Generative Modeling and Reinforcement Learning: Energy, Reward, and Value

Sangwoong Yoon

Graduate School of AI, UNIST

About me: Sangwoong Yoon

Assistant Prof. @ **UNIST Graduate School of AI**

(Prev.) Postdoc @ **University College London** (UCL)
(Prev.) AI Research Fellow @ **Korea Institute for Advanced Study** (KIAS)

Education

- Ph.D., Mechanical Engineering, Seoul National University (SNU)
- M.S., Neuroscience, SNU
- B.S., Chemical & Biological Engineering, SNU

Industry Experience (as a ML Researcher/Engineer)

- Amazon (Research Internship)
- Kakao Brain (Research Internship)
- Haezoom Inc. & Saige Inc. (Full-time ML Researcher/Engineer)

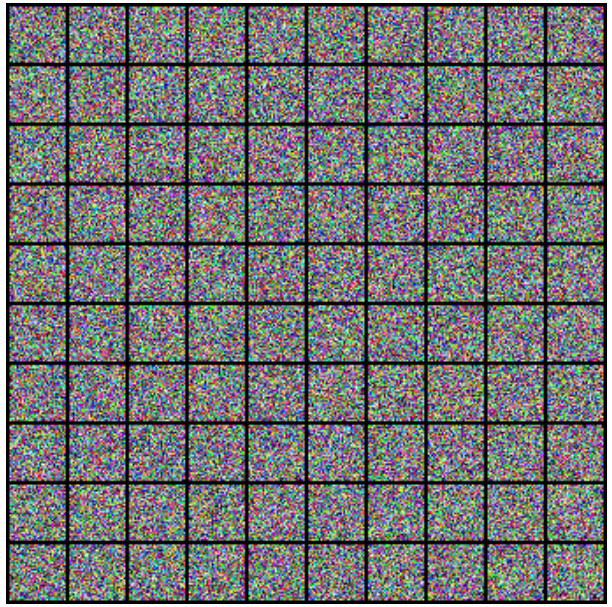
Two Kinds of Learning

Learning by watching

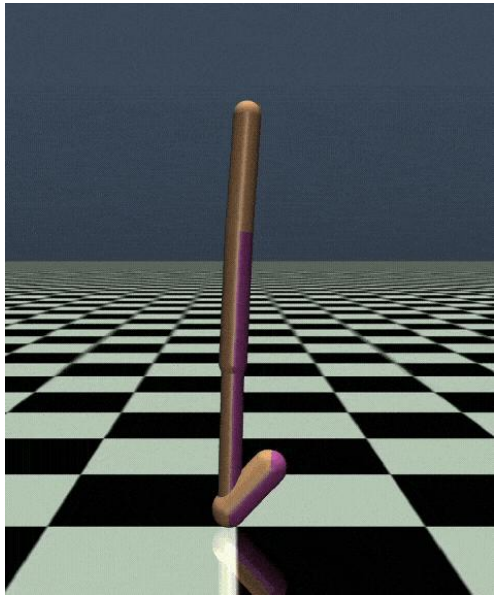
Learning by doing

https://www.smallsmart.co.uk/products/little-dutch-wooden-music-mobile?srsltid=AfmBOoq6oD8l-JaGooghlXUqDEpI4n-JYvpAw1069NumxTjSjccSj675
https://peekaby.com/article/%ED%94%BC%EC%B9%B4%EB%B9%84%EC%9D%98-%EC%A1%B0%EC%96%B8/8/3424/

Two Kinds of Machine Learning



Generative Modeling



Reinforcement Learning

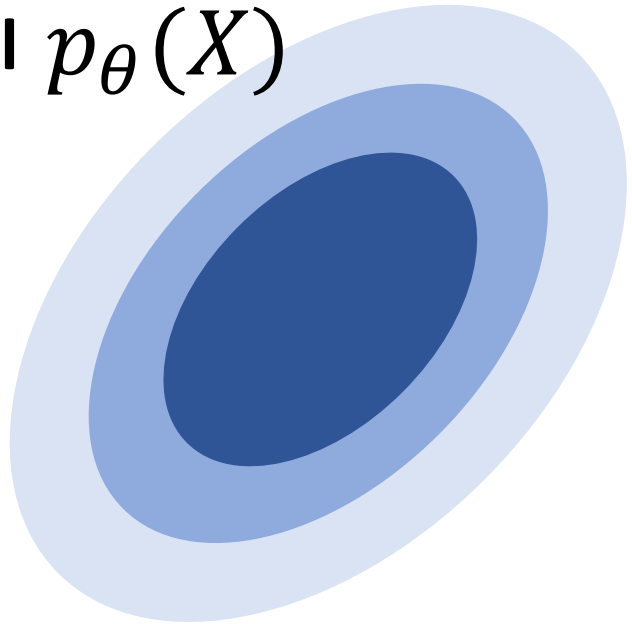
https://github.com/ermongroup/ncsn
https://medium.com/@jer_zhang/a-survey-of-reinforcement-learning-techniques-for-2d-and-3d-bipedal-locomotion-2f82f455396a

