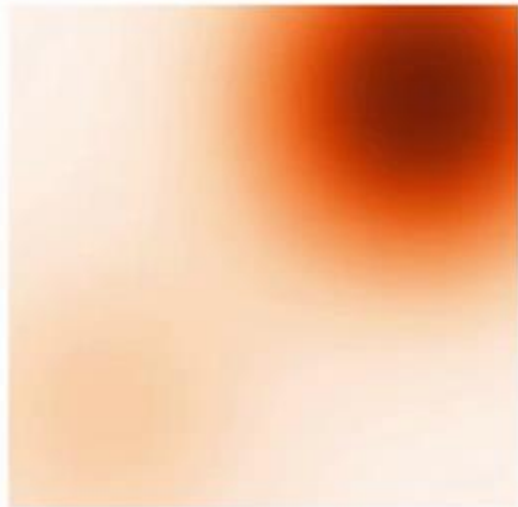
Challenge in low data regions



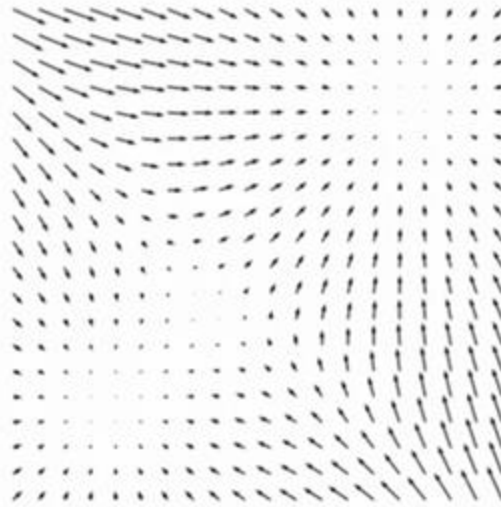
$$\frac{1}{2} \mathbb{E}_{p_{\text{data}}(\mathbf{x})} [\|\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - \mathbf{s}_{\theta}(\mathbf{x})\|_2^2]$$

Challenge in low data regions

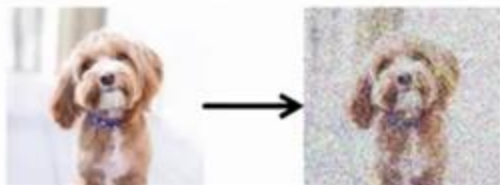
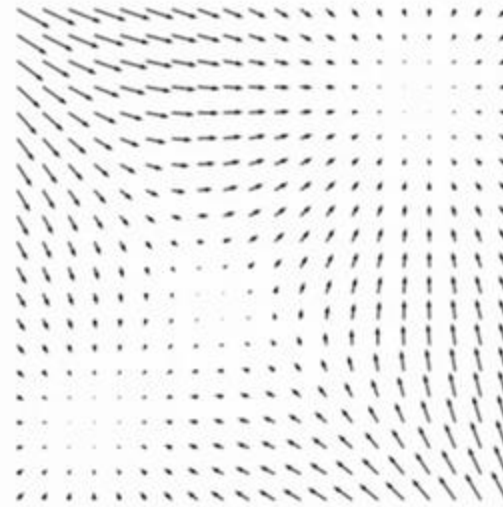
Perturbed density



Perturbed scores



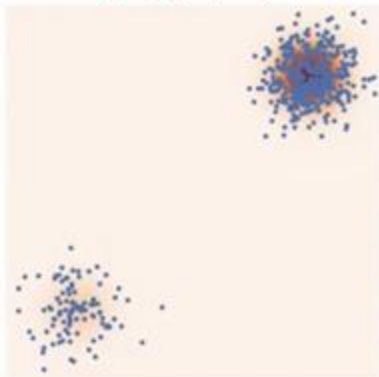
Estimated scores



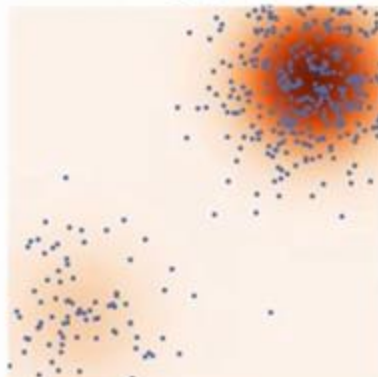
Using multiple noise levels

Data

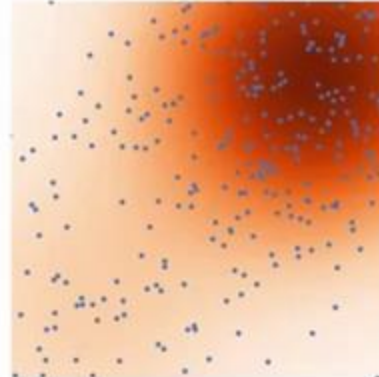
$$p_{\sigma_1}(\mathbf{x})$$



$$p_{\sigma_2}(\mathbf{x})$$



$$p_{\sigma_3}(\mathbf{x})$$



Using multiple noise levels

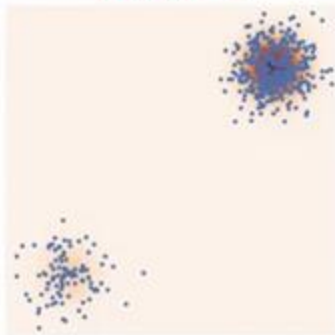
Data



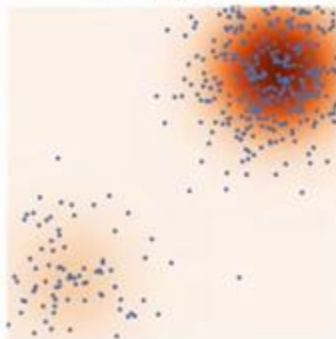
Using multiple noise levels

Data

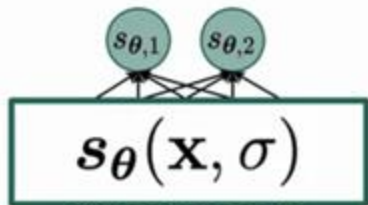
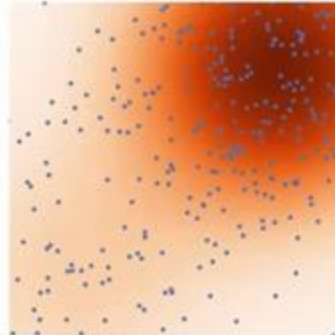
$p_{\sigma_1}(\mathbf{x})$



