Diffusion Probabilistic Models

- Concurrently, a similar to score-matching class of models: diffusion models
- Diffusion models also define a forward and reverse diffusion process, where $t = 0$ corresponds to the data distribution, and $t = T$ a unit-Gaussian distribution

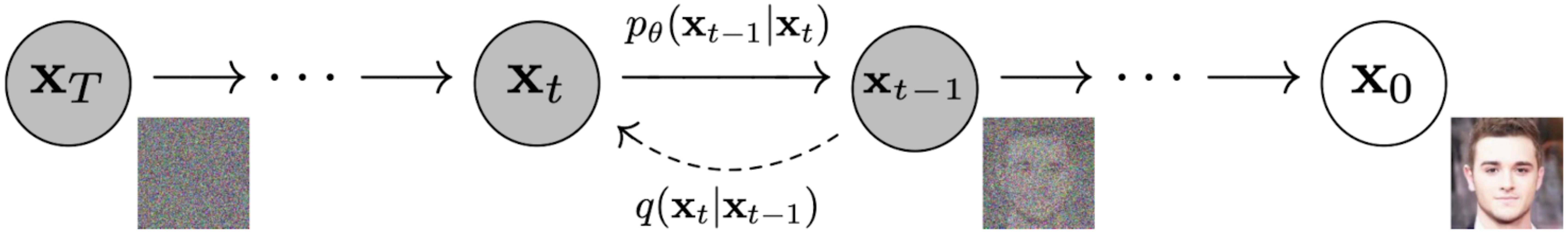

Figure 2: The directed graphical model considered in this work.

Diffusion probabilistic models, Sohl-Dickstein et al., 2015
Denoising diffusion probabilistic models, Ho et al., 2020
Diffusion models beat GANs on image synthesis, 2021
https://lilianweng.github.io/posts/2021-07-11-diffusion-models/

Forward diffusion process

- In forward diffusion we add small Gaussian noise to our data till it looks like isotropic Gaussian

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

- We can define the conditional distribution at any time step t w.r.t. step $t = 0$

$$\begin{aligned} \mathbf{x}_t &= \sqrt{a_t}\mathbf{x}_{t-1} + \sqrt{1-a_t}\mathbf{z}_{t-1} && \text{, where } \mathbf{z}_{t-1}, \mathbf{z}_{t-2}, \ldots \sim \mathcal{N}(0,1) \\ &= \sqrt{a_t a_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-a_t a_{t-1}}\bar{\mathbf{z}}_{t-2} && \text{, where } \bar{\mathbf{z}}_{t-2} \text{ merges two Guassians} \\ &= \ldots \\ &= \sqrt{\bar{a}_t}\mathbf{x}_0 + \sqrt{1-\bar{a}_t}\bar{\mathbf{z}} \end{aligned}$$

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

- For this we use that when merging two Gaussians, we get another Gaussian with variance $\sigma_1^2 + \sigma_2^2$

Reverse diffusion process

- Efficiently parameterising reverse diffusion we can combine with variational inference

$$L_{VLB} = L_T + L_{T-1} + \ldots + L_0$$

$$\text{where } L_T = D_{KL}\big(q(\mathbf{x}_T \,|\, \mathbf{x}_0) \| p_\theta(\mathbf{x}_T)\big)$$

$$L_t = D_{KL}\big(q(\mathbf{x}_t \,|\, \mathbf{x}_{t+1}, \mathbf{x}_0) \| p_\theta(\mathbf{x}_t \,|\, \mathbf{x}_{t+1})\big) \text{ for } 1 \leq t \leq T-1$$

$$L_0 = -\log p_\theta(\mathbf{x}_0 \,|\, \mathbf{x}_1)$$

- Since we have Gaussian distributions the KL terms can be computed in closed form
- L_T does not depend any parameters and it can be dropped
- L_0 depends on the final decoder output

Parameterising L_t

- By smart parameterisation of the Gaussians, learning boils down to minimising

$$L_t^{\text{simple}} = \mathbb{E}_{\mathbf{x}_0,\epsilon_t}\left[\left\|\epsilon_t - \epsilon_\theta\left(\sqrt{\bar{a}_t}\mathbf{x}_0 + \sqrt{1-\bar{a}_t}\epsilon_t, t\right)\right\|_2^2\right]$$

- The model learns to predict the noise added to the transformed signal!!!!