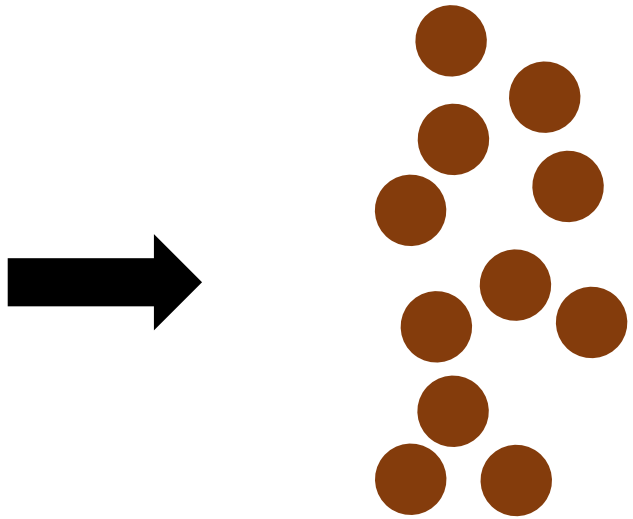
One Kind of Machine Learning

Probabilistic Machine Learning

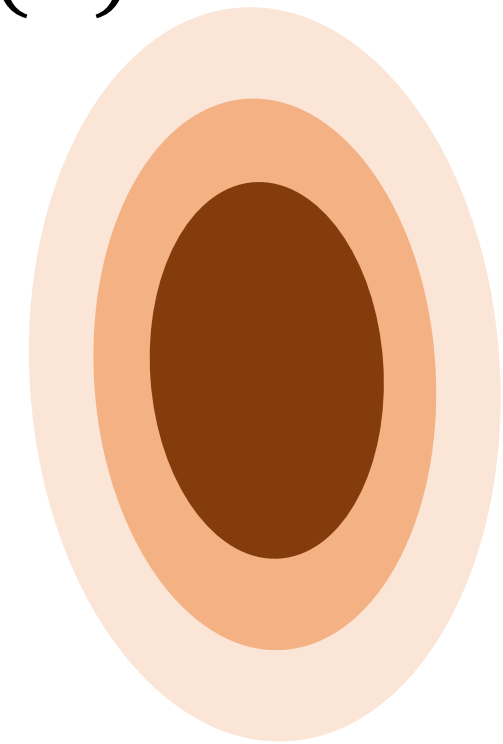
Target $q(X)$

Model $p_\theta(X)$





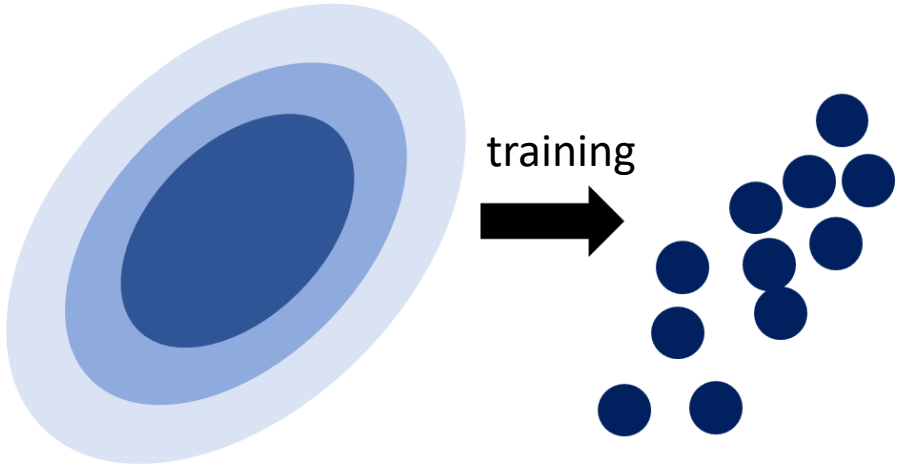
or



$$\min_\theta D(p_\theta, q)$$

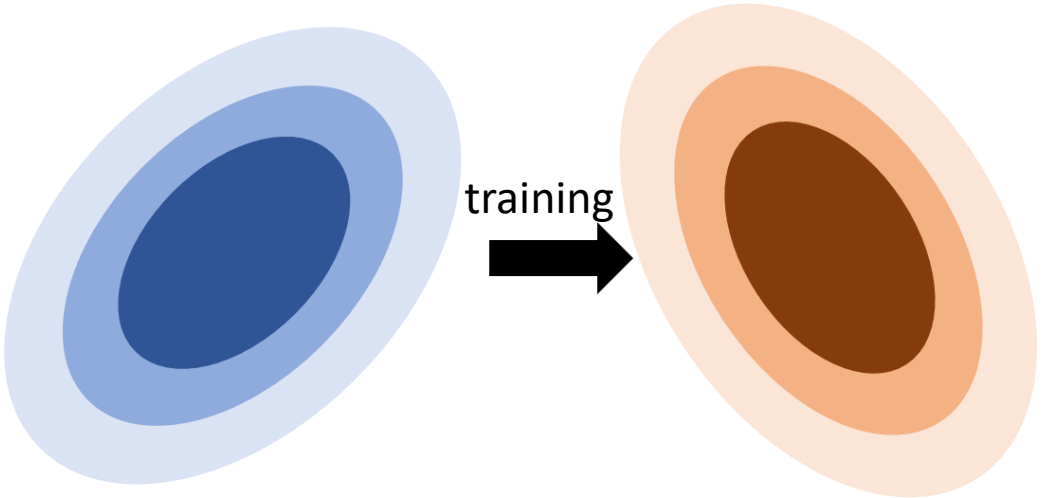
Generative Modeling vs RL

Generative Modeling



p_{model} p_{data}

RL



π_{policy} $\pi^*(x) \propto \exp\big(R(x)\big)$

My Research Journey

Generative Modeling **Reinforcement Learning**

Energy-Based Models (EBMs)
Non-parametric prob models

RL for Diffusion

RL for LLM

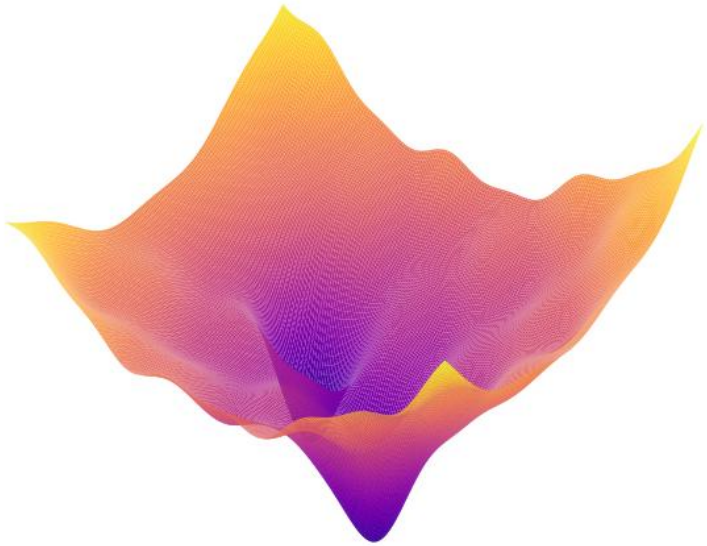
PhD (Prof. Chongwoo Park)

Post-doc 1 (Prof. Yung-Kyun Noh& Prof. Dohyun Kwon)

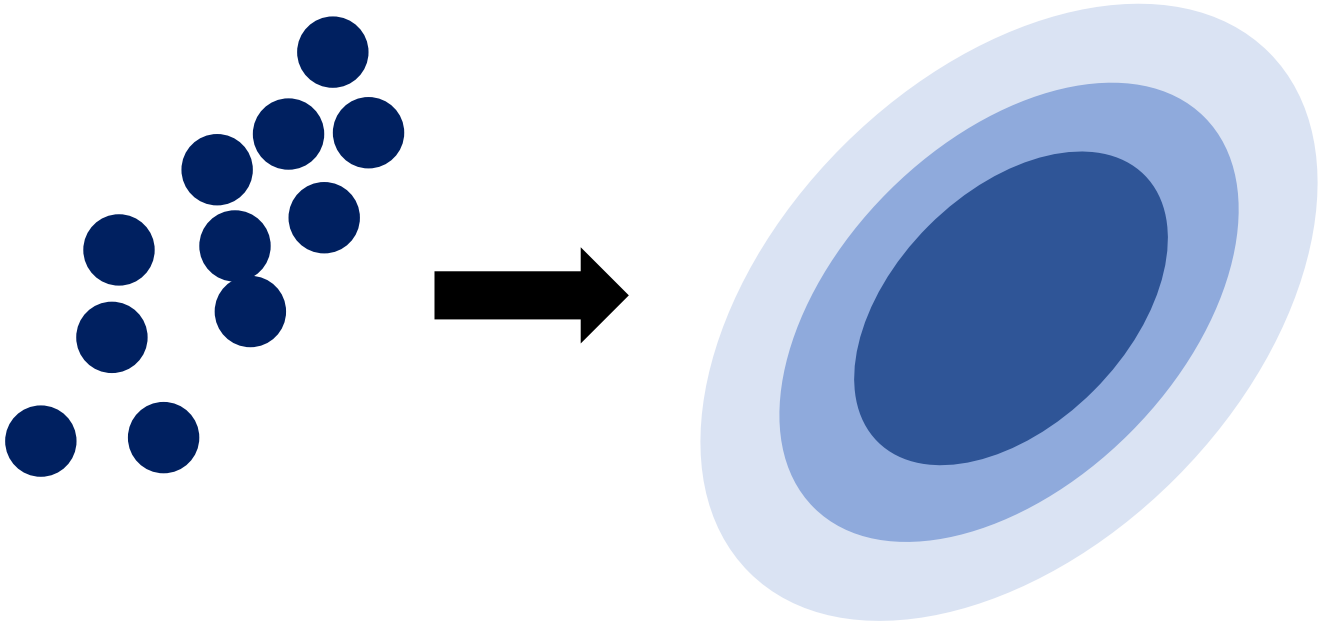
Post-doc 2 (Prof. Ilija Bogunovic)