$p_{\sigma_2}(\mathbf{x})$



$p_{\sigma_3}(\mathbf{x})$



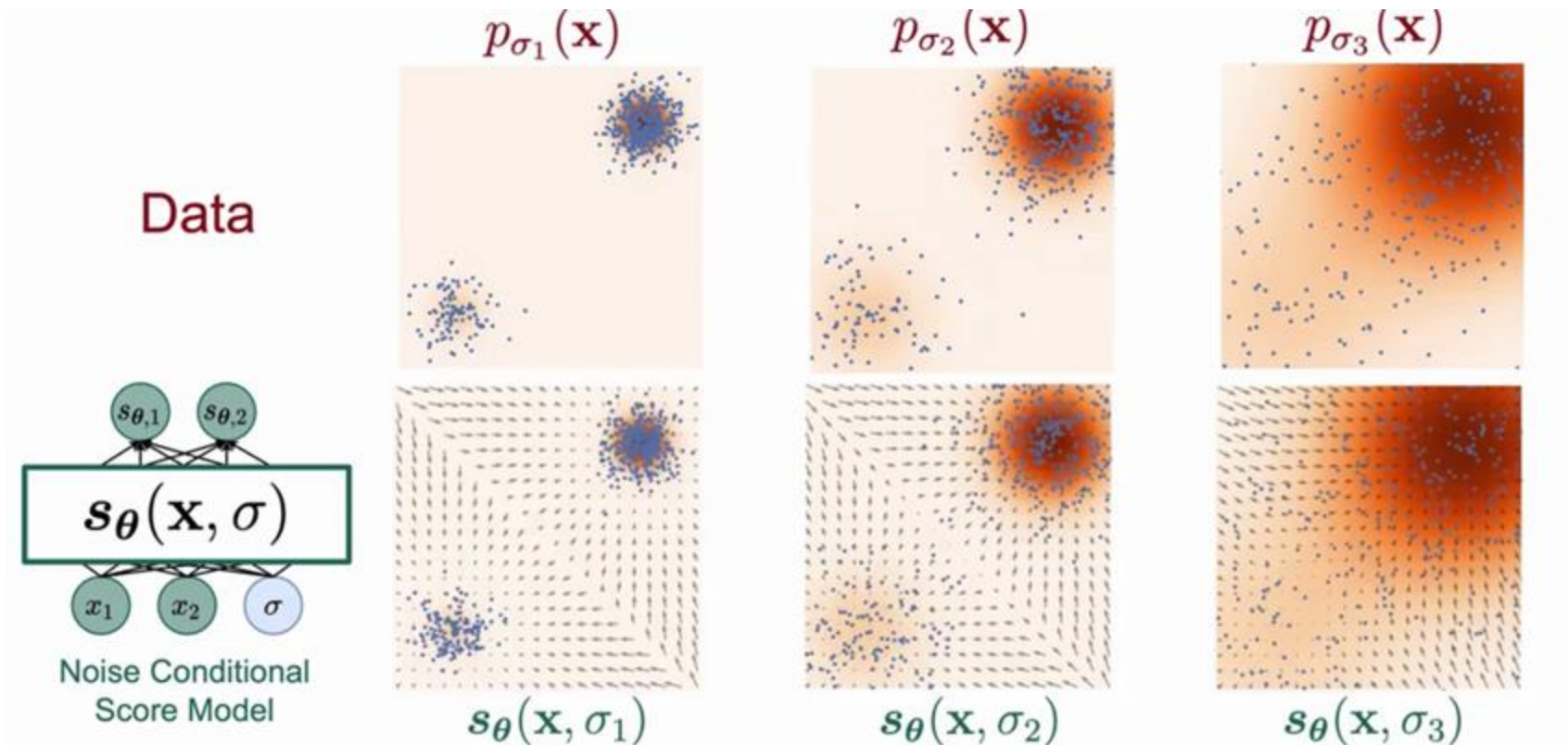
Noise Conditional
Score Model

Positive weighting
function

$$\frac{1}{N} \sum_{i=1}^N \lambda(\sigma_i) \underbrace{\mathbb{E}_{p_{\sigma_i}(\mathbf{x})} [\|\nabla_{\mathbf{x}} \log p_{\sigma_i}(\mathbf{x}) - \mathbf{s}_{\theta}(\mathbf{x}, \sigma_i)\|_2^2]}_{\text{Score matching loss}}$$

Score matching loss

Using multiple noise levels



Learning NCSNs via score matching

$$\ell(\boldsymbol{\theta}; \sigma) \triangleq \frac{1}{2} \mathbb{E}_{p_{\text{data}}(\mathbf{x})} \mathbb{E}_{\tilde{\mathbf{x}} \sim \mathcal{N}(\mathbf{x}, \sigma^2 I)} \left[\left\| \mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}, \sigma) + \frac{\tilde{\mathbf{x}} - \mathbf{x}}{\sigma^2} \right\|_2^2 \right]$$

$$\mathcal{L}(\boldsymbol{\theta}; \{\sigma_i\}_{i=1}^L) \triangleq \frac{1}{L} \sum_{i=1}^L \lambda(\sigma_i) \ell(\boldsymbol{\theta}; \sigma_i)$$

Algorithm 1 Annealed Langevin dynamics.

Require: $\{\sigma_i\}_{i=1}^L, \epsilon, T$.

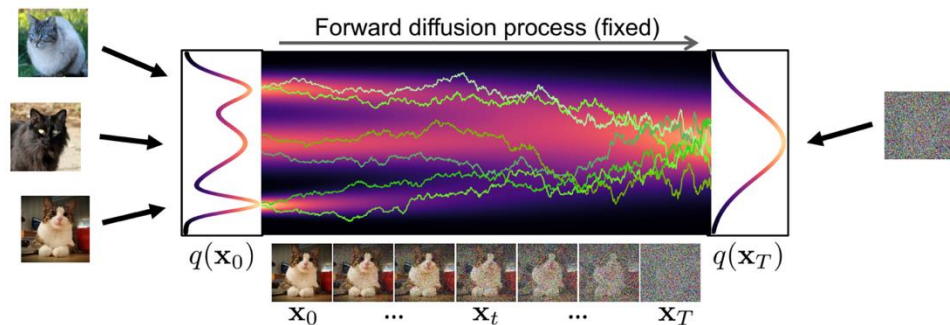
- 1: Initialize $\tilde{\mathbf{x}}_0$
 - 2: **for** $i \leftarrow 1$ to L **do**
 - 3: $\alpha_i \leftarrow \epsilon \cdot \sigma_i^2 / \sigma_L^2$ $\triangleright \alpha_i$ is the step size.
 - 4: **for** $t \leftarrow 1$ to T **do**
 - 5: Draw $\mathbf{z}_t \sim \mathcal{N}(0, I)$
 - 6: $\tilde{\mathbf{x}}_t \leftarrow \tilde{\mathbf{x}}_{t-1} + \frac{\alpha_i}{2} \mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_{t-1}, \sigma_i) + \sqrt{\alpha_i} \mathbf{z}_t$
 - 7: **end for**
 - 8: $\tilde{\mathbf{x}}_0 \leftarrow \tilde{\mathbf{x}}_T$
 - 9: **end for**
 - return** $\tilde{\mathbf{x}}_T$
-

Score matching perspective

- Estimating scores from data instead of density
- Apply score matching techniques
- Utilize denoising score matching
- Use multiple noise levels

SDE Perspective

The Generative Reverse Stochastic Differential Equation



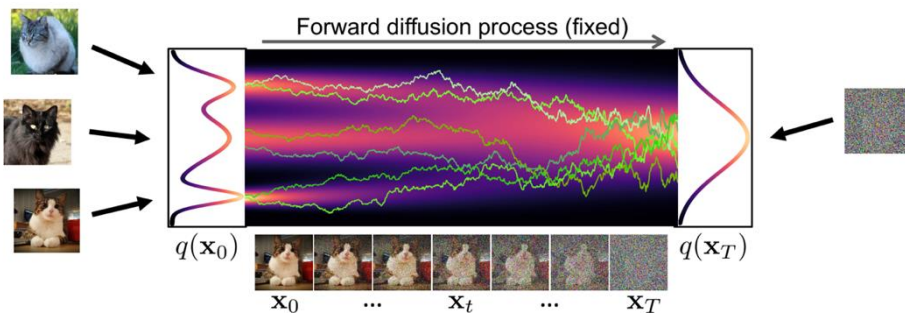
Forward Diffusion SDE:

$$dx_t = -\frac{1}{2}\beta(t)x_t dt + \sqrt{\beta(t)} d\omega_t$$

**Reverse Generative
Diffusion SDE:**

$$dx_t = \underbrace{\left[-\frac{1}{2}\beta(t)x_t - \beta(t) \underbrace{\nabla_{x_t} \log q_t(x_t)}_{\text{"Score Function"}} \right]}_{\text{drift term}} dt + \underbrace{\sqrt{\beta(t)} d\bar{\omega}_t}_{\text{diffusion term}}$$

