Manifold Learning and Artificial Intelligence

Lecture 3

Generative AI

Meeting ID: 933 1613 9423

Passcode: 416262

https://uwmadison.zoom.us/j/93316139423?pwd=Q0NVWFYvRFg5RmxCNkwxMmYrbW41dz09

momiao. xiong@gmail.com

shihcheng.guo@gmail.com

Github Address: https://ai2healthcare.github.io/

Introduction 3

Generative AI

Early Stage:

- Variational Autoencoder (VAE)
- Generative Adversarial Neural Network (GAN)
- Normalizing Flow

Current Stage

- Score Matching
- Denoise Diffusion Probability Models
- Stochastic Differential Equations (SDE)

The goal of generative modeling is to use the dataset to learn a model for generating new samples from $P_{data}(x)$

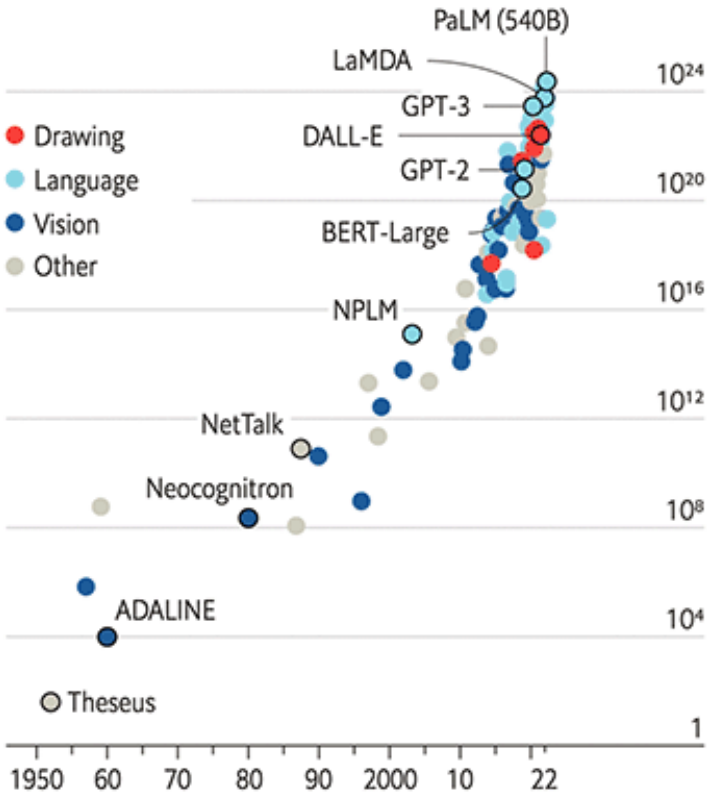
Sure enough, as the models get bigger and bigger, they begin to deliver human-level, and then superhuman results.

https://www.sequoiacap.com/article/generative-ai-a-creative-new-world/

The blessings of scale

AI training runs, estimated computing resources used

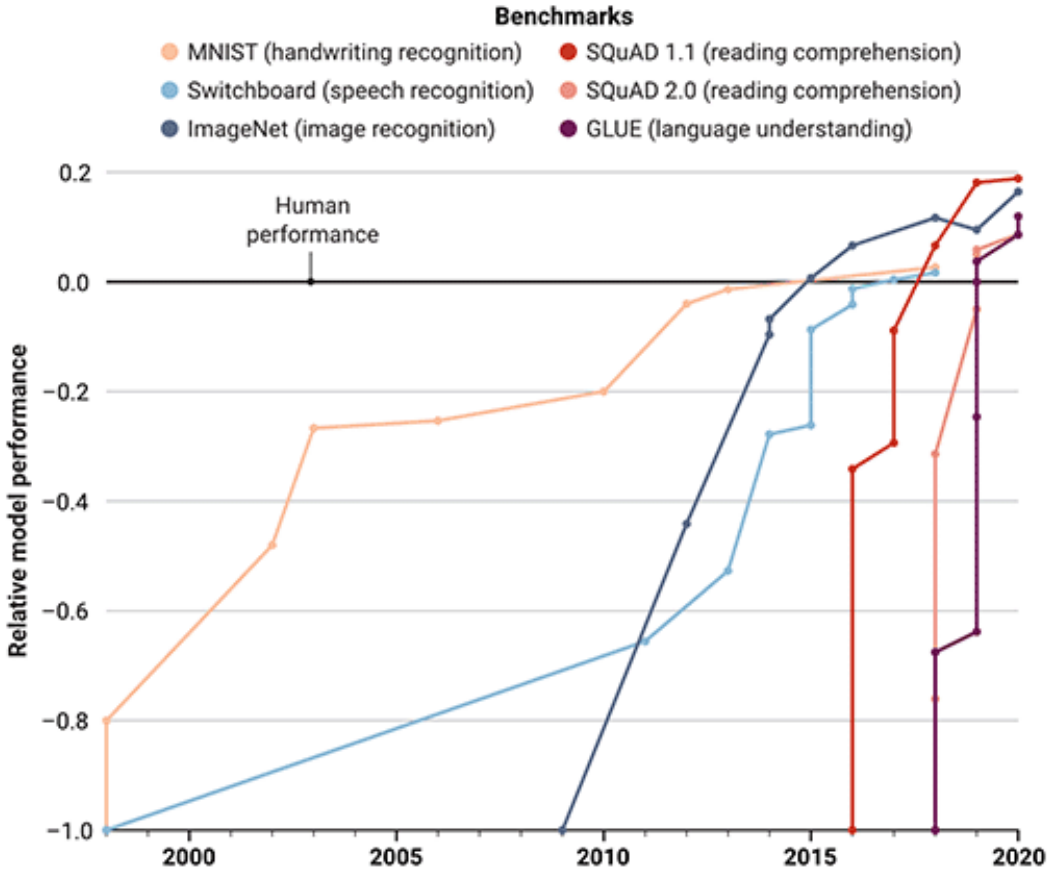
Floating-point operations, selected systems, by type, log scale



Sources: "Compute trends across three eras of machine learning", by J. Sevilla et al., arXiv, 2022; Our World in Data

Quick learners

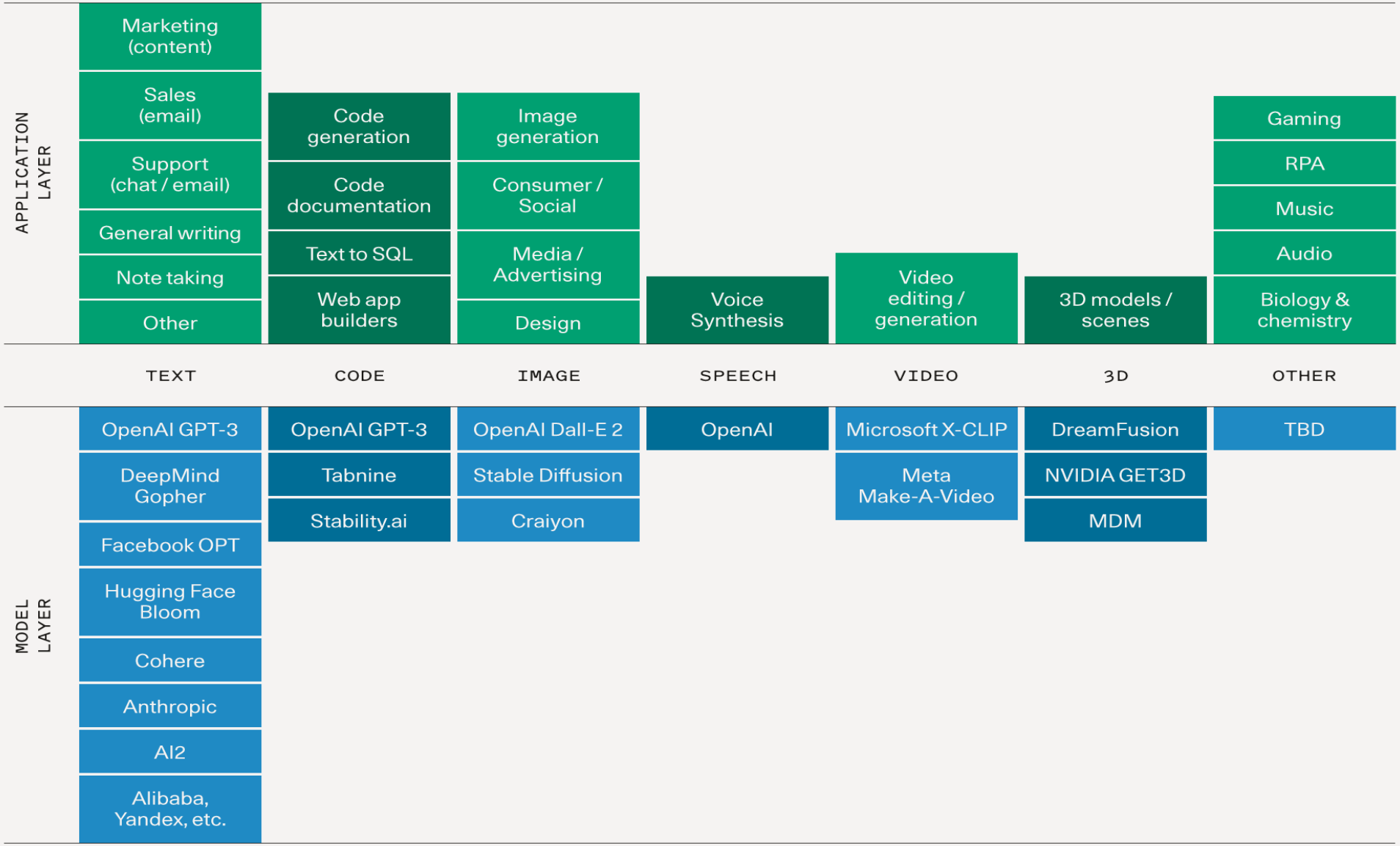
The speed at which artificial intelligence models master benchmarks and surpass human baselines is accelerating. But they often fall short in the real world.