Energy ➡ Reward ➡ Value

Energy



Generative Modeling



X_i

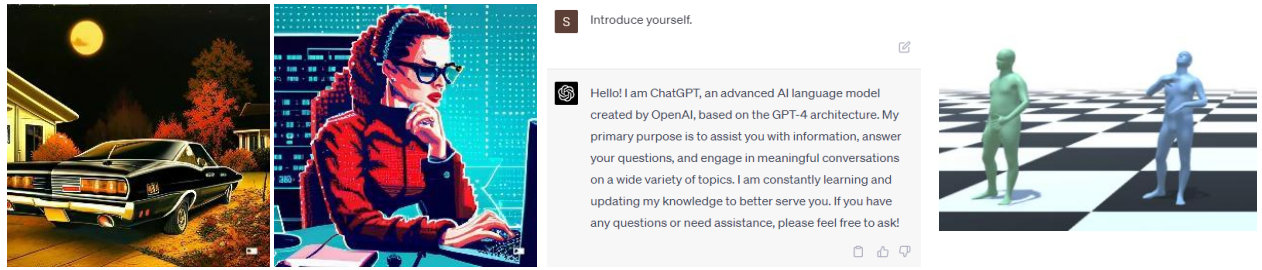
$P(X)$

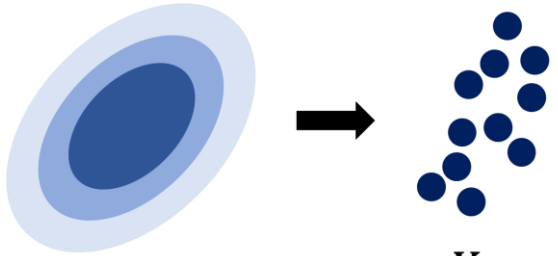
What Do We Do with a Probabilistic Model

Generation

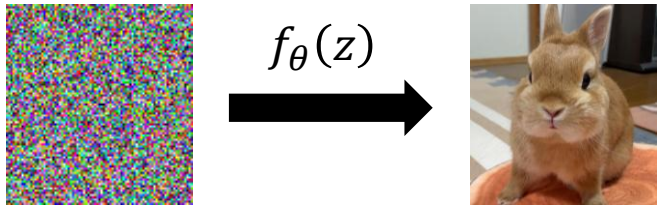
- Generate a new sample that follows the distribution

$$X \sim P(X)$$



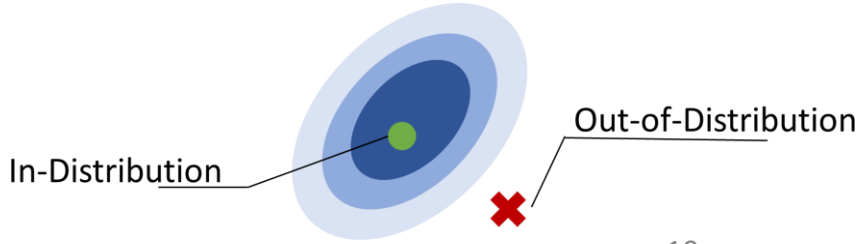


$$z \sim \mathcal{N}(0, I)$$



Probability (likelihood) evaluation

- How likely is a data given this distribution?



Deep Generative Models' Likelihood Evaluation is Incorrect

Out-of-distribution data is assigned higher likelihood

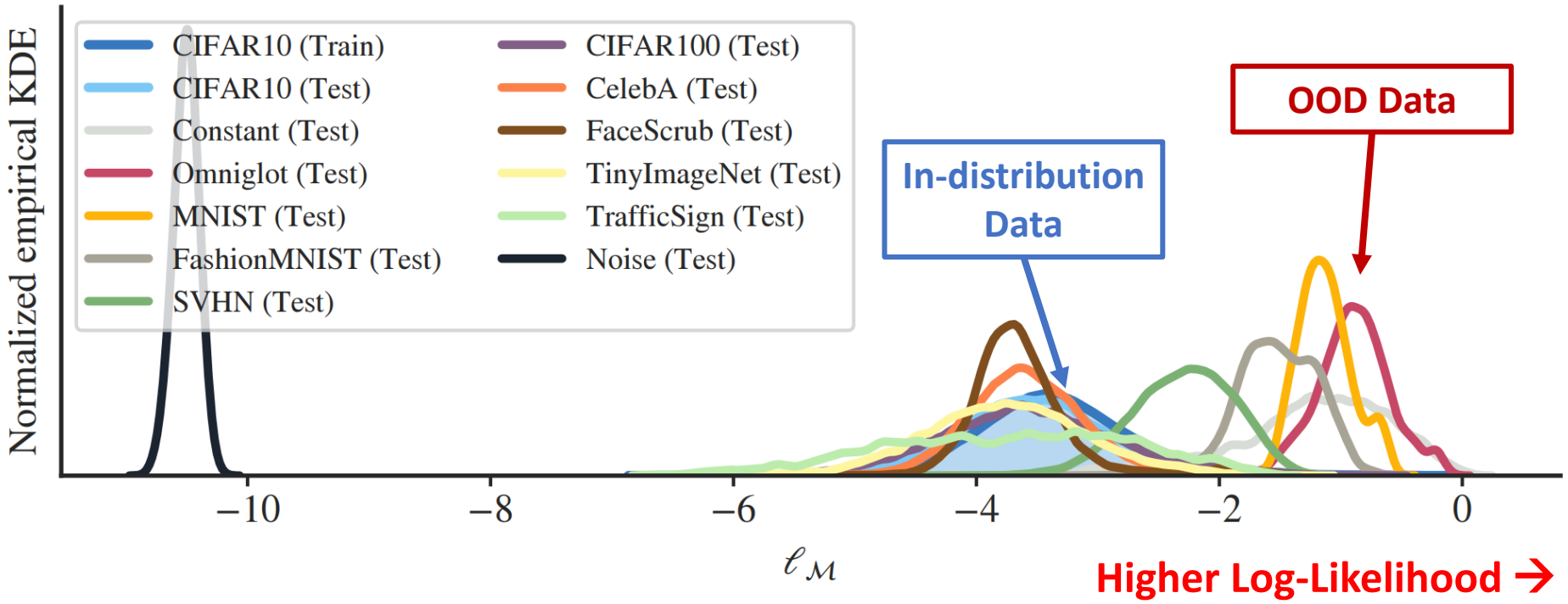


Figure 2: Log-likelihoods from a Glow model trained on CIFAR10. Qualitatively similar results are obtained for a PixelCNN++ model and when training with FashionMNIST.