Score Matching

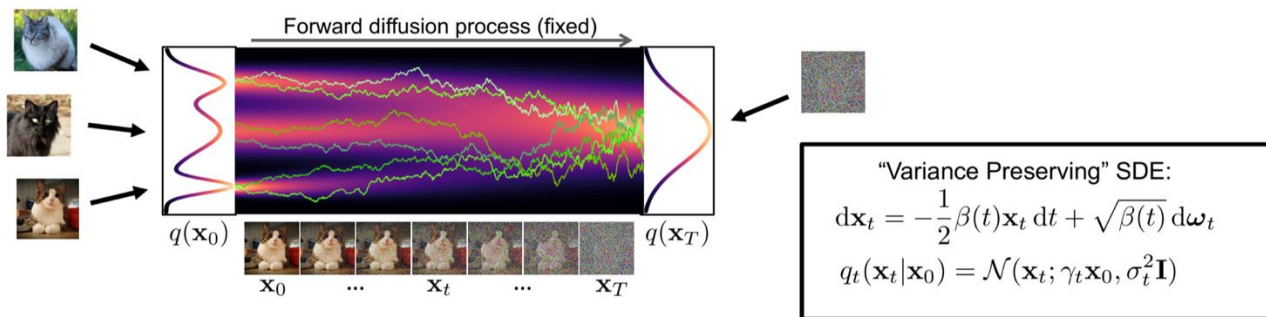


Naïve idea, learn model for the score function by direct regression?

$$\min_{\theta} \underbrace{\mathbb{E}_{t \sim \mathcal{U}(0, T)}}_{\text{diffusion time } t} \underbrace{\mathbb{E}_{\mathbf{x}_t \sim q_t(\mathbf{x}_t)}}_{\text{diffused data } \mathbf{x}_t} \underbrace{\|\mathbf{s}_{\theta}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log q_t(\mathbf{x}_t)\|_2^2}_{\text{score of diffused data (marginal)}}$$

➡ But $\nabla_{\mathbf{x}_t} \log q_t(\mathbf{x}_t)$ (**score of the *marginal diffused density* $q_t(\mathbf{x}_t)$**) is not tractable!

Denoising Score Matching



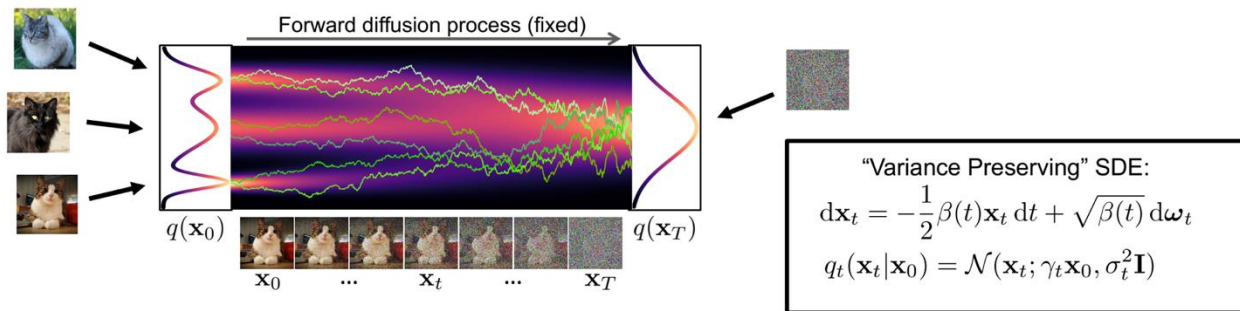
Instead, diffuse individual data points \mathbf{x}_0 . Diffused $q_t(\mathbf{x}_t|\mathbf{x}_0)$ *is* tractable!

Denoising Score Matching:

$$\min_{\boldsymbol{\theta}} \underbrace{\mathbb{E}_{t \sim \mathcal{U}(0,T)}}_{\text{diffusion time } t} \underbrace{\mathbb{E}_{\mathbf{x}_0 \sim q_0(\mathbf{x}_0)}}_{\text{data sample } \mathbf{x}_0} \underbrace{\mathbb{E}_{\mathbf{x}_t \sim q_t(\mathbf{x}_t|\mathbf{x}_0)}}_{\text{diffused data sample } \mathbf{x}_t} \underbrace{\|\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log q_t(\mathbf{x}_t|\mathbf{x}_0)\|_2^2}_{\text{score of diffused data sample}}$$

➔ **After expectations, $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) \approx \nabla_{\mathbf{x}_t} \log q_t(\mathbf{x}_t)$!**

Denoising Score Matching



Instead, diffuse individual data points \mathbf{x}_0 . Diffused $q_t(\mathbf{x}_t|\mathbf{x}_0)$ *is* tractable!

Denoising Score Matching:

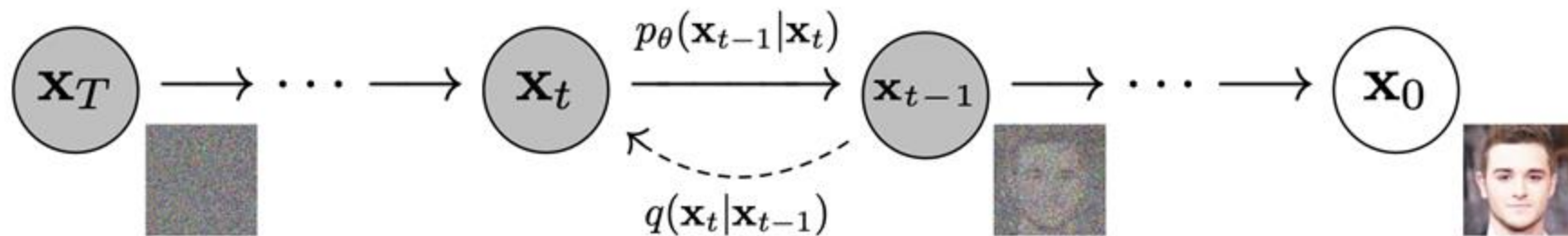
$$\min_{\theta} \underbrace{\mathbb{E}_{t \sim \mathcal{U}(0, T)}}_{\text{diffusion time } t} \underbrace{\mathbb{E}_{\mathbf{x}_0 \sim q_0(\mathbf{x}_0)}}_{\text{data sample } \mathbf{x}_0} \underbrace{\mathbb{E}_{\mathbf{x}_t \sim q_t(\mathbf{x}_t|\mathbf{x}_0)}}_{\text{diffused data sample } \mathbf{x}_t} \underbrace{\|\mathbf{s}_{\theta}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log q_t(\mathbf{x}_t|\mathbf{x}_0)\|_2^2}_{\text{score of diffused data sample}}$$

neural network

$$\rightarrow \min_{\theta} \mathbb{E}_{t \sim \mathcal{U}(0, T)} \mathbb{E}_{\mathbf{x}_0 \sim q_0(\mathbf{x}_0)} \mathbb{E}_{\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})} \frac{1}{\sigma_t^2} \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\theta}(\mathbf{x}_t, t)\|_2^2$$