(GRAPHIC) K. FRANKLIN/*SCIENCE*; (DATA) D. KIELA *ET AL.*, DYNABENCH: RETHINKING BENCHMARKING IN NLP, DOI:10.48550/ARXIV.2104.14337

SEQUOIA

The Generative AI Application Landscape



SEQUOIA

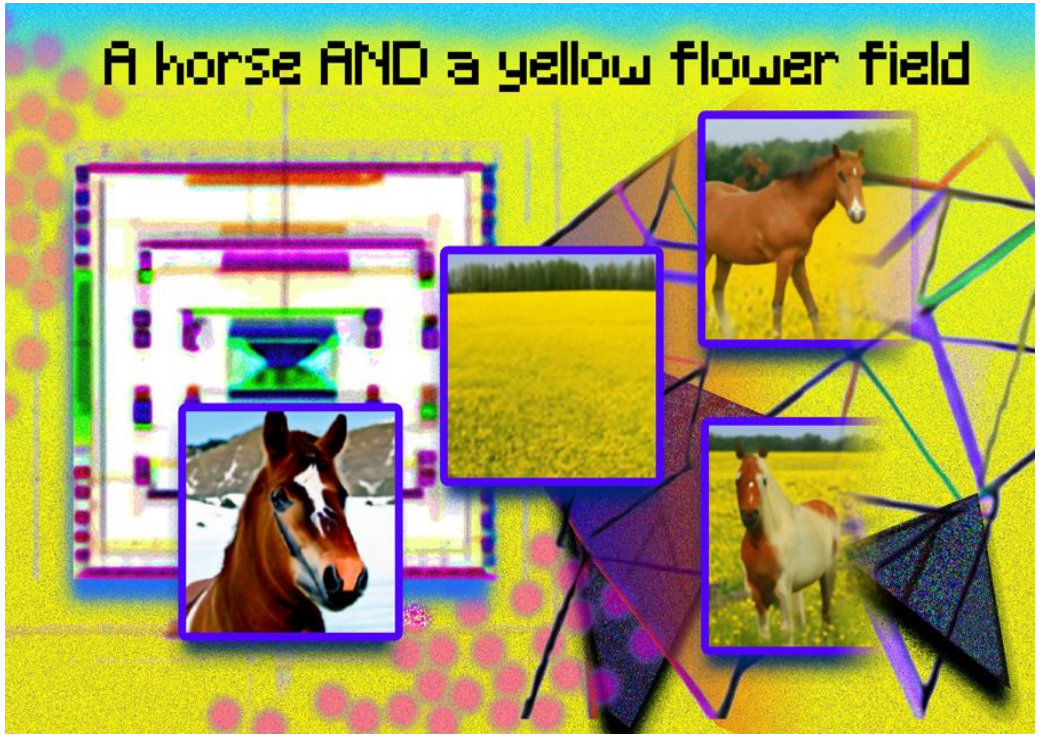
1.4. Generative Models

1.4.1. Introduction

- **Analytic AI: analyze a set of data and find patterns in it.**
- **Generative AI: Use existing content like text, audio files, or images to create new plausible content.**
- **The MIT Technology review described generative AI as one of the most promising advances in the world of AI in the past decade**
- **Generative AI is well on the way to becoming not just faster and cheaper, but better in some cases than what humans create by hand (SEQUOIA)**

IBM developer blog. What is generative AI and how much power does it have

MIT AI and ML



https://www.csail.mit.edu/

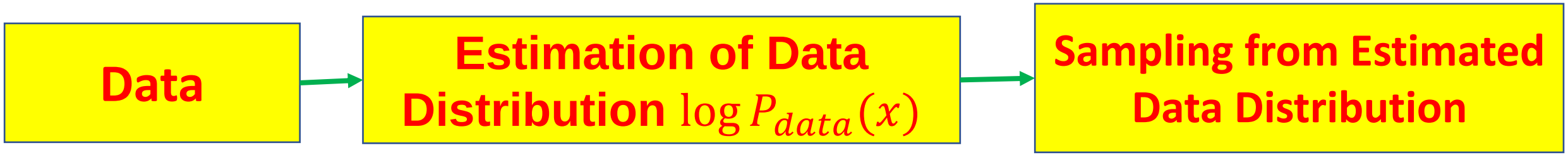
清華大學 丘成桐数学科学中心
Yau Mathematical Sciences Center, Tsinghua University

Deep Generative Models

This course introduces how to develop deep generative models (DGMs) by integrating probabilistic graphical models and deep learning to generate realistic data including images, texts, graphs, etc. Course contents include 1) basics of probabilistic graphical models, including Bayesian network and Markov random field; 2) posterior inference methods, including message passing, variational inference, and Markov chain Monte Carlo sampling; 3) parameter learning and structure learning methods, including maximum likelihood estimation, expectation–maximization algorithm, and graphical LASSO; 4) deep generative models (DGMs), including variational auto-encoder, generative adversarial networks, normalizing flows, and evaluation of DGMs; 5) applications of DGMs in image generation, text generation, and graph generation.

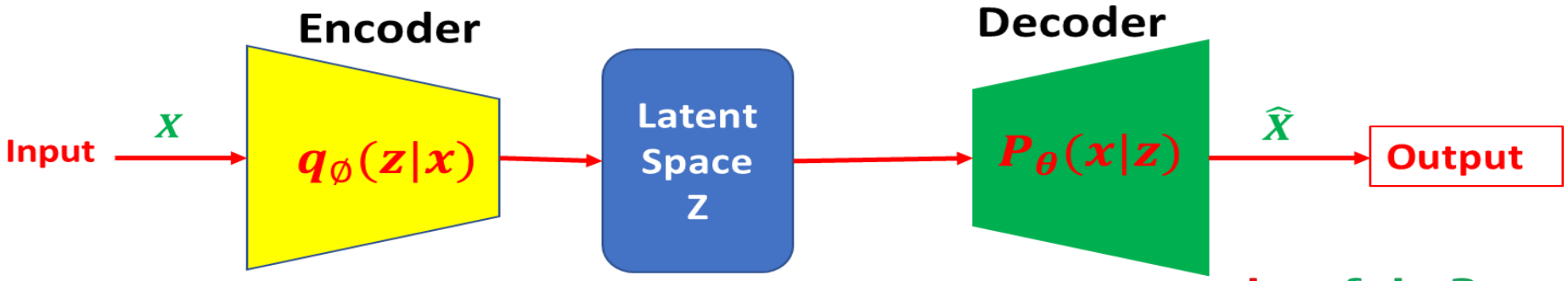
1.4.2. Stages of Generative Models

- **Basic Ideas:**

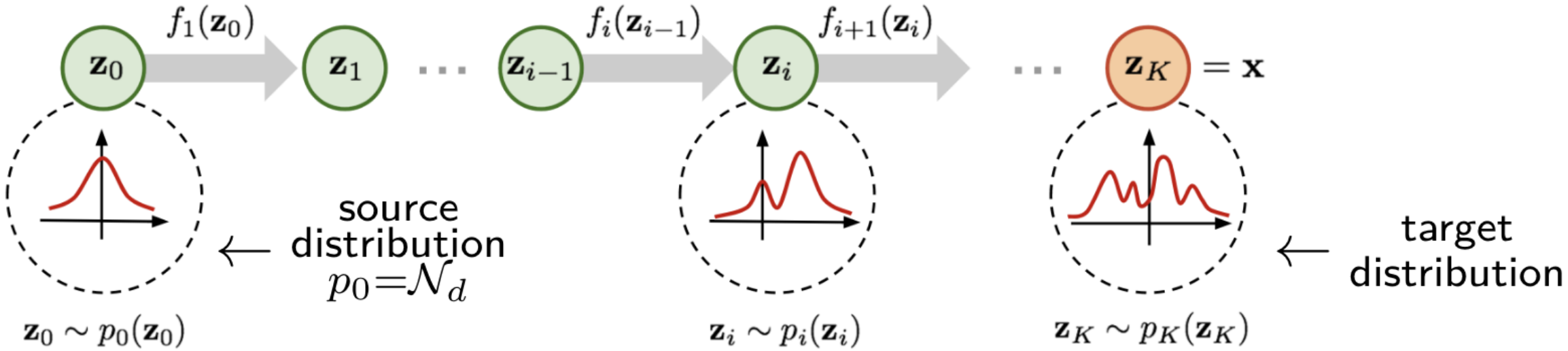


- **Early Stage**

VAE



GAN



Normalizing Flows

Kingma and Welling, 2014; Auto-Encoding Variational Bayes

Goodfellow et al. 2014; Generative Adversarial Nets

Rezende 2016, Variational inference with normalizing flows.