Figure from (Serra et al., ICLR 2020) Originally reported by (Nalisnick et al., ICLR 2019) and (Hendrycks et al., ICLR 2019)

Energy-Based Models aka Unnormalized Probabilistic Models

$$p_\theta(\mathbf{x}) = \frac{1}{Z} e^{-E_\theta(\mathbf{x})}$$

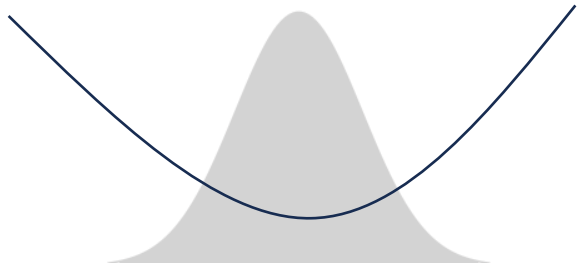
- Energy function $E_\theta: \mathbb{R}^D \to \mathbb{R}$,
- Partition function $Z = \int e^{-E_\theta(\mathbf{x})}\, d\mathbf{x} < \infty$,
- $\log p_\theta(\mathbf{x}) = -E_\theta(\mathbf{x}) - \log Z$ or $E_\theta(x) = -\log p_\theta(\mathbf{x}) - \log Z$

$E_\theta(x) \downarrow \Rightarrow p_\theta(x) \uparrow$

Example

A Gaussian distribution $p_\theta(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$

$\rightarrow E_\theta(x) = \frac{(x-\mu)^2}{2\sigma^2}, \theta = (\mu, \sigma)$



Energy-Based Models aka Unnormalized Probabilistic Models

$$p_\theta(\mathbf{x}) = \frac{1}{Z} e^{-E_\theta(\mathbf{x})}$$

Strength

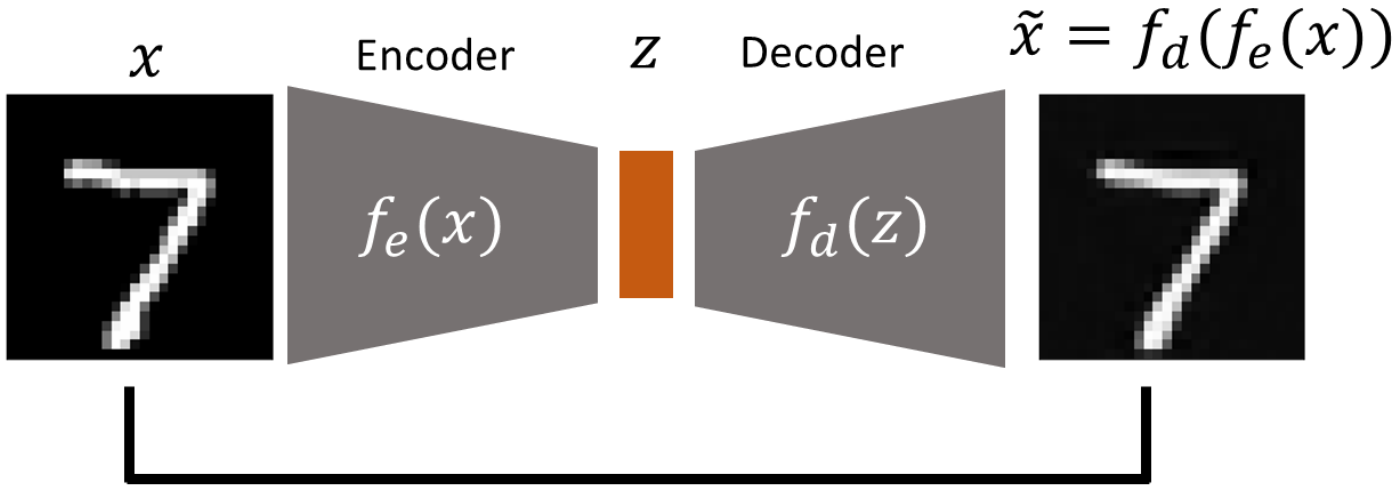
- Allows flexible design of $E_\theta(\mathbf{x})$. (no need to be normalized)
- Efficient OOD detection inference.

Weakness

- Training of $E_\theta(\mathbf{x})$ is not trivial and often requires MCMC.

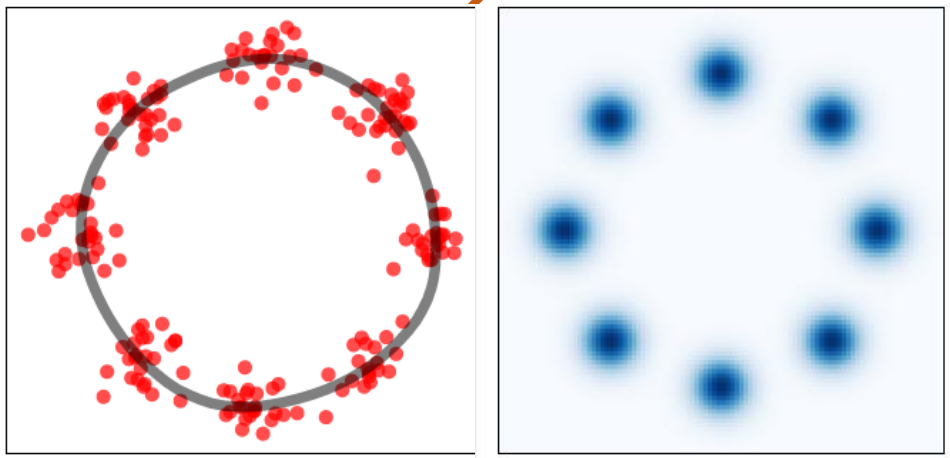
Yoon et al., Autoencoding Under Normalization Constraints, ICML 2021
Yoon et al., Energy-Based Models for Anomaly Detection: A Manifold Diffusion Recovery Approach, NeurIPS 2023

Normalized Autoencoder: Autoencoder as an Energy Function



$$p_\theta(\mathbf{x}) = \frac{1}{Z} e^{-E_\theta(\mathbf{x})}$$

Reconstruction Error
$l_\theta(x) = \|x - \tilde{x}\|^2$



Training of NAE

Gradient of Likelihood

$$\mathbb{E}_{x\sim p(x)}[\nabla_\theta \log p_\theta(x)] = -\mathbb{E}_{x\sim p(x)}[\nabla_\theta l_\theta(x)] + \mathbb{E}_{x'\sim p_\theta(x')}[\nabla_\theta l_\theta(x')]$$

Decreasing **Real Data** Reconstruction Error

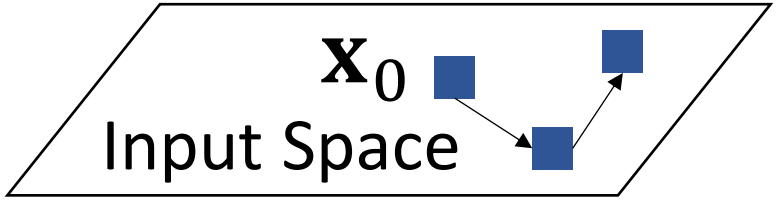
Increasing **Generated Sample** Reconstruction Error

Sampling from NAE $\mathbf{x}' \sim p_\theta(\mathbf{x}')$

$$\mathbb{E}_{\mathbf{x}' \sim p_\theta(\mathbf{x}')}[\nabla_\theta l_\theta(\mathbf{x}')]$$