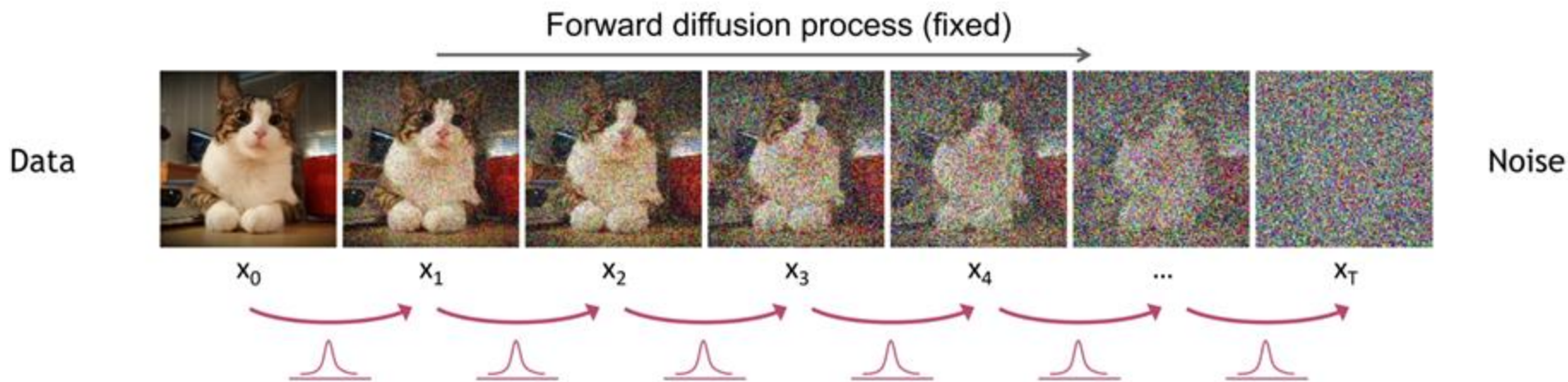
Latent Probabilistic Models Perspective

Denoising Diffusion Probabilistic Models

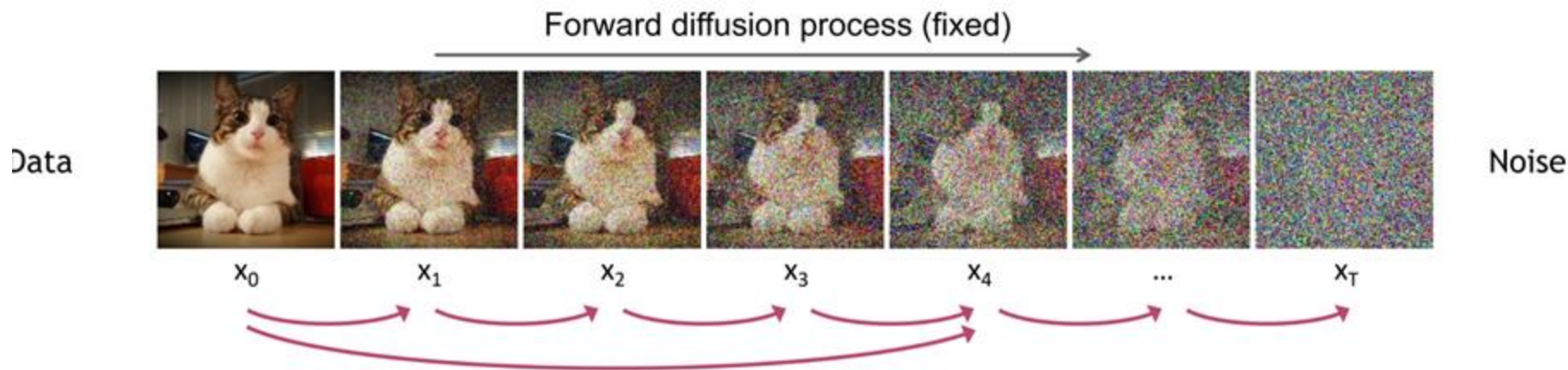


Denoising Diffusion Probabilistic Models

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) := \prod_{t=1}^T q(\mathbf{x}_t|\mathbf{x}_{t-1}), \quad q(\mathbf{x}_t|\mathbf{x}_{t-1}) := \mathcal{N}(\mathbf{x}_t; \sqrt{1 - \beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$



Denoising Diffusion Probabilistic Models

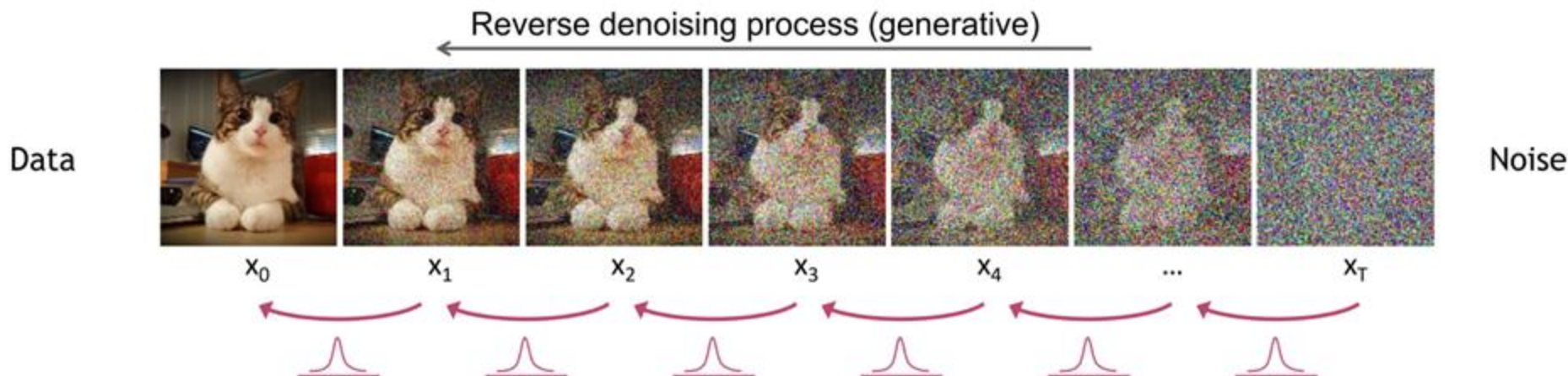


Define $\bar{\alpha}_t = \prod_{s=1}^t (1 - \beta_s)$ $\rightarrow q(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t} \mathbf{x}_0, (1 - \bar{\alpha}_t) \mathbf{I})$ (Diffusion Kernel)

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

Denoising Diffusion Probabilistic Models

$$p_{\theta}(\mathbf{x}_{0:T}) := p(\mathbf{x}_T) \prod_{t=1}^T p_{\theta}(\mathbf{x}_{t-1} | \mathbf{x}_t), \quad p_{\theta}(\mathbf{x}_{t-1} | \mathbf{x}_t) := \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_{\theta}(\mathbf{x}_t, t), \boldsymbol{\Sigma}_{\theta}(\mathbf{x}_t, t))$$



Denoising Diffusion Probabilistic Models

$$\mathbb{E}[-\log p_{\theta}(\mathbf{x}_0)] \leq \mathbb{E}_q \left[-\log \frac{p_{\theta}(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \right] = \mathbb{E}_q \left[-\log p(\mathbf{x}_T) - \sum_{t \geq 1} \log \frac{p_{\theta}(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})} \right] =: L$$

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

Denoising Diffusion Probabilistic Models

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

$$\text{where } \tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) := \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1 - \bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t}\mathbf{x}_t \quad \text{and} \quad \tilde{\beta}_t := \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t}\beta_t$$

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

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

Denoising Diffusion Probabilistic Models

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

$$\boldsymbol{\mu}_\theta(\mathbf{x}_t, t) = \tilde{\boldsymbol{\mu}}_t \left(\mathbf{x}_t, \frac{1}{\sqrt{\bar{\alpha}_t}} (\mathbf{x}_t - \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t)) \right)$$

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

Denoising Diffusion Probabilistic Models

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

Algorithm 1 Training

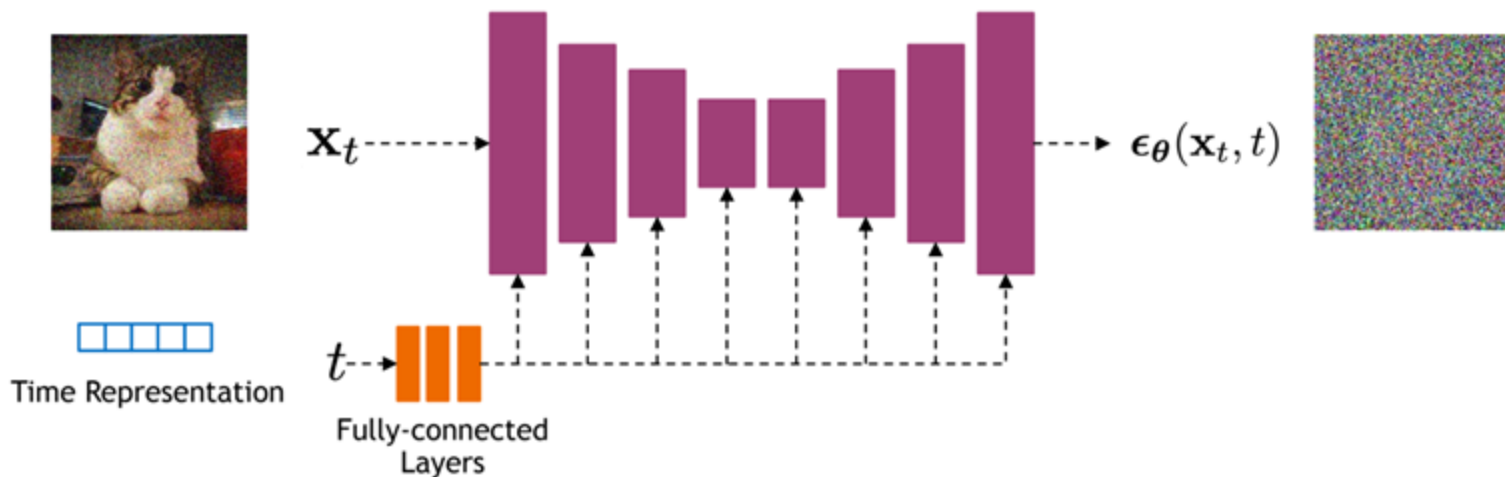
- 1: **repeat**
 - 2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
 - 3: $t \sim \text{Uniform}(\{1, \dots, T\})$
 - 4: $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
 - 5: Take gradient descent step on
 $\nabla_{\theta} \left\| \epsilon - \epsilon_{\theta}(\sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 -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