Xiong MM (2022) Artificial Intelligence and Causal Inference. CRC Press

1.4.2. Stages of Generative Models

- **Current Stage**

  Score Matching

  Denoise Diffusion Probability Models

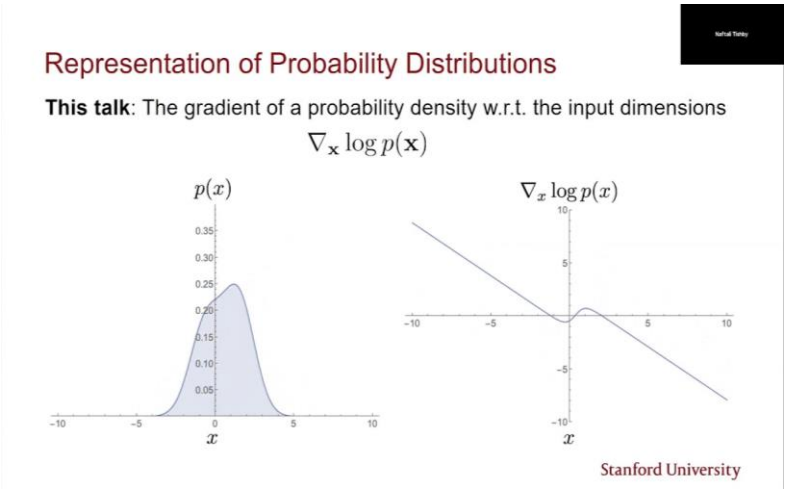
  Stochastic Differential Equations

1.4.3. Score Match

1.4.3.1. Optimal Score

- Let $L(x, \theta)$ be a likelihood function. The score function is defined as

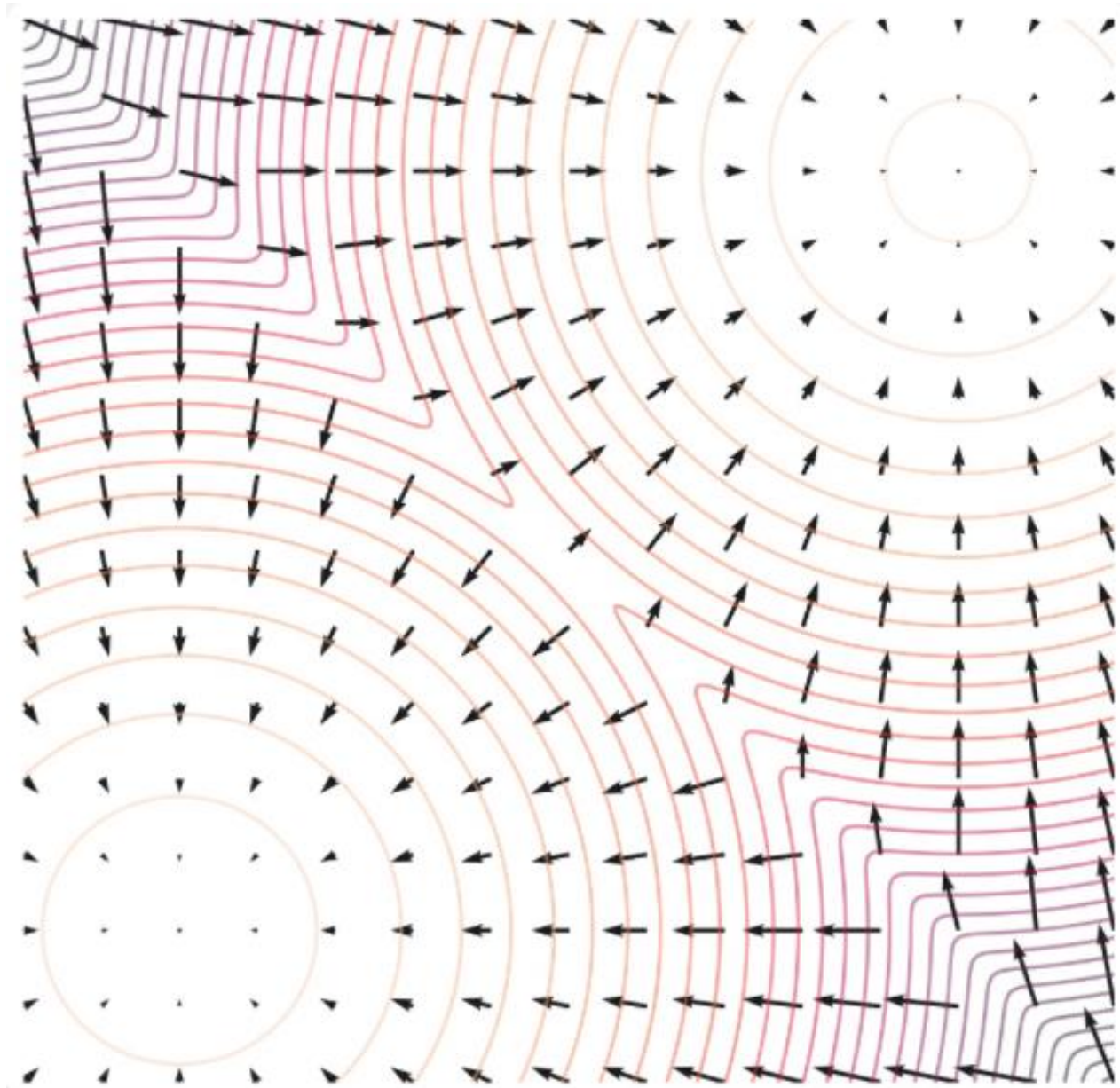
$$U(\theta) = \frac{\partial \log L(x,\theta)}{\partial \theta}.$$



Define Scores in Diffusion Models:

- Consider i.i.d. samples $\{x_1, \ldots, x_n\}, x_i \in R^d$.
- Unnormalized distribution for model: $\tilde{P}_m(x, \theta)$
- Normalized distribution for model: $P_m(x, \theta), P_m(x, \theta) = \frac{\tilde{P}_m(x,\theta)}{Z(\theta)}. \mathrm{Z}(\theta) = \int \tilde{P}_m(x, \theta) d$
- **Score for Data:** $\boldsymbol{S_{data}(x) = \nabla_x \log P_{data}(x)}$
- **Score for Model:** $\boldsymbol{S_m(x, \theta) = \nabla_x \log P_m(x, \theta) = \nabla_x \log \tilde{P}_m(x, \theta)}$

Song et al. 2019, Sliced Score Matching: A Scalable Approach to Density and Score Estimation



1.4.3.2. Fisher Divergence and Score Matching

- **Fisher Divergence between $P_{data}(x)$ and $P_m(x,\theta)$:**

$$L(\theta) = \frac{1}{2} E_{P_{data}(x)} \left[\left\| S_m(x,\theta) - S_{data}(x) \right\|_2^2 \right]$$

- **Fisher divergence is equal to zero, if and only if two densities are equal.**

Since we only have samples and **do not have access to the score function of the data $S_{data}(x)$,** we cannot compute the Fisher divergence.

- However, the Fisher divergence can be transformed to

$J(\theta) = E_{p_{data}(x)} \left[Tr\left(\nabla_x S_m(x,\theta)\right) + \frac{1}{2} \left\| S_m(x,\theta) \right\|_2^2 \right]$, Tr denotes trace of matrix.,

$$\nabla_x S_m(x,\theta) = \begin{bmatrix} \frac{\partial^2 P_m(x,\theta)}{\partial x_1^2} & \cdots & \frac{\partial^2 P_m(x,\theta)}{\partial x_1 \partial x_N} \\ \vdots & \vdots & \vdots \\ \frac{\partial^2 P_m(x,\theta)}{\partial x_N \partial x_1} & \cdots & \frac{\partial^2 P_m(x,\theta)}{\partial x_N^2} \end{bmatrix}$$

Is the Hessian of the log-density function of model.

Sampling Formula for Computing Fisher Divergence:

- $J(\theta, x_1, \ldots, x_N) = \frac{1}{N}\sum_{i=1}^{N}\left[Tr\left(\nabla_x S_m(x_i,\theta)\right) + \frac{1}{2}\left\|S_m(x_i,\theta)\right\|_2^2\right]$

Proof

$$\mathrm{L}(\theta) = \frac{1}{2}E_{P_{data}(x)}\left[\left\|S_m(x,\theta) - S_{data}(x)\right\|_2^2\right]$$

$$= \frac{1}{2}E_{P_{data}(x)}\left[S_{data}^T(x)S_{data}(x) + S_m^T(x,\theta)S_m(x,\theta) - 2\boldsymbol{S_m^T(x,\theta)S_{data}(x)}\right]$$

Since the first term does not depend on the parameter θ, we **only need** to consider the second and third terms in the above equation.

$$\frac{1}{2}E_{P_{data}(x)}[\boldsymbol{S_m^T(x,\theta)S_m(x,\theta)}] = \boldsymbol{\frac{1}{2}\sum_{i=1}^{n}} E_{P_{data}(x)}[\boldsymbol{(S_m(x_i,\theta)^2}]$$

$$\frac{1}{2}E_{P_{data}(x)}[\boldsymbol{2S_m^T(x,\theta)S_{data}(x)}] = \boldsymbol{\sum_{i=1}^{n}\int P_{data}(x)\, S_m\,(x_i,\theta)\,\frac{\partial \log P_{data}(x_i)}{\partial x_i}}$$

$$= \sum_{i=1}^{n}\int S_m(x_i,\theta)\frac{\partial P_{data}(x_i)}{\partial x_i}dx_i$$

$$= \sum_{i=1}^{n} \left[\overset{= 0}{\uparrow} P_{data}(x_i) S_m(x_i, \theta)|_{-\infty}^{\infty} - \int P_{data}(x_i) \frac{\partial S_m(x_i, \theta)}{\partial x_i} dx_i \right]$$

$$= -\sum_{i=1}^{n} E_{P_{data}}(x_i) \frac{\partial^2 \log P_m(x_i, \theta)}{\partial x_i^2}$$

$$= -E_{P_{data}(x)}\left[Tr\left(\nabla_x S_m(x, \theta)\right)\right] \qquad \left(\boldsymbol{\nabla_x S_m(x, \theta) = \frac{\partial^2 P_m(x,\theta)}{\partial x \partial x^T}}\right)$$

which implies $\qquad Tr(A) = a_{11} + \cdots + a_{kk}$

$$\boldsymbol{J(\theta) = E_{p_{data}(x)}\left[Tr\left(\nabla_x S_m(x, \theta)\right) + \frac{1}{2}\left\|S_m(x, \theta)\right\|_2^2\right]}$$

1.4.3.3. Score Estimation for Implicit Distribution

- **Motivation**

Previous score matching is used for parameter estimation in unnormalized models. It can also be used to estimate scores of implicit distributions. They have a tractable sampling process but without a tractable density. For example, distribution $q_\theta(x)$ of generated samples from the generator of a GAN is an implicit distribution.