Increasing **Generated Sample** Reconstruction Error

Langevin Monte Carlo (LMC) Sampling

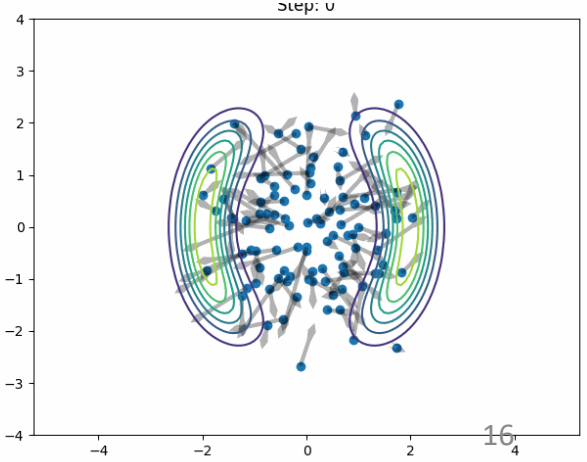


$t = 0, \cdots, T$

$\mathbf{x}_0 \sim$ Noise distribution

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \frac{\lambda_1}{2}\nabla_{\mathbf{x}} E_\theta(\mathbf{x}_t) + \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, \lambda_2)$$

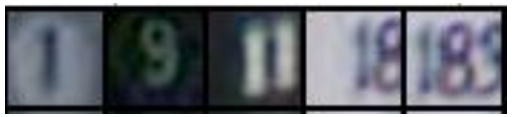
- $\mathbf{x}_T$: a sample from $p_\theta(\mathbf{x})$
- $E_\theta(\mathbf{x}) = l_\theta(\mathbf{x})$
- λ_1, λ_2: Hyperparameters. Step size and noise strength.



MNIST 0 - 8 (in) VS **MNIST 9** (out)

	AUROC
MPDR	**.971**
NAE	**.934**
AE	.537
DAE	.537
VAE(R)	.721
VAE(L)	.614
WAE	.799
GLOW	.426
IGEBM	.522

CIFAR-10 (in) VS **SVHN** (out)



	AUROC
MPDR	**.981**
NAE	**.920**
AE	.175
DAE	.175
VAE(R)	.191
VAE(L)	.185
WAE	.168
GLOW	.260
IGEBM	.371

NAE Deployed at Large Hadron Collider

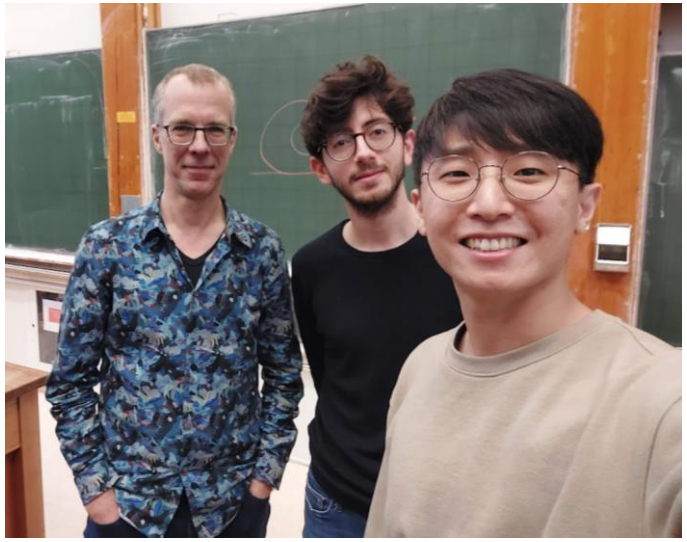
SciPost Physics Submission

A Normalized Autoencoder for LHC Triggers

Barry M. Dillon[1], Luigi Favaro[1], Tilman Plehn[1], Peter Sorrenson[2], and Michael Krämer[3]

1 Institut für Theoretische Physik, Universität Heidelberg, Germany
2 Heidelberg Collaboratory for Image Processing, Universität Heidelberg, Germany
3 Institute for Theoretical Particle Physics and Cosmology (TTK), RWTH Aachen University, Germany

February 28, 2023



(at Institute for Theoretical Physics, Heidelberg University, 2023 March)

Now deployed at CMS, a part of Large Hadron Collider

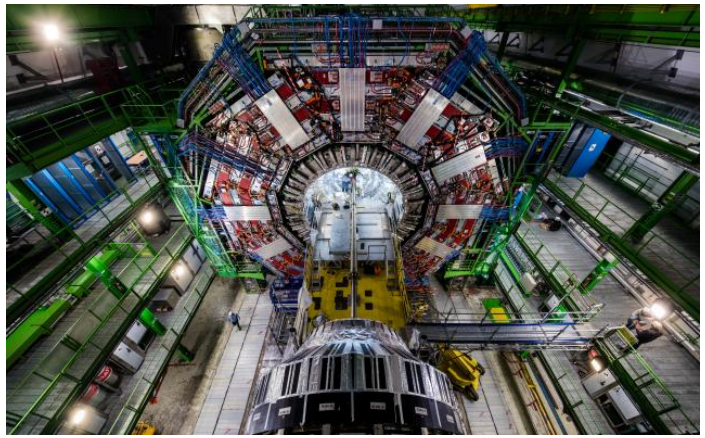
Available on the CERN CDS information server **CMS PAS MLG-24-002**

CMS Physics Analysis Summary

Contact: cms-conveners-ml@cern.ch 2024/09/23

Wasserstein normalized autoencoder

The CMS Collaboration



"**Normalized autoencoder was the only autoencoder that worked.**
It was a game changer."
– Prof. Tilman Plehn (Heidelberg University)

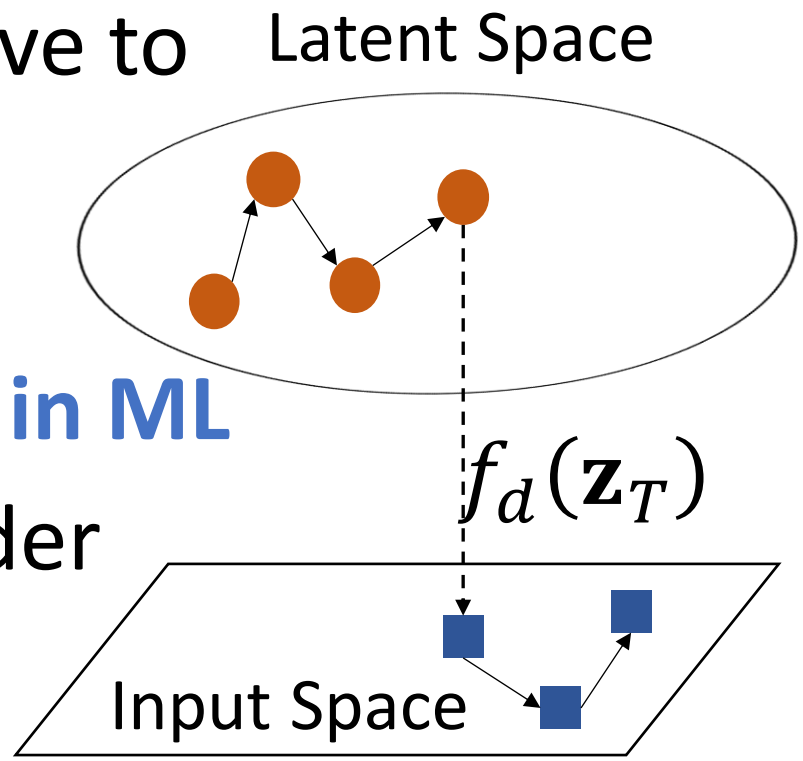
Two Frustrations

Tuning of NAE is extremely difficult

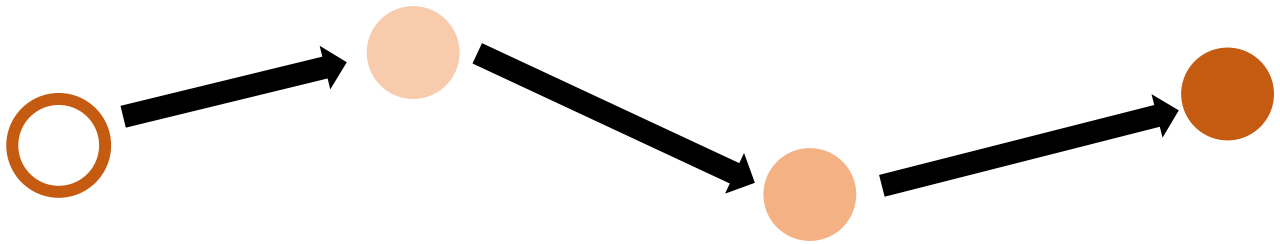
- The performance of NAE is highly sensitive to the hyperparameters of MCMC.