Network architectures

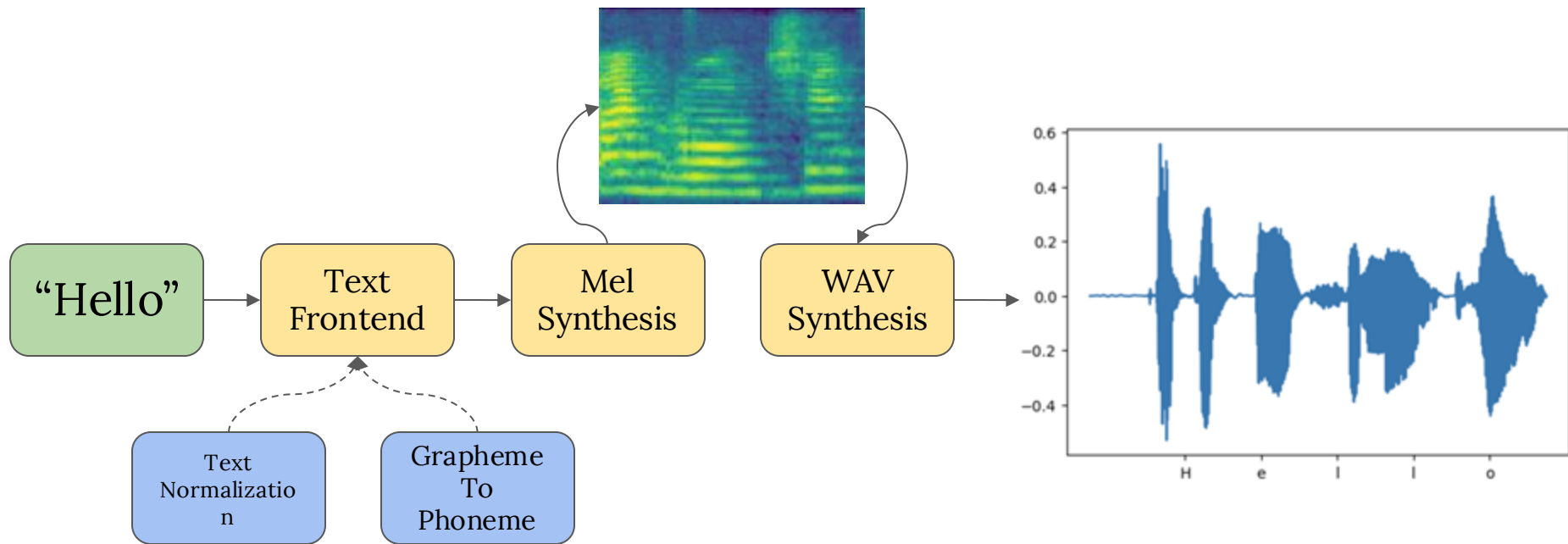
Diffusion models often use U-Net architectures with ResNet blocks and self-attention layers to represent $\epsilon_{\theta}(\mathbf{x}_t, t)$



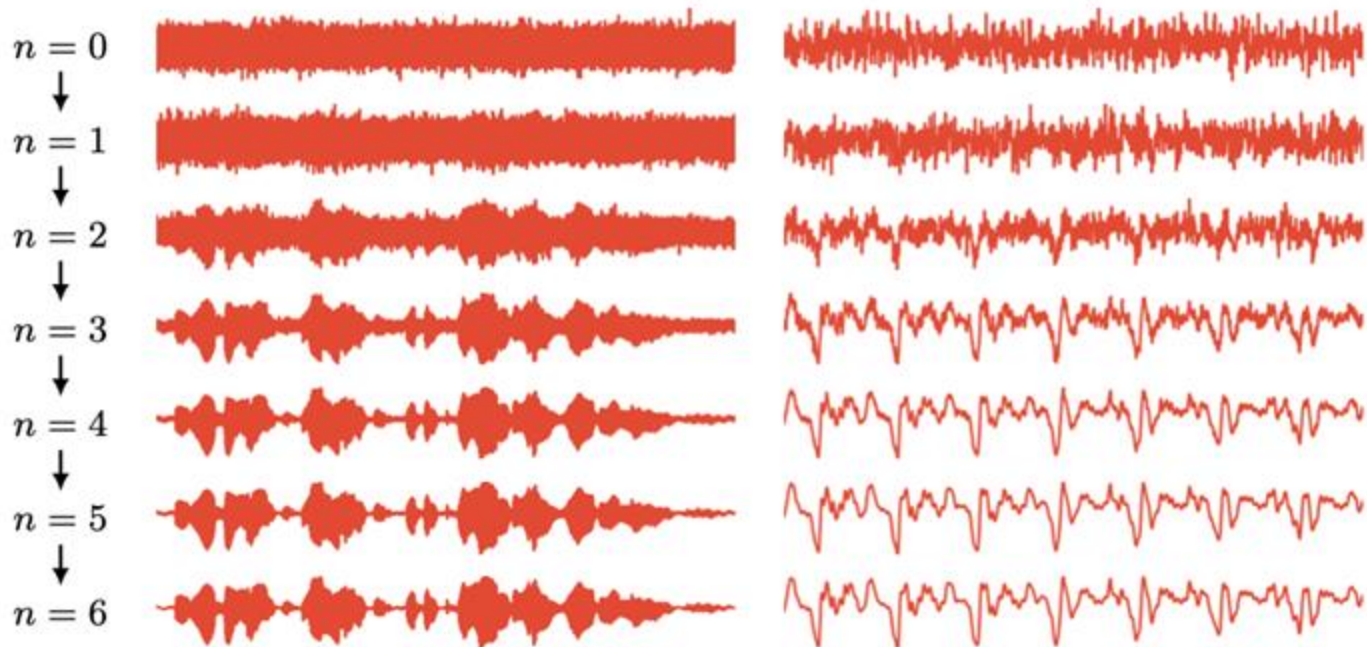
Time representation: sinusoidal positional embeddings or random Fourier features.