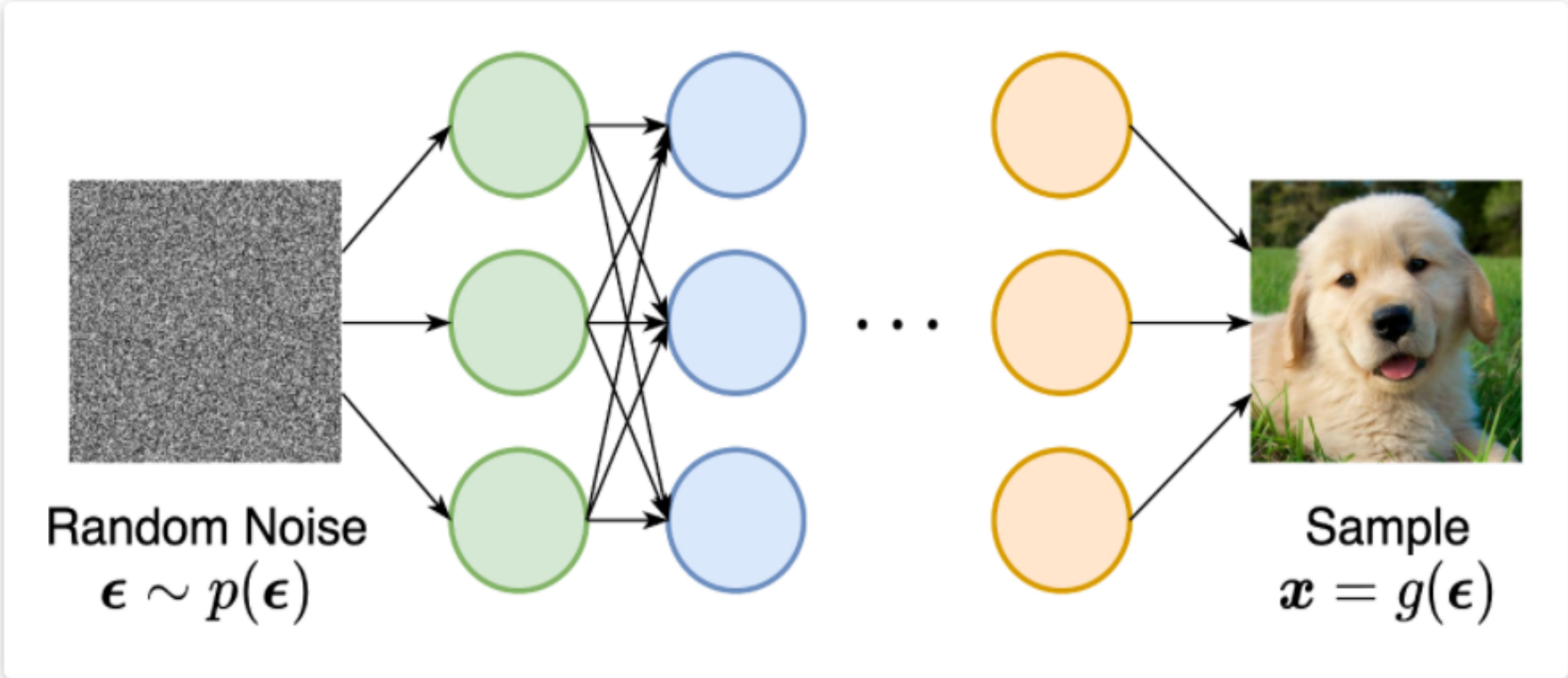
Sometimes, we need to estimate the score function $S_q(x) = \nabla_x \log q_\theta(x)$. For example, optimizing the entropy $H\left(q_\theta(x)\right)$, we need to calculate score $S_q(x)$.

Let $x \sim q_\theta(x)$, $\mathrm{x} = g_\theta(\varepsilon), \varepsilon \sim N(0, I), g_\theta$ is a deterministic function. Then,

$$q_\theta(x) = P_\theta(\varepsilon)\left|\nabla_\varepsilon(g_\theta(\varepsilon))\right|^{-1}$$

$$\nabla_\theta H(q_\theta) = -\nabla_\theta\left\{E_{q_\theta(x)}[\log q_\theta(x)]\right\} = -\nabla_\theta\left\{E_{p(\varepsilon)}[\log q_\theta(g_\theta(\varepsilon))]\right\}$$

$$= -E_{p(\varepsilon)}\left[\boldsymbol{\nabla_x \log q_\theta(x)\Big|_{x=g_\theta(\varepsilon)}}\ \nabla_\theta g_\theta(\varepsilon)\right]$$



GAN is an example of implicit models. It implicitly represents a distribution over all objects that can be produced by the generator network.

https://blog.csdn.net/g11d111

1.4.3.4. Score Estimation

- $\nabla_x \log q_\theta(x)$ is intractable, but can be approximated by score function.

- **Score function:** $h(x, \theta)$: a neural network

- $L\left(\theta, P_{data}(x)\right) = \frac{1}{2} E_{P_{data}(x)} \left[\left\| h(x, \theta) - \nabla_x \log q_\theta(x) \right\|_2^2 \right]$

$$L\left(\theta, P_{data}(x)\right) = 0 \leftrightarrow h(x, \theta) = \nabla_x \log q_\theta(x)$$

1.4.3.5. Sliced Score Matching

- **Motivation**

  To avoid difficulty in computing the trace of the Hessian of a log-density function $\nabla_x^2 \log \tilde{P}_m$, Consider projecting score functions $S_d(x)$ and $S_m(x,\theta)$ On to some random direction v and compare their average difference along that random direction

- **Objective Function for Sliced Score Matching**

$$L(\theta, P_v) = \frac{1}{2} E_{P_v} E_{P_d} \left[\left(V^T S_m(x,\theta) - V^T S_d(x) \right)^2 \right] \qquad (4.1)$$

  where $V \sim P_v$ and $x \sim P_d$ are independent, $E_{P_v}[VV^T]$ is positive definite and $E_{P_v}[\|V\|_2^2] < \infty$ is finite.

- **Reduction**

  To eliminate the dependence of $L(\theta, P_v)$ on S_d, $L(\theta, P_v)$ can be reduced to

$$L(\theta, P_v) = J(\theta, P_v) + C \qquad (4.2)$$

  where $\boldsymbol{J(\theta, P_v) = E_{P_v} E_{P_d} [V^T \nabla_x S_m(x,\theta) V + \frac{1}{2} \left(V^T S_m(x,\theta) \right)^2]}$, C is a constant.

Proof

Let $C = \frac{1}{2} E_{P_V} E_{P_d} \left[\left(V^T S_d(x) \right)^2 \right]$ (4.3)

Note

$$-E_{P_V} E_{P_d} \left[V^T S_m(\theta, x) S_d^T(x) V \right] = -E_{P_V} \left[\int P_d(x) V^T S_m(\theta, x) \left(\frac{\partial \log P_d(x)}{\partial x} \right)^T V dx \right]$$

$$= E_{P_V} \left[-V^T S_m(\theta, x) P_d^T(x) V \Big|_{-\infty}^{\infty} + \int V^T \nabla_x S_m(x) P_d^T(x) V dx \right] \text{ (part by integration)}$$

$= 0$

$$= E_{P_V} E_{P_d} \left[V^T \nabla_x S_m(x) V \right] \quad (4.4)$$

Substituting equations (4.3) and (4.4) into equatiom (4.1) yields equation (4.2).

1.4.3.6. Sampling Formula of Sliced Score

Let $X_1^N = \{x_1, \dots, x_N\}$ and $V_{11}^{NM} = \{V_{ij}\}_{1 \le i \le N, 1 \le j \le M}$

Define Sampling Formula of $\boldsymbol{J(\theta, P_v)}$:

$$\hat{\boldsymbol{J}}\left(\boldsymbol{\theta}, \boldsymbol{X}_{\boldsymbol{1}}^{\boldsymbol{NN}}, \boldsymbol{V}_{\boldsymbol{11}}^{\boldsymbol{NM}}\right) = \frac{\boldsymbol{1}}{\boldsymbol{N}} \frac{\boldsymbol{1}}{\boldsymbol{M}} \sum_{\boldsymbol{i-1}}^{\boldsymbol{N}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{M}} \left[\boldsymbol{V}_{\boldsymbol{ij}}^{\boldsymbol{T}} \boldsymbol{\nabla}_{\boldsymbol{x}} \boldsymbol{S}_{\boldsymbol{m}}(\boldsymbol{x}_{\boldsymbol{i}}, \boldsymbol{\theta}) \boldsymbol{V}_{\boldsymbol{ij}} + \frac{\boldsymbol{1}}{\boldsymbol{2}} \left(\boldsymbol{V}_{\boldsymbol{j}}^{\boldsymbol{T}} \boldsymbol{S}_{\boldsymbol{m}}(\boldsymbol{x}_{\boldsymbol{i}}, \boldsymbol{\theta}) \right)^{\boldsymbol{2}} \right] \tag{4.5}$$

1.4.3.7. Estimator with Reduced Variance

Assume that $V \sim N(0, I)$. Then, $E_{P_V}[VV^T] = I$

$$E_{P_V}\left[\left(V^T S_m(x,\theta)\right)^2\right] = E_{P_V}\left[Tr\left(\left(V^T S_m(x,\theta)\right)^2\right)\right]$$

$$= E_{P_V}\left[Tr\left(\left(V^T S_m(x,\theta) S_m(x,\theta)^T V\right)\right]\right.$$

$$= E_{P_V}[Tr(S_m(x,\theta) S_m(x,\theta)^T VV^T)]$$

$$= Tr(S_m(x,\theta) S_m(x,\theta)^T \; E_{P_V}[VV^T])$$

$$= Tr(S_m(x,\theta) S_m(x,\theta)^T) = \|S_m(x,\theta)\|_2^2 \qquad (4.6)$$

Substituting equation (4.6) into equation (4.2) yields

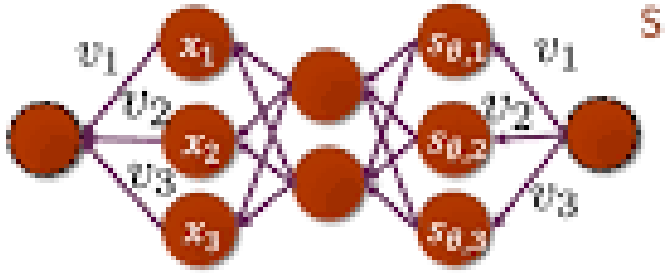
$$\boldsymbol{J_{vr}(\theta, P_v) = E_{P_v} E_{P_d}\left[V^T \nabla_x S_m(x,\theta) V + \frac{1}{2}\|S_m(x,\theta)\|_2^2\right]} \qquad (4.7)$$

$$\hat{J}_{vr}\left(\theta, X_1^{NN}, V_{11}^{NM}\right) = \frac{1}{N}\frac{1}{M}\sum_{i-1}^{N}\sum_{j=1}^{M}\left[V_{ij}^{T}\nabla_x S_m(x_i,\theta)V_{ij} + \frac{1}{2}\|S_m(x_i,\theta)\|_2^2\right] \quad (4.8)$$

Computing Jacobian-vector products is scalable

$$\mathbf{v}^\intercal \nabla_{\mathbf{x}} \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})\mathbf{v} = \nabla_{\mathbf{x}}[\mathbf{v}^\intercal \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x})]\mathbf{v}$$

$\mathbf{v}^\intercal \nabla_{\mathbf{x}}(\mathbf{v}^\intercal \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}))$



One Backprop!
Sliced Score Matching
is scalable

Song*, Garg*, Shi, Ermon. "Sliced Score Matching: A Scalable Approach to Density and Score Estimation." UAI 2019.