Anomaly detection is not well respected in ML

- I wanted to be connected with the broader ML community, e.g., diffusion or RL.

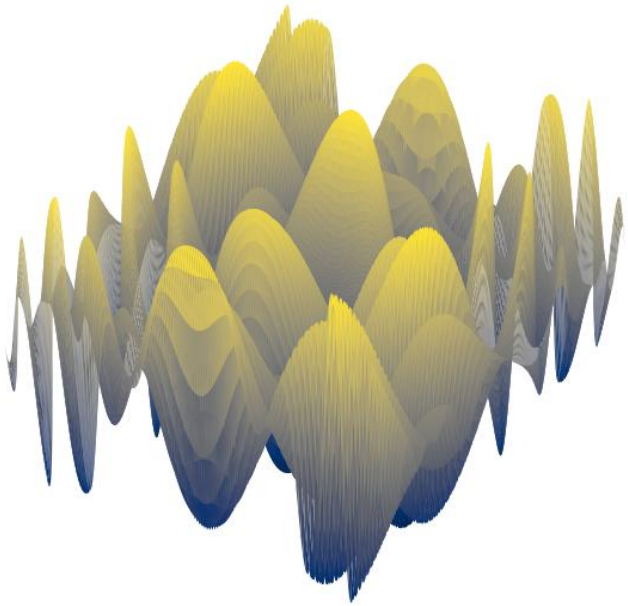


“What if Markov chain is tuned by itself?”



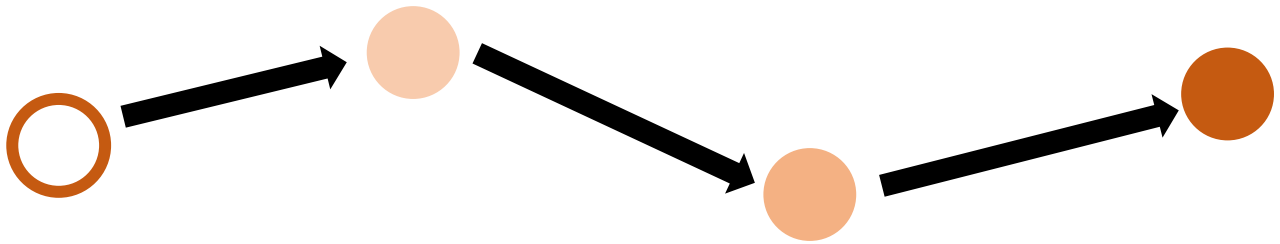
Markov Chain (in MCMC)

Reward



“Maybe this is Reinforcement Learning.”

Finding a **policy** that maximizes the **reward**



Langevin Monte Carlo

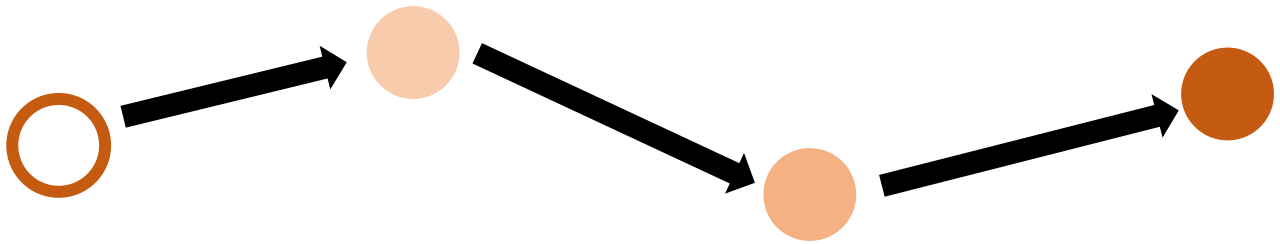
Policy (or Agent)

Energy-based Model
(Normalized Autoencoder)

Reward

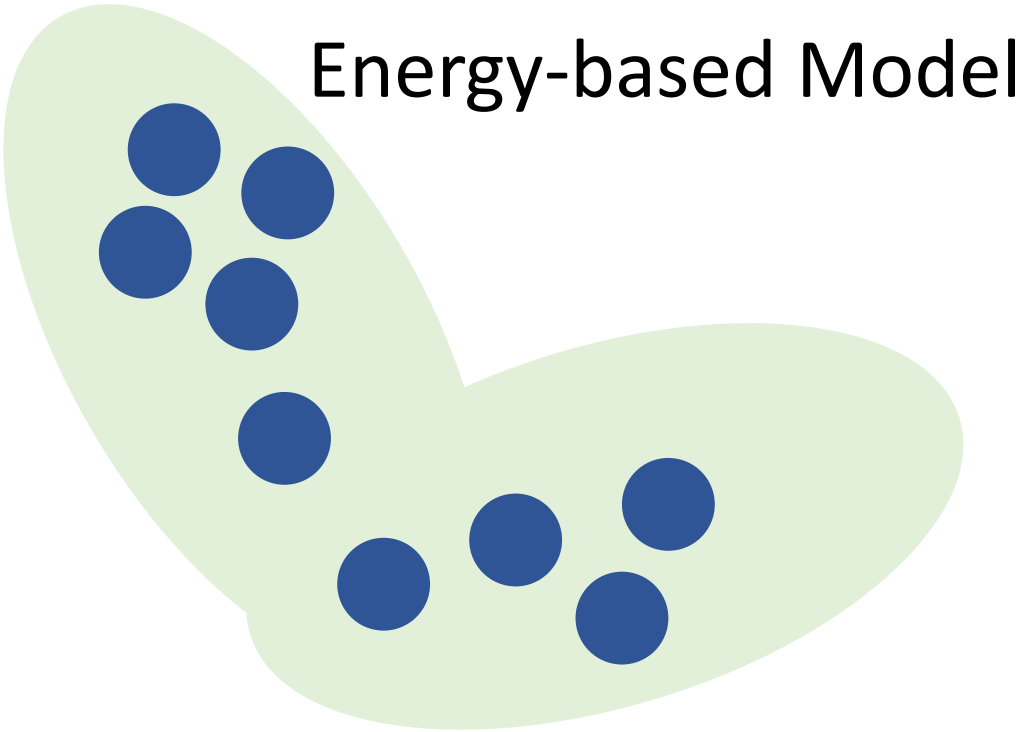
“This is Inverse Reinforcement Learning!”