Applications of Diffusion Models in Audio

TTS recap



WaveGrad - Vocoder

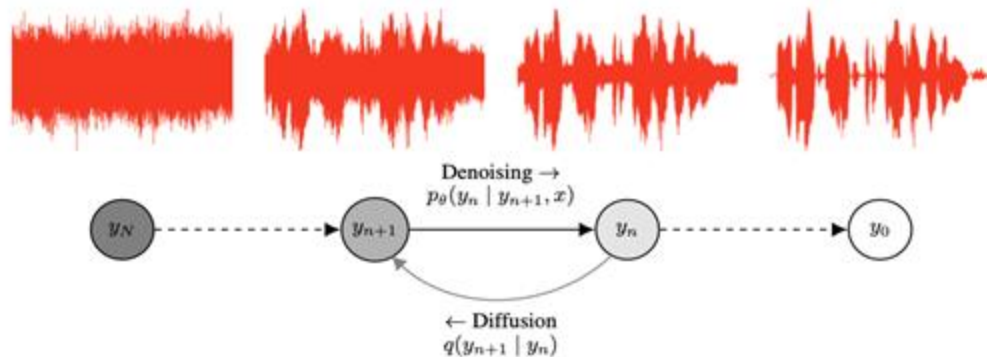


WaveGrad - Vocoder

$$s(y) = \nabla_y \log p(y).$$

$$\tilde{y}_{i+1} = \tilde{y}_i + \frac{\eta}{2} s(\tilde{y}_i) + \sqrt{\eta} z_i,$$

$$\mathbb{E}_{y \sim p(y)} \mathbb{E}_{\tilde{y} \sim q(\tilde{y} | y)} \left[\left\| s_{\theta}(\tilde{y}) - \nabla_{\tilde{y}} \log q(\tilde{y} | y) \right\|_2^2 \right]$$



WaveGrad - Vocoder

$$q(y_{1:N} \mid y_0) := \prod_{n=1}^N q(y_n \mid y_{n-1}), \quad q(y_n \mid y_{n-1}) := \mathcal{N}\left(y_n; \sqrt{(1 - \beta_n)} y_{n-1}, \beta_n I\right),$$

$$y_n = \sqrt{\bar{\alpha}_n} y_0 + \sqrt{(1 - \bar{\alpha}_n)} \epsilon \quad \nabla_{y_n} \log q(y_n \mid y_0) = -\frac{\epsilon}{\sqrt{1 - \bar{\alpha}_n}}.$$

$$p_\theta(y_0 \mid x) := \int p_\theta(y_{0:N} \mid x) \, dy_{1:N}$$

$$\mathbb{E}_{n,\epsilon} \left[C_n \left\| \epsilon_\theta \left(\sqrt{\bar{\alpha}_n} y_0 + \sqrt{1 - \bar{\alpha}_n} \epsilon, x, n \right) - \epsilon \right\|_2^2 \right]$$

WaveGrad - Vocoder

$$\mathbb{E}_{\bar{\alpha}, \epsilon} \left[\left\| \epsilon_{\theta} \left(\sqrt{\bar{\alpha}} y_0 + \sqrt{1 - \bar{\alpha}} \epsilon, x, \sqrt{\bar{\alpha}} \right) - \epsilon \right\|_1 \right],$$

$$l_0 = 1, \quad l_s = \sqrt{\prod_{i=1}^s (1 - \beta_i)}.$$

We first sample a segment $s \sim U(\{1, \dots, S\})$, which provides a segment (l_{s-1}, l_s) , and then sample from this segment uniformly to give $\sqrt{\bar{\alpha}}$.

WaveGrad - Vocoder

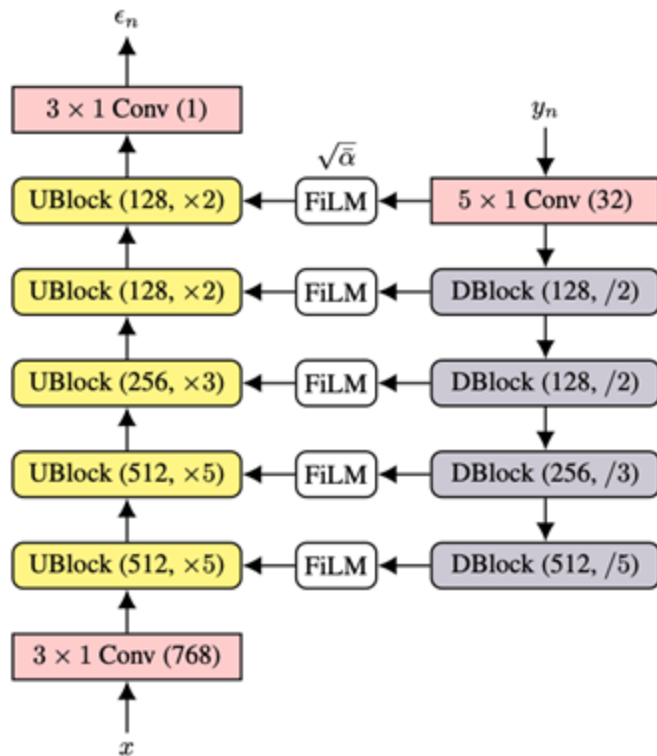
Algorithm 1 Training. WaveGrad directly conditions on the continuous noise level $\sqrt{\bar{\alpha}}$. l is from a predefined noise schedule.

```
1: repeat  
2:    $y_0 \sim q(y_0)$   
3:    $s \sim \text{Uniform}(\{1, \dots, S\})$   
4:    $\sqrt{\bar{\alpha}} \sim \text{Uniform}(l_{s-1}, l_s)$   
5:    $\epsilon \sim \mathcal{N}(0, I)$   
6:   Take gradient descent step on  
      $\nabla_{\theta} \|\epsilon - \epsilon_{\theta}(\sqrt{\bar{\alpha}} y_0 + \sqrt{1 - \bar{\alpha}} \epsilon, x, \sqrt{\bar{\alpha}})\|_1$   
7: until converged
```

Algorithm 2 Sampling. WaveGrad generates samples following a gradient-based sampler similar to Langevin dynamics.

```
1:  $y_N \sim \mathcal{N}(0, I)$   
2: for  $n = N, \dots, 1$  do  
3:    $z \sim \mathcal{N}(0, I)$   
4:    $y_{n-1} = \frac{\left(y_n - \frac{1 - \alpha_n}{\sqrt{1 - \bar{\alpha}_n}} \epsilon_{\theta}(y_n, x, \sqrt{\bar{\alpha}_n})\right)}{\sqrt{\alpha_n}}$   
5:   if  $n > 1$ ,  $y_{n-1} = y_{n-1} + \sigma_n z$   
6: end for  
7: return  $y_0$ 
```

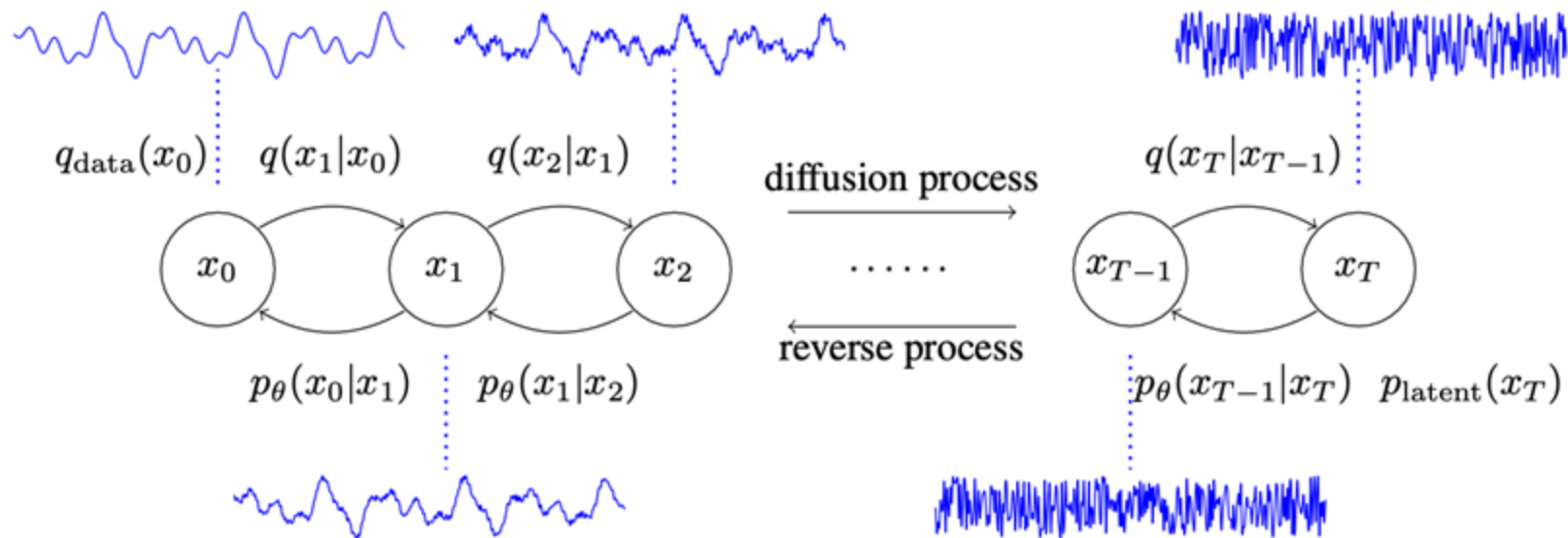
WaveGrad - Vocoder



WaveGrad - Vocoder

Model	MOS (\uparrow)
WaveRNN	4.49 ± 0.04
Parallel WaveGAN	3.92 ± 0.05
MelGAN	3.95 ± 0.06
Multi-band MelGAN	4.10 ± 0.05
GAN-TTS	4.34 ± 0.04
WaveGrad	
Base (6 iterations, continuous noise levels)	4.41 ± 0.03
Base (1,000 iterations, discrete indices)	4.47 ± 0.04
Large (1,000 iterations, discrete indices)	4.51 ± 0.04
Ground Truth	4.58 ± 0.05