TechBeat
Stanford University

song et al. 2019, Sliced Score Matching: A Scalable Approach to Density and Score Estimation

1.4.3.8. Algorithm for Sliced Score Matching

Step 1: Input $\widetilde{\boldsymbol{P}}_{\boldsymbol{m}}(.;\boldsymbol{\theta}), \boldsymbol{x}, \boldsymbol{V}$

Step 2: $\boldsymbol{S_m(x;\theta)} \leftarrow \boldsymbol{\nabla_x} \log \widetilde{\boldsymbol{P}}_m(x,\theta)$

Step 3: $\boldsymbol{V^T \nabla_x S_m(x,\theta)} \leftarrow \boldsymbol{\nabla_x}\left(\boldsymbol{V^T S_m(x,\theta)}\right)$

Step 4: $\mathbf{J} \leftarrow \frac{\mathbf{1}}{\mathbf{2}}\left(\boldsymbol{V^T S_m(x,\theta)}\right)^{\mathbf{2}}$ **(or** $\frac{\mathbf{1}}{\mathbf{2}}\|\boldsymbol{S_m(x,\theta)}\|_{\mathbf{2}}^{\mathbf{2}}$**)**

Step 5: $\boldsymbol{J} \leftarrow \boldsymbol{J} + \boldsymbol{V^T \nabla_x S_m(x,\theta)}\mathbf{V}$

Step 6: Output $\boldsymbol{J}$

1.4.3.9. Consistency of Estimator

- **Data Distribution $P_d(x)$**
- **Model Distribution $P_m(x, \theta)$**

$$P_m(x, \theta^*) = P_d(x)$$

$$J(\theta, P_V) = 0 \Leftrightarrow \theta = \theta^*$$

- Let

$$\hat{\theta}_{N,M} = \underset{\theta}{\operatorname{argmin}} \hat{J}(\theta, X_1^N, V_{11}^N)$$

$$\hat{\theta}_{N.M} \to \theta$$

- Then, for a fixed M, under some regular conditions

$$\hat{\theta}_{N,M} \xrightarrow{p} \theta^*, \qquad N \to \infty$$

1.4.3.10. Asymptotic Normality

Under some assumptions, we have

$$\sqrt{N}(\widehat{\boldsymbol{\theta}}_{N,m} - \boldsymbol{\theta}^*) \xrightarrow{d} \mathbf{N}(\mathbf{0}, \boldsymbol{\Sigma})$$

where

$$\Sigma = \left(\nabla_\theta^2 J(\theta^*)\right)^{-1} \left(\sum_{1 \le i,j \le D} V_{ij}\right) \left(\nabla_\theta^2 J(\theta^*)\right)^{-1}$$

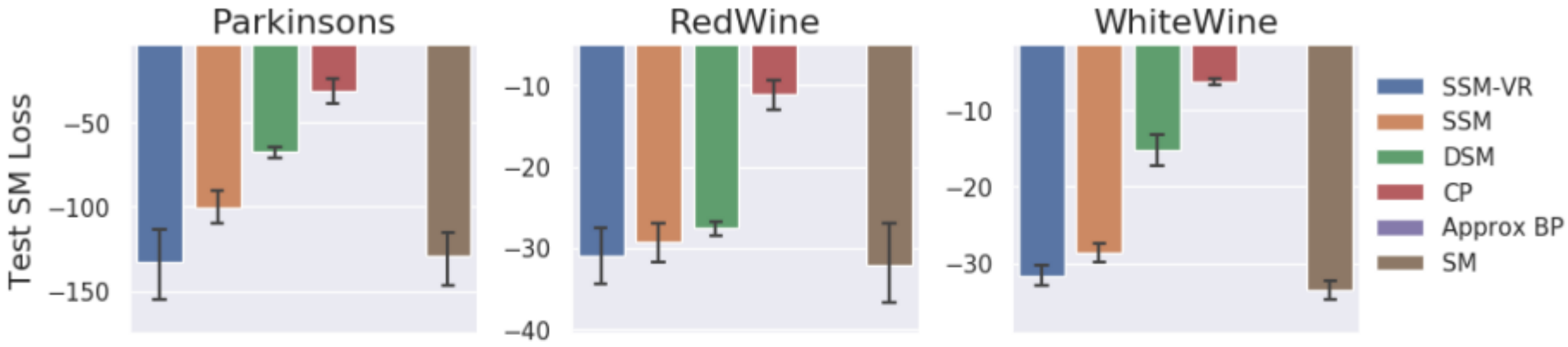


Figure 1: SM loss after training DKEF models on UCI datasets with different loss functions; lower is better. Results for approximate backprapogation are not shown because losses were larger than 10^9.

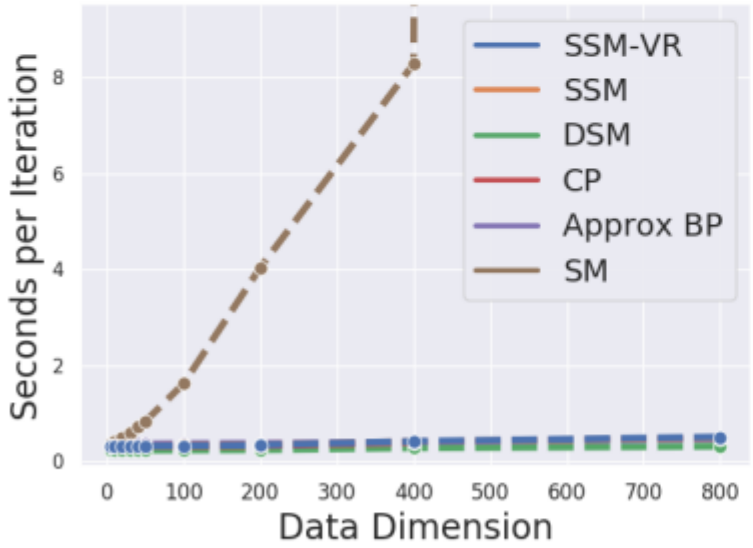


Figure 2: SM performance degrades linearly with the data dimension, while efficient approaches have relatively similar performance.

	Test SM Loss	Test LL
MLE	-579	**-791**
SSM-VR	**-8054**	-3355
SSM	-2428	-2039
DSM ($\sigma = 0.10$)	-3035	-4363
DSM ($\sigma = 1.74$)	-97	-8082
CP	-1694	-1517
Approx BP	-48	-2288

Table 1: Score matching losses and log-likelihoods for NICE models on MNIST. $\sigma = 0.1$ is by grid search and $\sigma = 1.74$ is from the heuristic of Saremi et al. (2018).

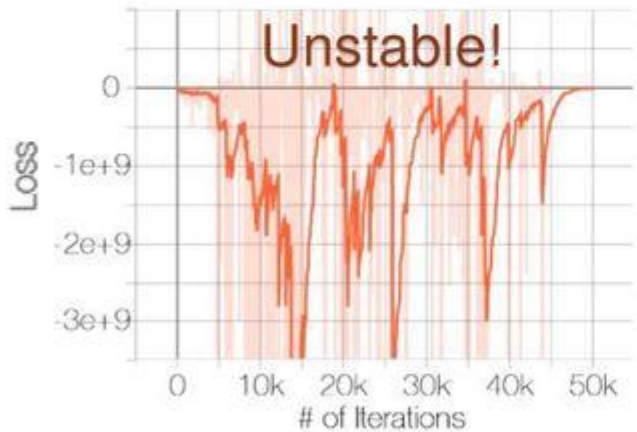
1.4.3.11. Challenges of score-based generative modeling

- Data are located in low dimensional manifold, not in the entire database, including high dimensional ambient space.
- Since the score $\nabla_x \log P_{data}(x)$ may be taken in the ambient space, it is undefined when x is only located in a low dimensional manifold.
- Score matching algorithm provides a consistent score estimator only when the support of the data distribution is in the entire space.
- We then produce samples using Langevin dynamics, which approximately works by gradually moving a random initial sample to high density regions along the (estimated) vector field of scores

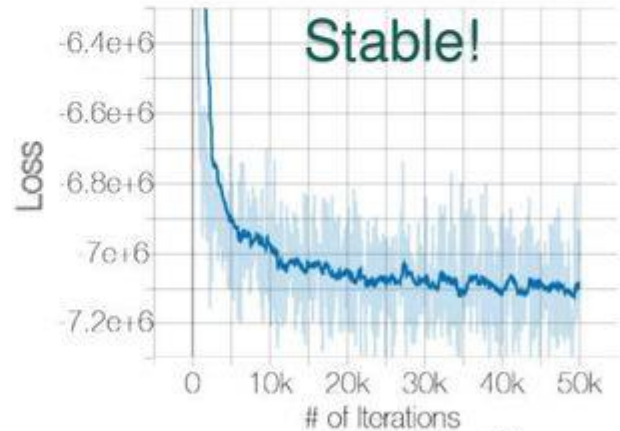
Son and Ermon, 2020. Generative Modeling by Estimating Gradients of the Data Distribution