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** $\mathbf{x}_0$

Example trajectories

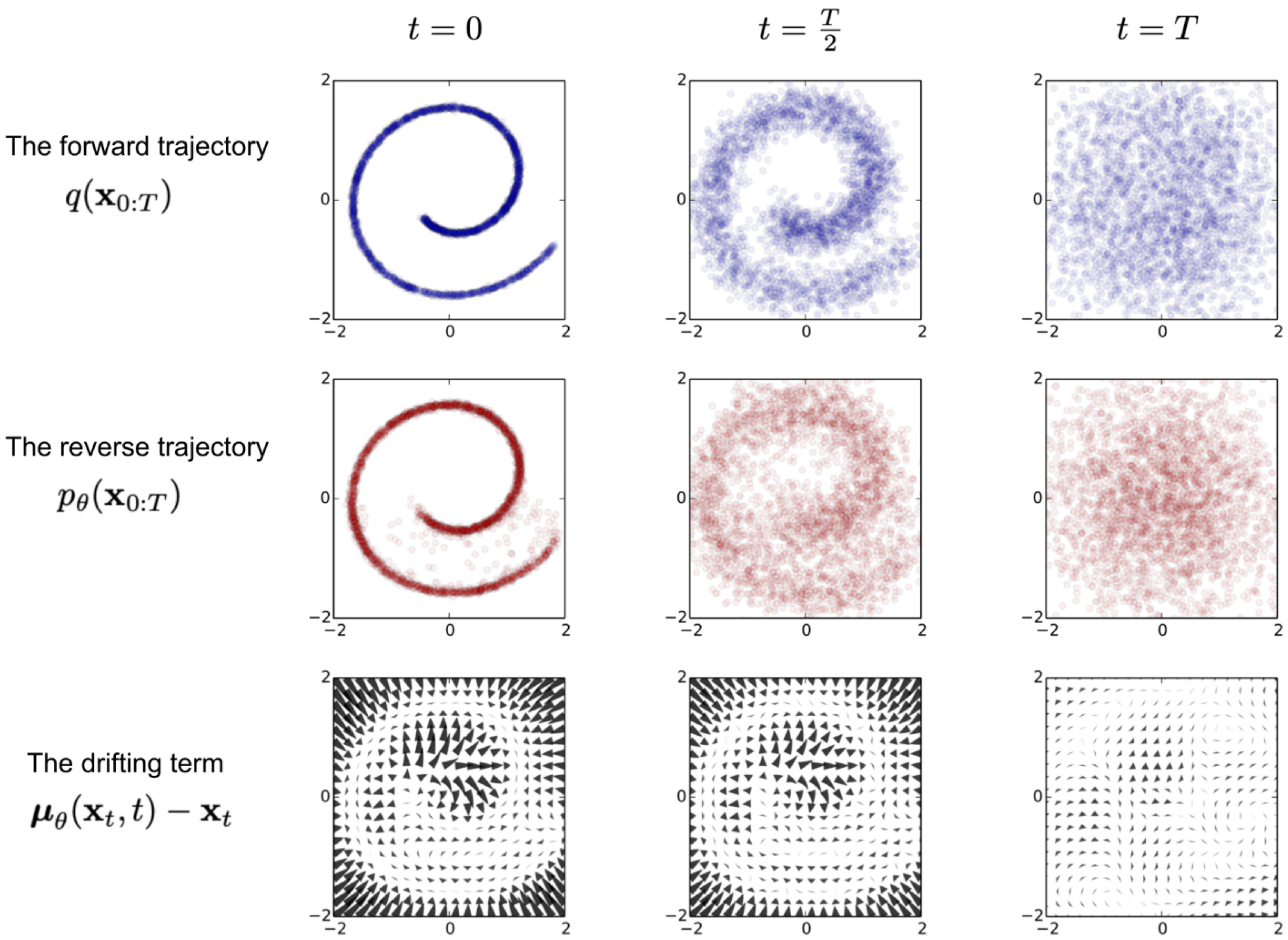

Fig. 3. An example of training a diffusion model for modeling a 2D swiss roll data. (Image source: Sohl-Dickstein et al., 2015)

https://lilianweng.github.io/posts/2021-07-11-diffusion-models/

Qualitative results

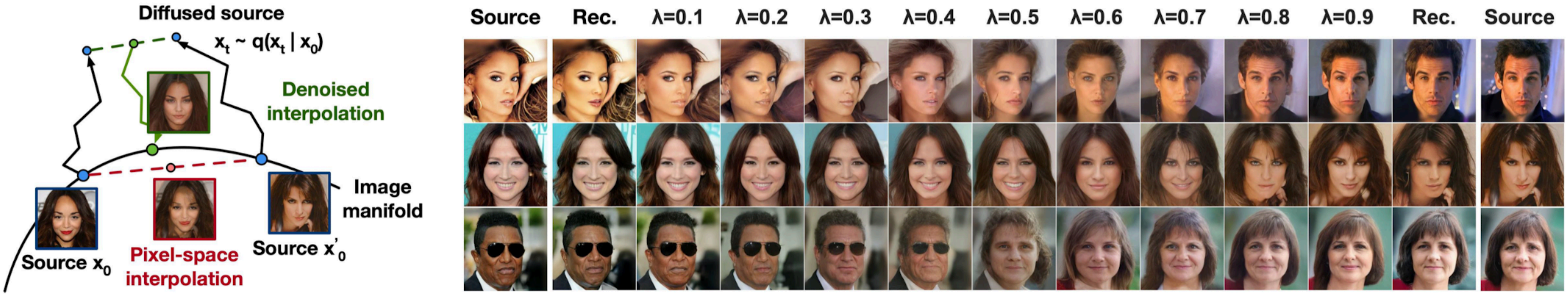

Figure 8: Interpolations of CelebA-HQ 256x256 images with 500 timesteps of diffusion.

Denoising diffusion probabilistic models, Ho et al., 2020

Take-home message

- Diffusion/score-matching models are both tractable and flexible
- However, they are still quite slow to sample from compared to GANs
- The reason is that they require very long chains of time steps up to $T = 1{,}000$
- Great opportunities for learning the data structure effectively and efficiently enough
- Promising results in modelling inverse problems