DiffWave - Vocoder



DiffWave - Vocoder

$$q(x_1, \dots, x_T | x_0) = \prod_{t=1}^T q(x_t | x_{t-1}),$$

$$p_{\text{latent}}(x_T) = \mathcal{N}(0, I), \text{ and } p_{\theta}(x_0, \dots, x_{T-1} | x_T) = \prod_{t=1}^T p_{\theta}(x_{t-1} | x_t),$$

$$\min_{\theta} L_{\text{unweighted}}(\theta) = \mathbb{E}_{x_0, \epsilon, t} \|\epsilon - \epsilon_{\theta}(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon, t)\|_2^2$$

DiffWave - Vocoder

Algorithm 1 Training

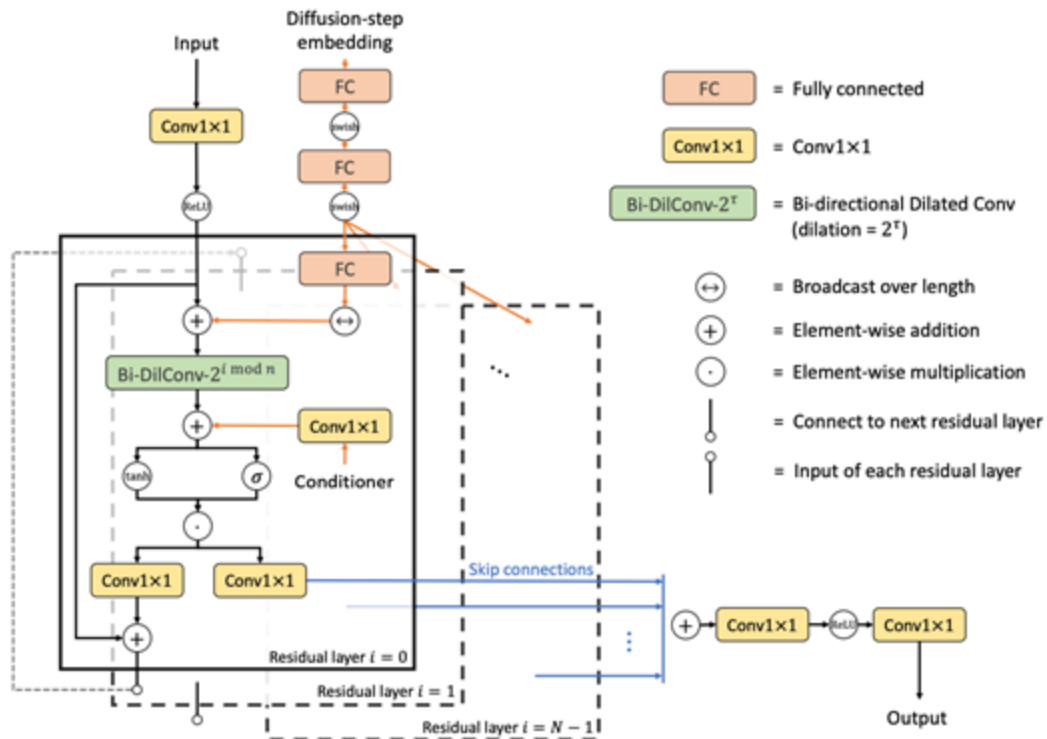
```
for  $i = 1, 2, \dots, N_{\text{iter}}$  do
  Sample  $x_0 \sim q_{\text{data}}$ ,  $\epsilon \sim \mathcal{N}(0, I)$ , and
   $t \sim \text{Uniform}(\{1, \dots, T\})$ 
  Take gradient step on
   $\nabla_{\theta} \|\epsilon - \epsilon_{\theta}(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon, t)\|_2^2$ 
  according to Eq. (7)
end for
```

Algorithm 2 Sampling

```
Sample  $x_T \sim p_{\text{latent}} = \mathcal{N}(0, I)$ 
for  $t = T, T-1, \dots, 1$  do
  Compute  $\mu_{\theta}(x_t, t)$  and  $\sigma_{\theta}(x_t, t)$  using Eq. (5)
  Sample  $x_{t-1} \sim p_{\theta}(x_{t-1}|x_t) =$ 
   $\mathcal{N}(x_{t-1}; \mu_{\theta}(x_t, t), \sigma_{\theta}(x_t, t)^2 I)$ 
end for
return  $x_0$ 
```

$$\mu_{\theta}(x_t, t) = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1 - \bar{\alpha}_t}} \epsilon_{\theta}(x_t, t) \right), \text{ and } \sigma_{\theta}(x_t, t) = \tilde{\beta}_t^{\frac{1}{2}}, \quad (5)$$

DiffWave - Vocoder



DiffWave - Vocoder

Model	T	T_{infer}	layers	res. channels	#param(\downarrow)	MOS(\uparrow)
WaveNet	—	—	30	128	4.57M	4.43 \pm 0.10
ClariNet	—	—	60	64	2.17M	4.27 \pm 0.09
WaveGlow	—	—	96	256	87.88M	4.33 \pm 0.12
WaveFlow	—	—	64	64	5.91M	4.30 \pm 0.11
WaveFlow	—	—	64	128	22.25M	4.40 \pm 0.07
DiffWave _{BASE}	20	20	30	64	2.64M	4.31 \pm 0.09
DiffWave _{BASE}	40	40	30	64	2.64M	4.35 \pm 0.10
DiffWave _{BASE}	50	50	30	64	2.64M	4.38 \pm 0.08
DiffWave _{LARGE}	200	200	30	128	6.91M	4.44 \pm 0.07
DiffWave _{BASE} (Fast)	50	6	30	64	2.64M	4.37 \pm 0.07
DiffWave _{LARGE} (Fast)	200	6	30	128	6.91M	4.42 \pm 0.09
Ground-truth	—	—	—	—	—	4.52 \pm 0.06