Adding Noise to Data for Well-Defined Scores

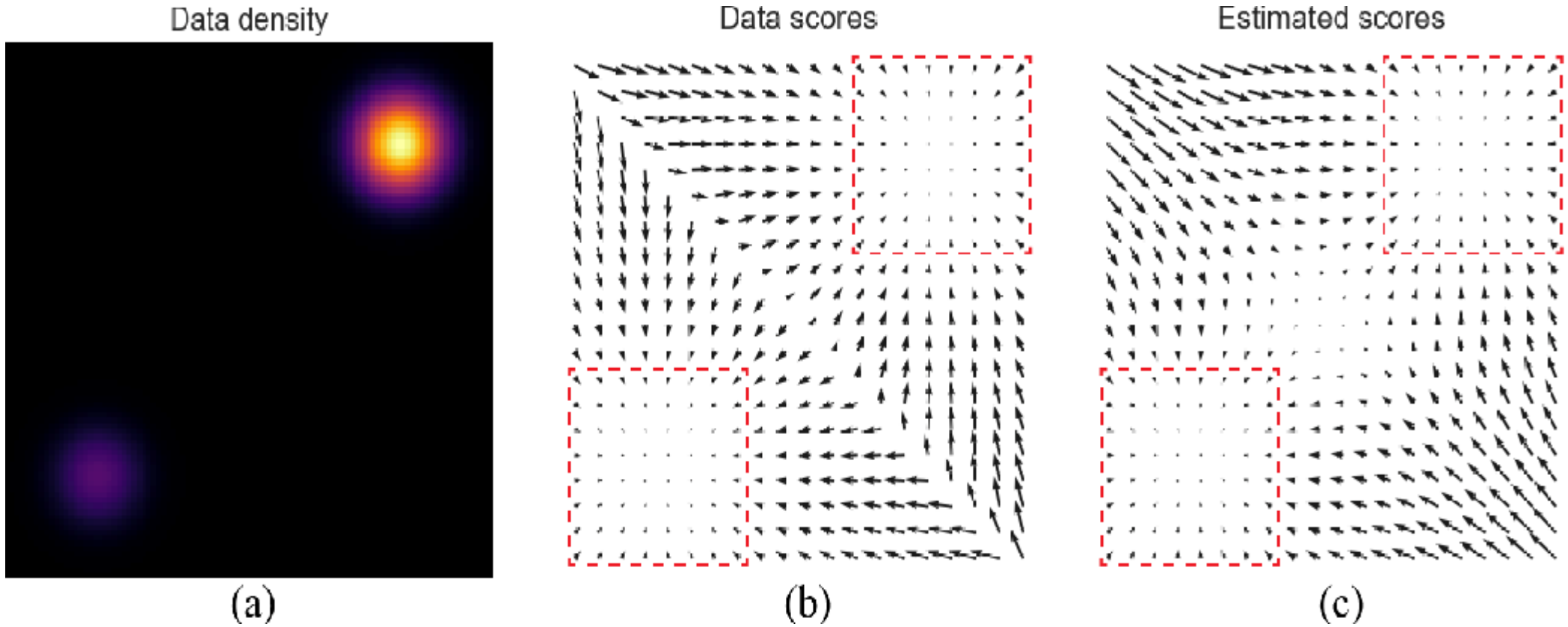
- Scores can be undefined when
  - The support of data distribution is on a low-dimensional manifold
  - The data distribution is discrete
- Solution: adding noise



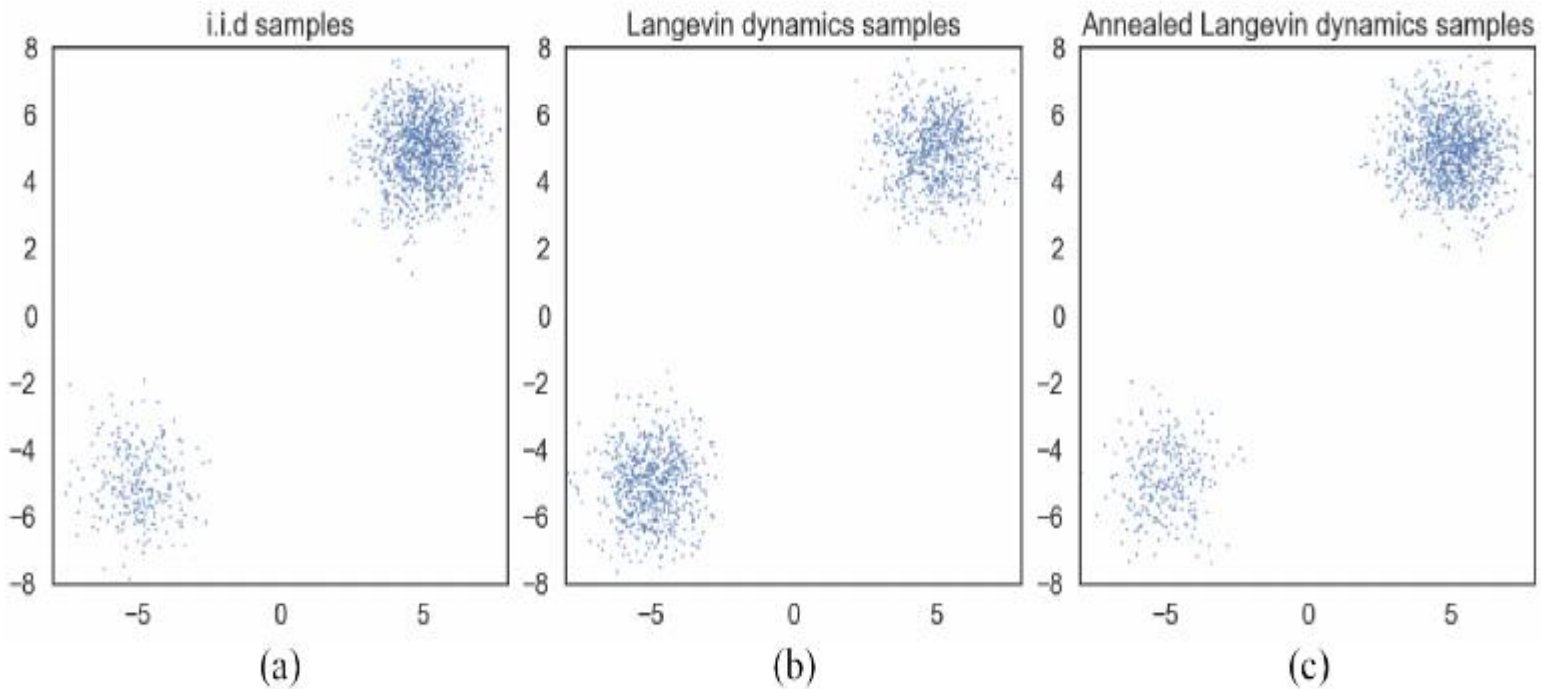
Data unperturbed



Data perturbed with $\mathcal{N}(0; 0.0001)$



$$P_{data}(x) = \frac{1}{5}N\left((-5,-5), I\right) + \frac{4}{5}N((5,5), I)$$


i.i.d samples
Langevin dynamics samples
Annealed Langevin dynamics samples
8
6
4
2
0
−2
−4
−6
−8
−5
0
5
(a)
(b)
(c)

1.4.3.12. Denoising score matching

Step 1:

To avoid computing cost of $\boldsymbol{Tr\left(\nabla_x S_m(x,\theta)\right)}$, we first perturb the data x with a pre-specified noise distribution $q_\sigma(\tilde{x}|x)$.

Step 2:

Then, employ score matching to estimate the score of the perturbed data distribution

$$q_\sigma(\tilde{x}) = \int q_\sigma(\tilde{x}|x) P_{data}(x) dx$$

Step 3:

The equivalent objective is given by

$$\frac{1}{2} E_{q_\sigma(\tilde{x}|x)P_{data}(x)}\left[\left\| S_\theta(\tilde{x}) - \nabla_{\tilde{x}} \log q_\sigma(\tilde{x}|x) \right\|_2^2\right]$$

Note that $\nabla_{\tilde{x}} \log q(\tilde{x}) = \nabla_{\tilde{x}}[\log q_\sigma(\tilde{x}|x) + \log P_{data}(x)] = \nabla_{\tilde{x}} \log q_\sigma(\tilde{x}|x)$

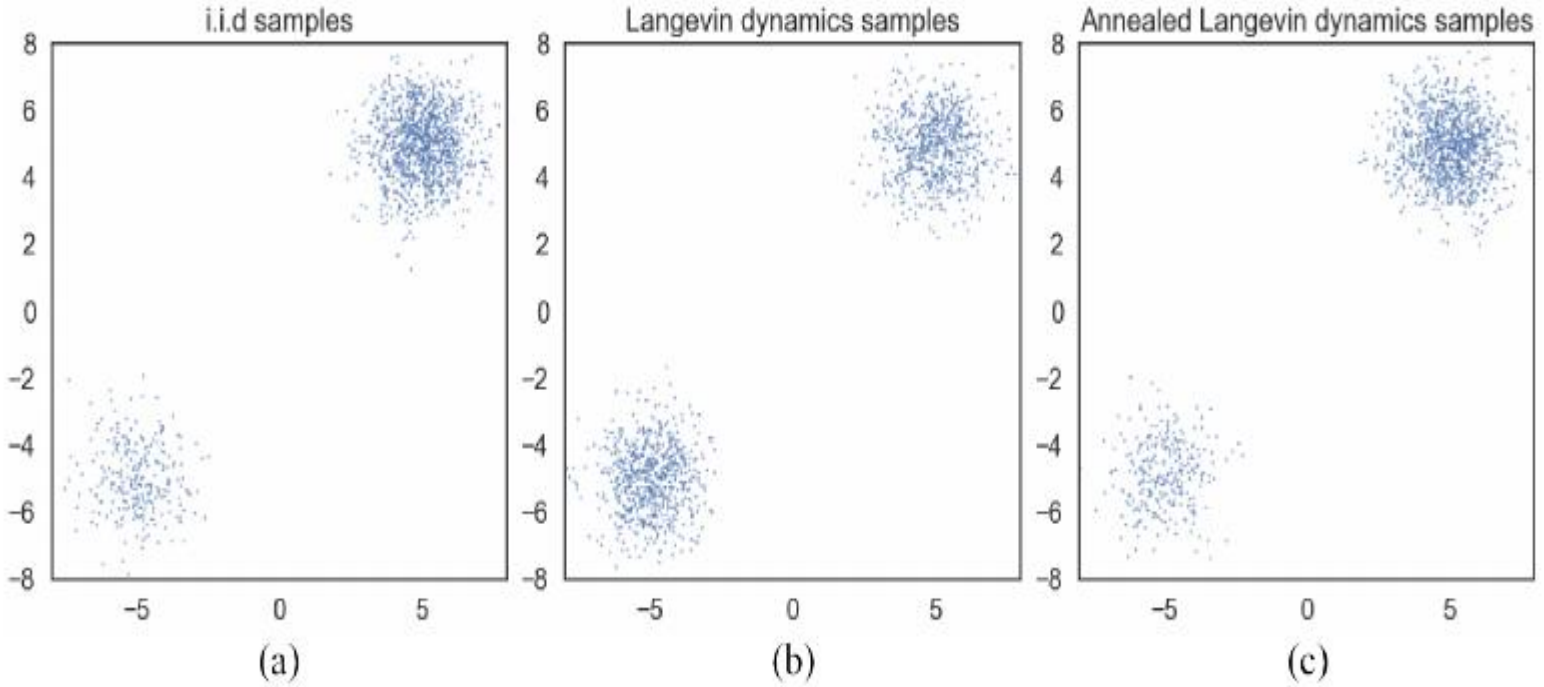
Step 4:

The optimal score network minimizes the above objective functions satisfies

$$S_{\theta^*}(x) = \nabla_x \log q_\sigma(x) \approx \nabla_x \log P_{data}(x)$$

when the noise is small such that

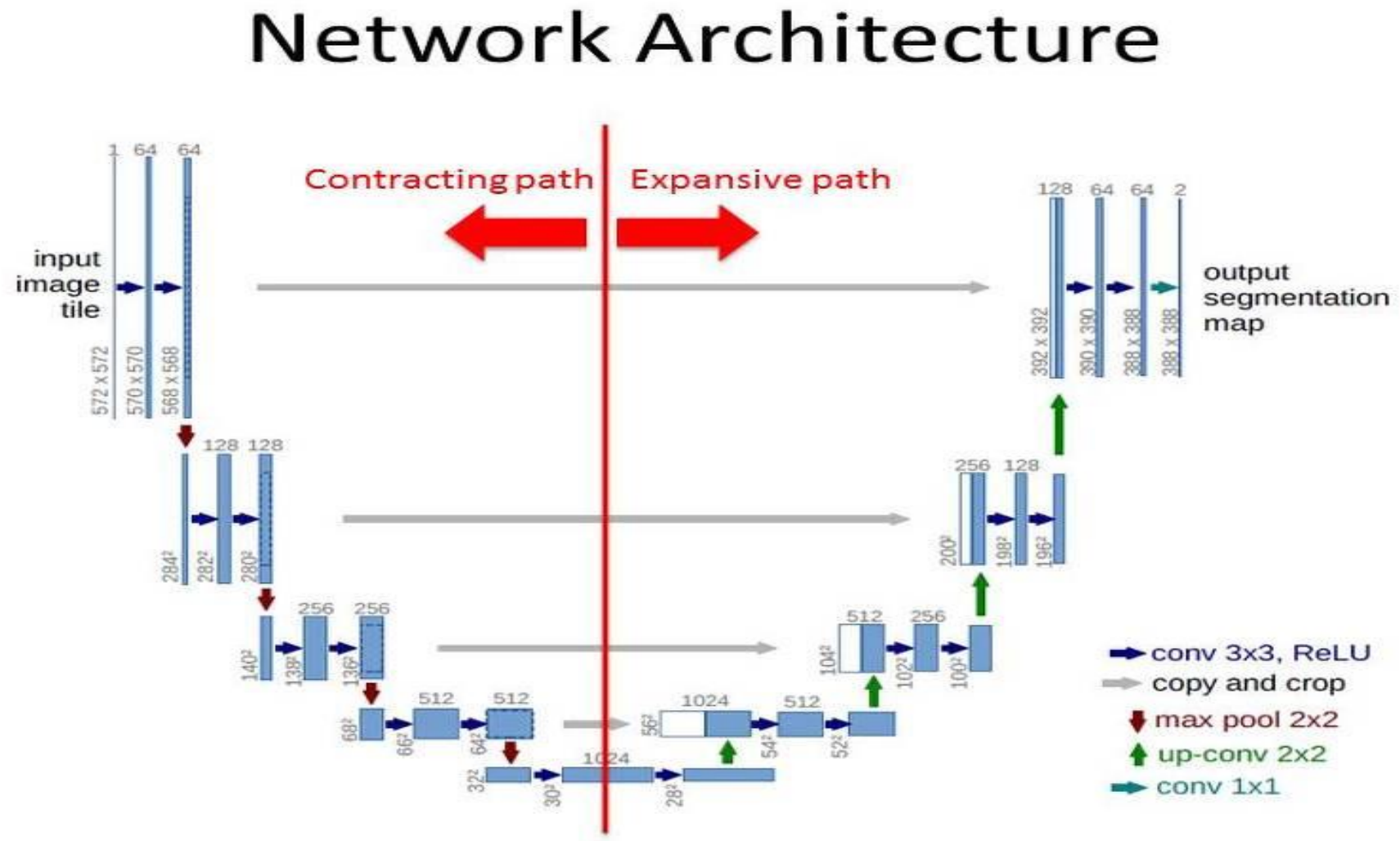
$$q_\sigma(x) \approx P_{data}(x)$$



Built upon this intuition, we propose to improve score-based generative modeling by

- **1) perturbing the data using various levels of noise; and**
- **2) simultaneously estimating scores corresponding to all noise levels by training a single conditional score network**.

1.4.3.13. Generative Model by Score Matching



Ronneberger et al. 2015. U-Net Convolutional Networks for Biomedical Image Segmentation

- **Noise Conditional Score Network $S_\theta(x, \sigma)$**

Define

$$\{\sigma_i\}_{i=1}^{L}, \frac{\sigma_1}{\sigma_2} = \cdots = \frac{\sigma_{L-1}}{\sigma_L} > 1$$

$$q_\sigma(\widetilde{x}) = \int P_{data}(x) N(\tilde{x} \,|x, \sigma^2 I) dx$$

- **Objective Function**

$$\nabla_{\tilde{x}} \log q_\sigma(\tilde{x}|x) = -\frac{\tilde{x} - x}{\sigma^2}$$

$$l(\theta, \sigma) = \frac{1}{2} E_{P_{data}(x)} E_{\tilde{x} \sim N(x, \sigma^2)} \left[\left\| \boldsymbol{S_\theta}(\tilde{x}, \boldsymbol{\sigma}) + \frac{\tilde{x} - x}{\sigma^2} \right\|_2^2 \right]$$

$$l\left(\theta, \{\sigma_i\}_{i=1}^{L}\right) = \frac{1}{L} \sum_{i=1}^{L} \lambda(\sigma_i) l(\theta, \sigma_i)$$

$$\boldsymbol{S_\theta}(\tilde{x}, \boldsymbol{\sigma}) = \boldsymbol{\nabla_{\tilde{x}}} \log q_\sigma(x), \lambda(\sigma_i) = \boldsymbol{\sigma_i^2}$$

1.4.3.14. Stochastic Gradient and Langevin MCMC

(Algorithm 1)

Step 1: **Solve**

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \sum_{i=1}^{n} \sigma_i^2 E_{P_{data}(x)} E_{P_{\sigma_i}} \left[\left\| S_\theta(\tilde{x}, \sigma_i) - \nabla_{\tilde{x}} \log P_{\sigma_i}(\tilde{x}|x) \right\|_2^2 \right]$$

Step 2:

Similar to stochastic gradient algorithm, we use Langevin MCMC to get a sample for each P_{σ_i}(x) sequentially

$$x_i^m = x_i^{m-1} + \varepsilon_i \nabla_x \log P_{\sigma_i}\left(x_i^{m-1}\right) + \sqrt{2\varepsilon_i} z_i^m$$

$$= x_i^{m-1} + \varepsilon_i S_{\theta^*}\left(x_i^{m-1}, \sigma_i\right) + \sqrt{2\varepsilon_i} z_i^m, m = 1, \ldots, M, \varepsilon_i > 0, z_i^m \text{ is standard normal.}$$

Step 3: Repeat for $i = N, N-1, \ldots, 1$, with $x_N^0 \sim N\left(x \middle| 0, \sigma_{max}^2 I\right), x_i^0 = x_{i+1}^M$ for $i < N$

As $M \to \infty$ and $\varepsilon_i > 0$, for all i, x_1^M becomes an exact sample from $P_{\sigma_{min}} \approx P_{data}(x)$.

1.4.3.15. Stochastic Gradient and Langevin MCMC (Algorithm 2)

- **Input** $\{\sigma_i\}_{i=1}^{L}, \varepsilon, T$
- **Step 1** Initialize $\tilde{x}_0$
- **Step 2** For $i \leftarrow 1\ to\ L$ do