Finding a **reward** function
from **demonstration data**



Langevin Monte Carlo

Policy (or Agent)



Reward

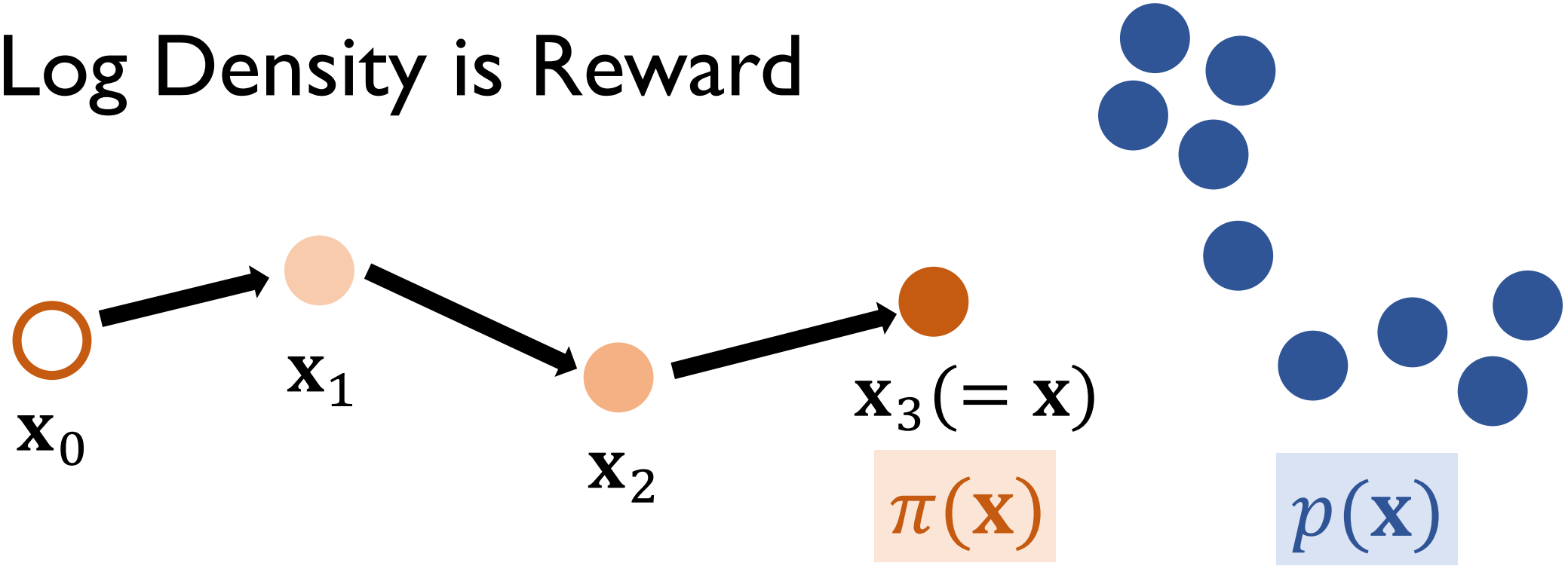
My Realization

1. Learning Energy is equivalent to learning the Reward in (Inverse) RL.

2. Langevin Monte Carlo can be replaced with a diffusion model.

We can revisit the whole generative modeling with these ideas

Log Density is Reward



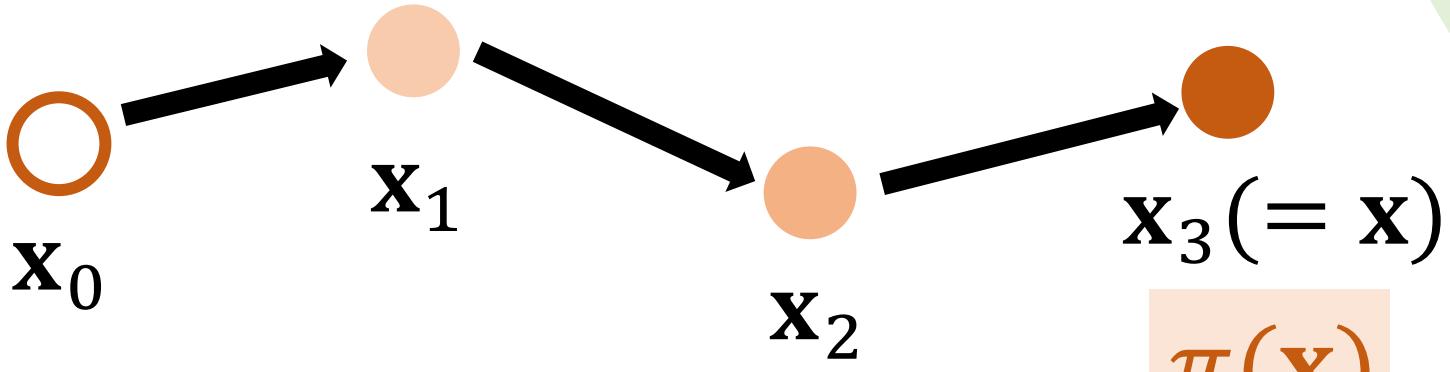
$$\min_{\pi} KL(\pi(\mathbf{x}) || p(\mathbf{x}))$$

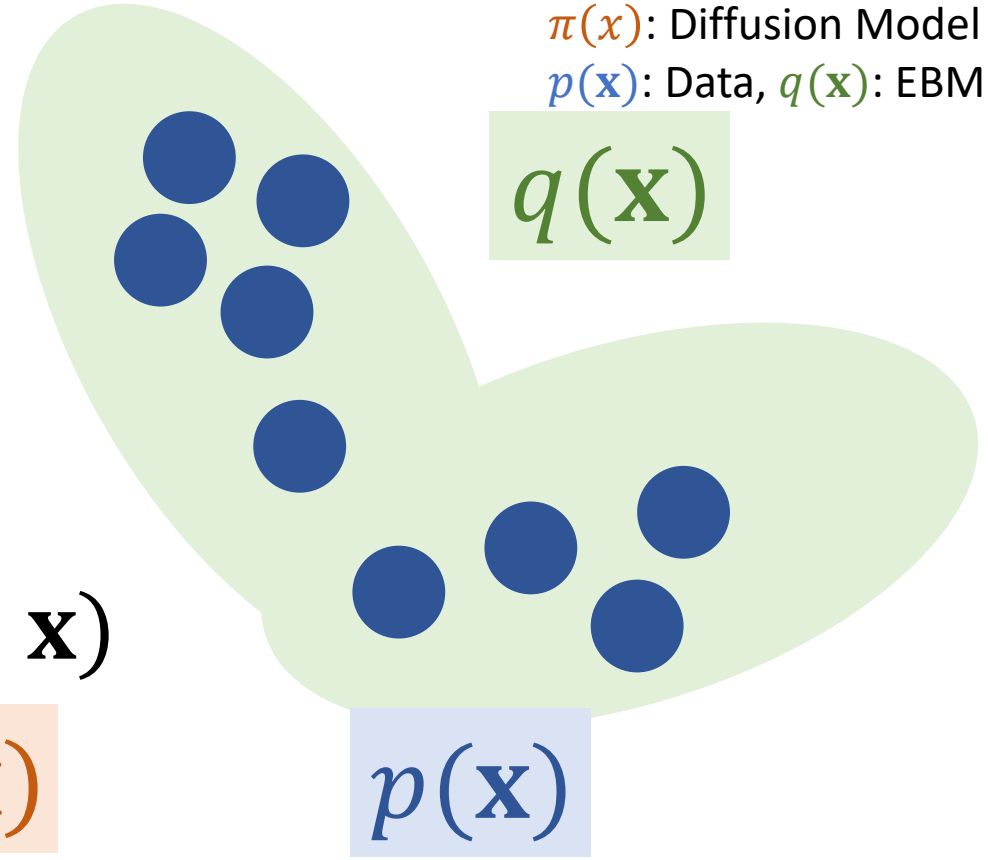
MaxEnt RL

$$= \min_{\pi} \mathbb{E}_{\pi}[-\log p(\mathbf{x})] - \mathcal{H}(\pi(\mathbf{x}))$$