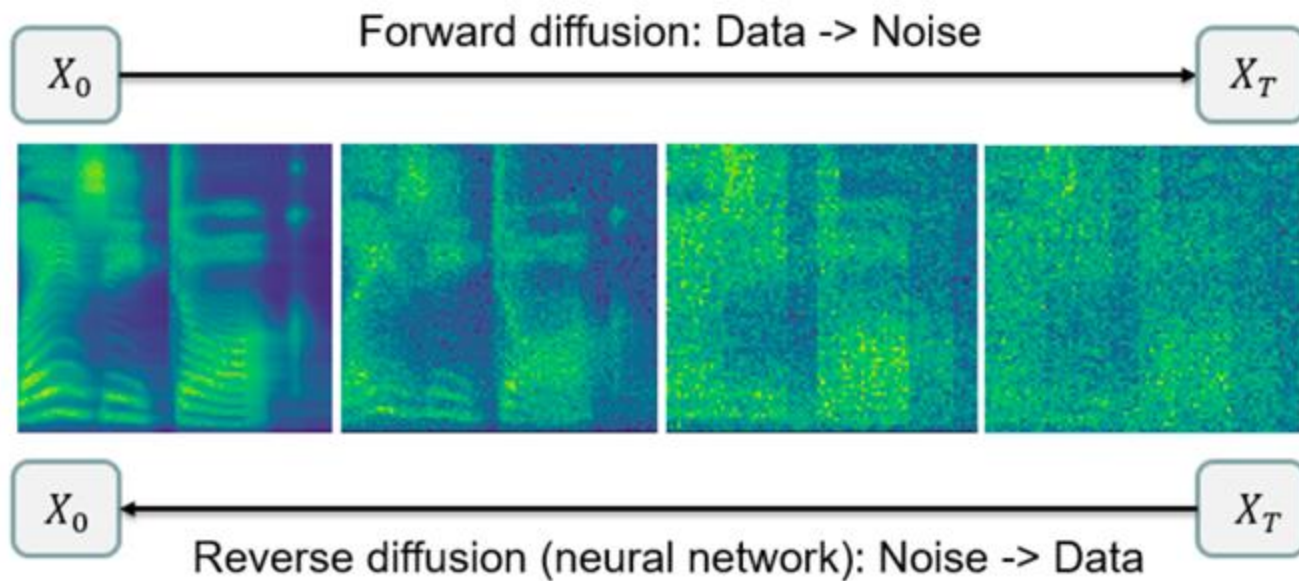
DiffWave - Vocoder

Model	FID(↓)	IS(↑)	mIS(↑)	AM(↓)	NDB/ K (↓)	MOS(↑)
WaveNet-128	3.279	2.54	7.6	1.368	0.86	1.34 ± 0.29
WaveNet-256	2.947	2.84	10.0	1.260	0.86	1.43 ± 0.30
WaveGAN	1.349	4.53	36.6	0.796	0.78	2.03 ± 0.33
DiffWave	1.287	5.30	59.4	0.636	0.74	3.39 ± 0.32
Trainset	0.000	8.48	281.4	0.164	0.00	—
Testset	0.011	8.47	275.2	0.166	0.10	3.72 ± 0.28

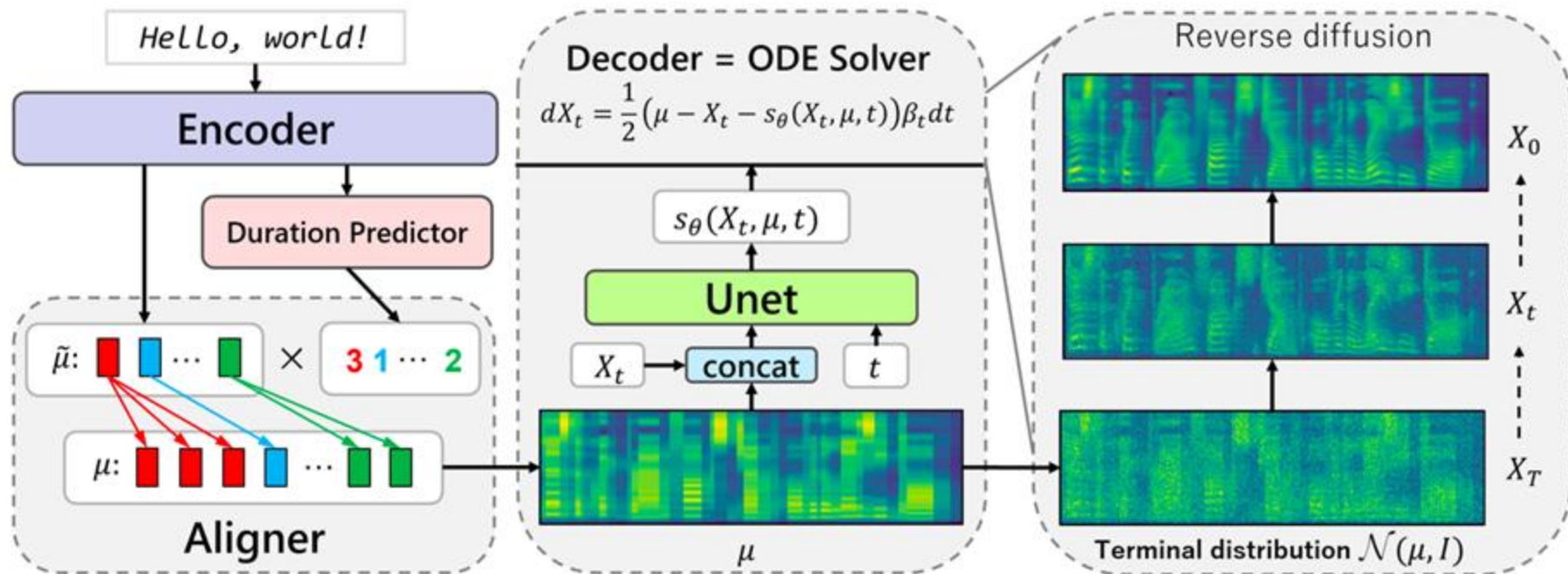
DiffWave - Vocoder

Model	Accuracy(\uparrow)	FID-class(\downarrow)	IS(\uparrow)	mIS(\uparrow)	MOS(\uparrow)
WaveNet-128	56.20%	7.876 ± 2.469	3.29	15.8	1.46 ± 0.30
WaveNet-256	60.70%	6.954 ± 2.114	3.46	18.9	1.58 ± 0.36
DiffWave	91.20%	1.113 ± 0.569	6.63	117.4	3.50 ± 0.31
DiffWave (deep & thin)	94.00%	0.932 ± 0.450	6.92	133.8	3.44 ± 0.36
Trainset	99.06%	0.000 ± 0.000	8.48	281.4	—
Testset	98.76%	0.044 ± 0.016	8.47	275.2	3.72 ± 0.28

GradTTS - Mel synthesis



GradTTS - Mel synthesis



GradTTS - Mel synthesis

Table 2. Model comparison.

Model	Enc params ¹	Dec params	RTF	Log-likelihood	MOS
Grad-TTS-1000	<i>7.2m</i>	<i>7.6m</i>	3.663	0.174 ± 0.001	4.44 ± 0.05
Grad-TTS-100			0.363		4.38 ± 0.06
Grad-TTS-10			0.033		4.38 ± 0.06
Grad-TTS-4			0.012		3.96 ± 0.07
Glow-TTS	<i>7.2m</i>	<i>21.4m</i>	0.008	0.082	4.11 ± 0.07
FastSpeech	<i>24.5m</i>		0.004	—	3.68 ± 0.09
Tacotron2	<i>28.2m</i>		0.075	—	4.32 ± 0.07
Ground Truth	—		—	—	4.53 ± 0.06

Important papers

- [Generative Modeling by Estimating Gradients of the Data Distribution](#)
- [Denoising Diffusion Probabilistic Models](#)
- [Score-Based Generative Modeling through Stochastic Differential Equations](#)
- [WaveGrad: Estimating Gradients for Waveform Generation](#)
- [DiffWave: A Versatile Diffusion Model for Audio Synthesis](#)
- [Grad-TTS: A Diffusion Probabilistic Model for Text-to-Speech](#)

Additional papers

- [Diff-TTS: A Denoising Diffusion Model for Text-to-Speech](#)
- [ProDiff: Progressive Fast Diffusion Model For High-Quality Text-to-Speech](#)
- [DiffGAN-TTS: High-Fidelity and Efficient Text-to-Speech with Denoising Diffusion GANs](#)
- [BDDM: Bilateral Denoising Diffusion Models for Fast and High-Quality Speech Synthesis](#)
- [SpecGrad: Diffusion Probabilistic Model based Neural Vocoder with Adaptive Noise Spectral Shaping](#)
- [WaveGrad 2: Iterative Refinement for Text-to-Speech Synthesis](#)

Contacts



Head of Controllable Generative AI,
FusionBrain Lab, AIRI
Head of BayesLab, HSE University

Email: alanov.aibek@gmail.com

Telegram: [@aibrain](https://t.me/aibrain)