$$\alpha_i \leftarrow \varepsilon \frac{\sigma_i^2}{\sigma_L^2}$$

- **Step 3** for $t \leftarrow 1\ to\ T$ **do**

$$Draw\ z_t \sim N(0, I)$$

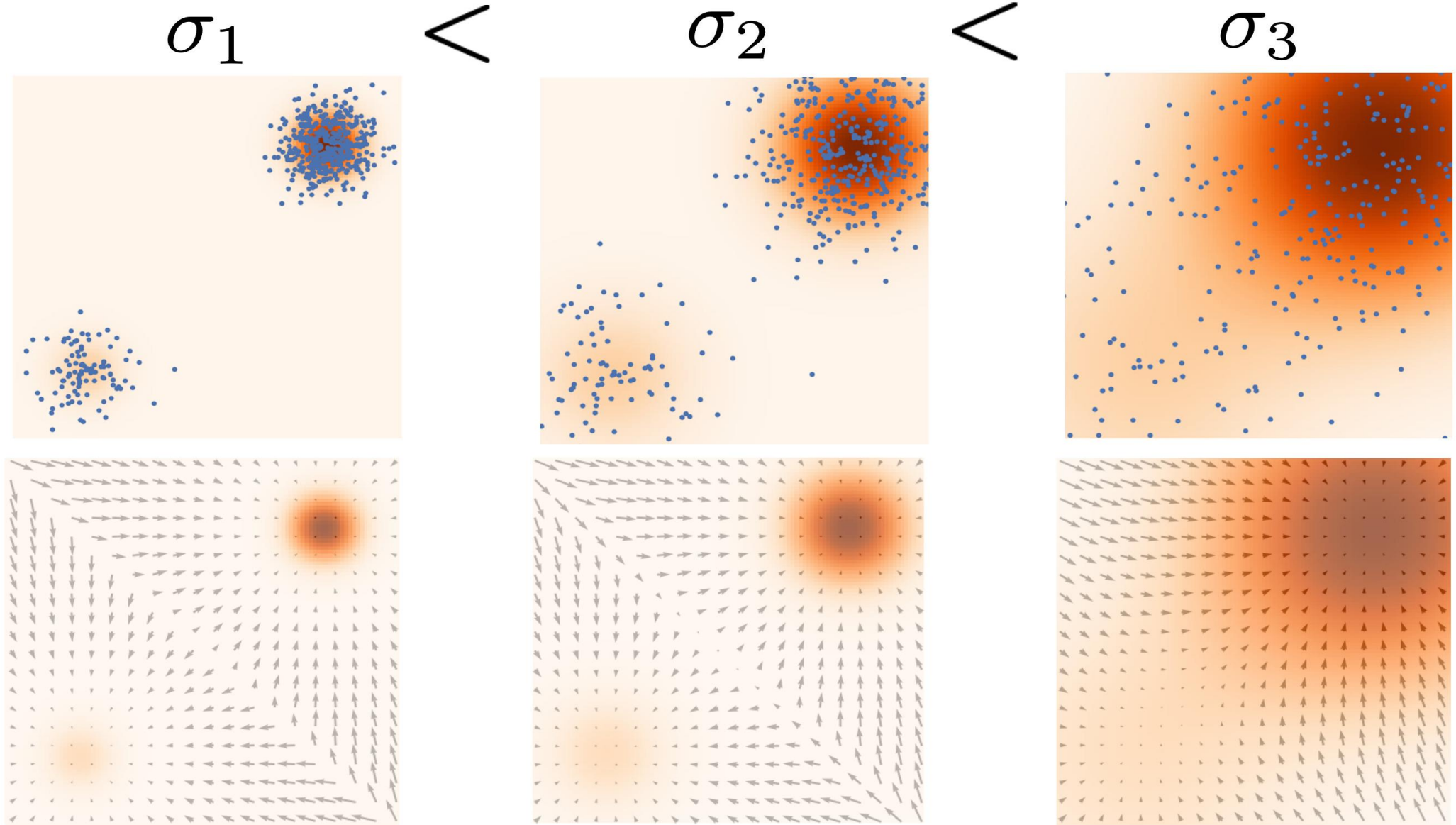
$$\tilde{x}_t \leftarrow \tilde{x}_{t-1} + \frac{\alpha_i}{2} S_\theta(\tilde{x}_{t-1}, \sigma_i) + \sqrt{\alpha_i} z_t$$

end for

$$\tilde{x}_0 \leftarrow \tilde{x}_T$$

end for

- **Step 4** Return $\tilde{x}_T$


σ_1
<
σ_2
<
σ_3

Model	Inception	FID
CIFAR-10 Unconditional		
PixelCNN [48]	4.60	65.93
PixelIQN [36]	5.29	49.46
EBM [10]	6.02	40.58
DCGAN [37]	6.40	37.11
WGAN-GP [15]	6.50	36.4
EBM (ensemble) [10]	6.78	38.2
MoLM [38]	7.90	**18.9**
SNGAN [31]	8.22	21.7
NCSN (Ours)	**8.91**	25.32
CIFAR-10 Conditional		
Improved GAN [41]	8.09	-
EBM [10]	8.30	37.9
SNGAN [31]	8.59	25.5
BigGAN [4]	**9.22**	**14.73**

Table 1: Inception and FID scores for CIFAR-10

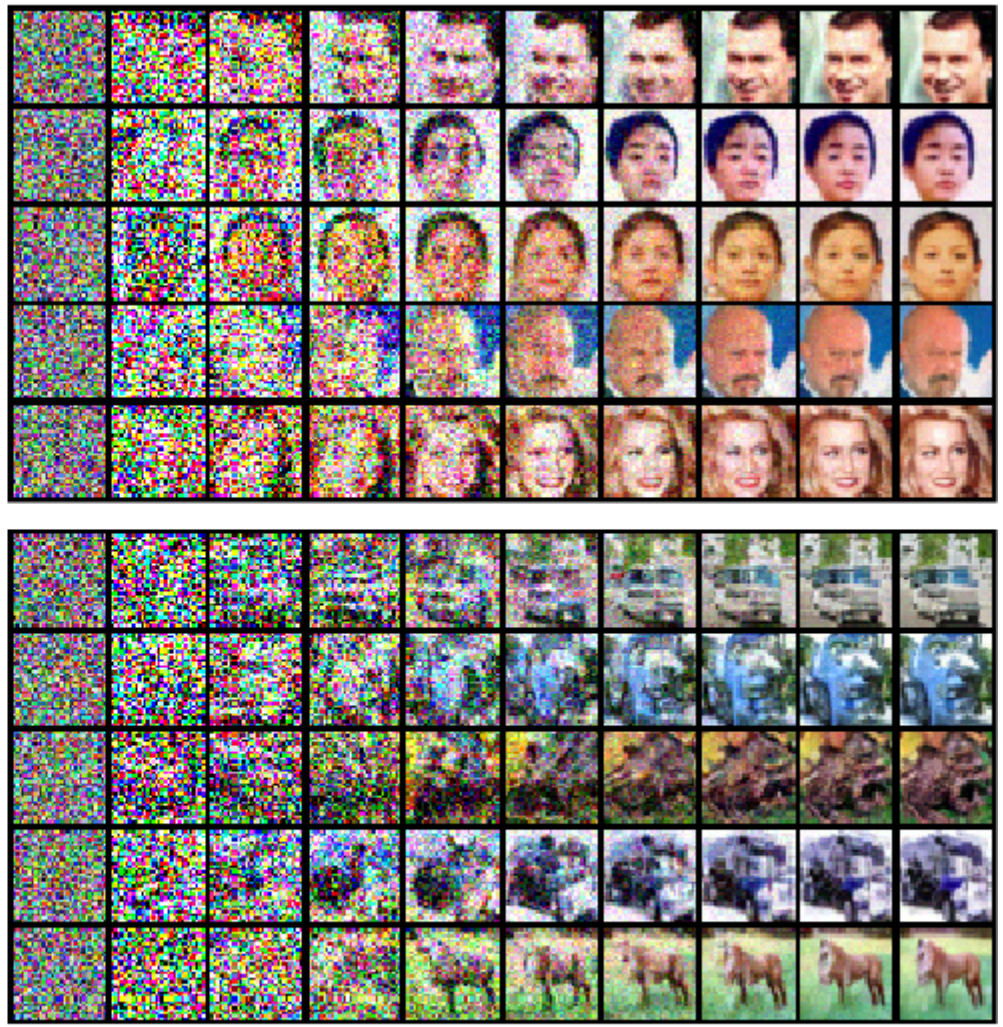
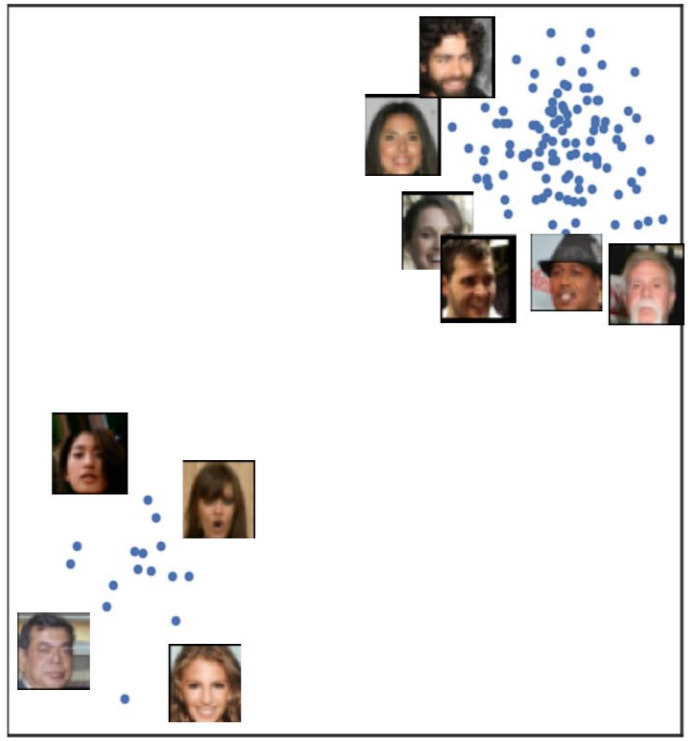
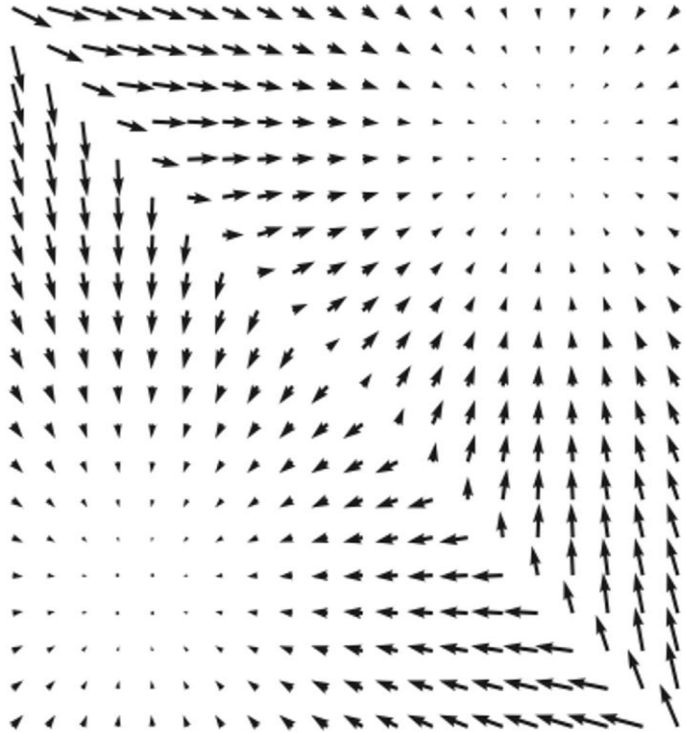


Figure 4: Intermediate samples of annealed Langevin dynamics.



score matching



Langevin dynamics

Data samples

Scores

New samples

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

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