$\pi(x)$: Diffusion Model
$p(\mathbf{x})$: Data, $q(\mathbf{x})$: EBM

EBM is Reward Model



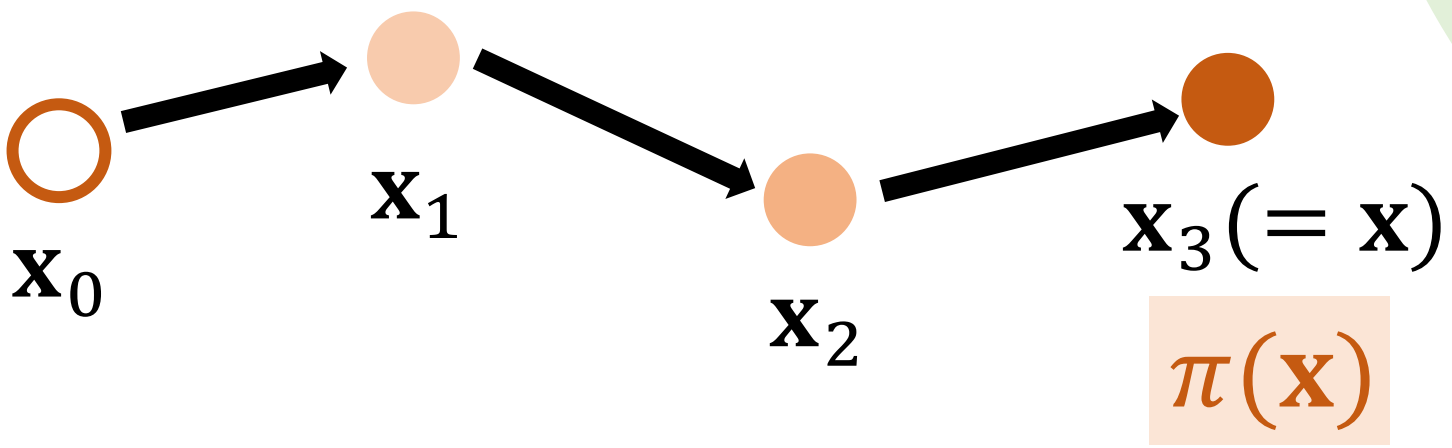


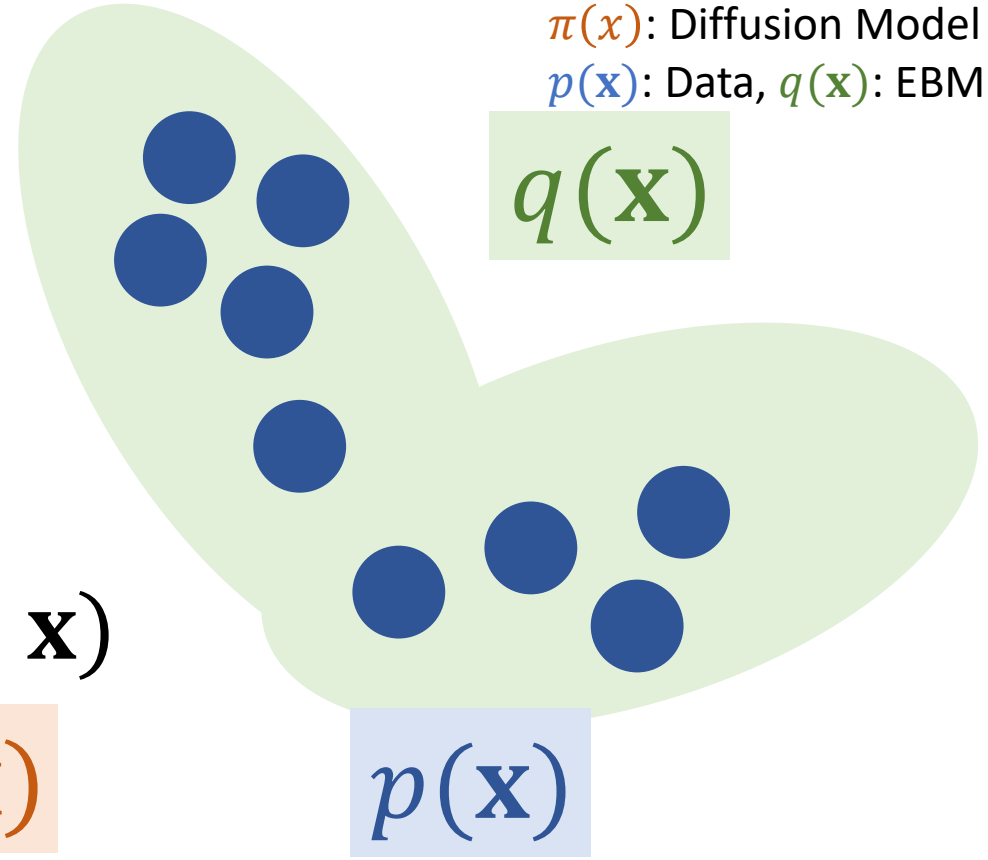
$$\min_q KL\big(p(\mathbf{x})||q(\mathbf{x})\big)$$

$$q(\mathbf{x}) = \frac{1}{Z} e^{-E(\mathbf{x})/\tau}$$

$\pi(x)$: Diffusion Model
$p(\mathbf{x})$: Data, $q(\mathbf{x})$: EBM

Joint Training is MaxEnt IRL





Reward Learning: $\min_q KL(p(\mathbf{x})||q(\mathbf{x}))$

Policy Learning : $\min_\pi KL(\pi(\mathbf{x})||q(\mathbf{x}))$

Difficult due to normalization constant of $q(x)$

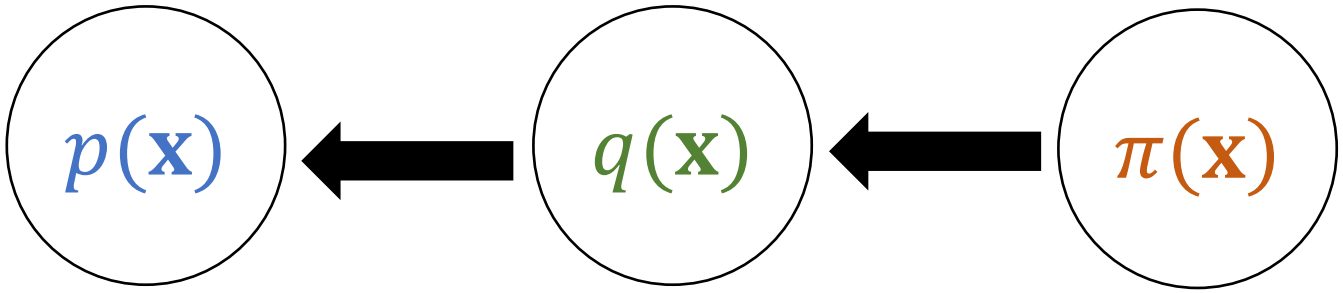
$\pi(x)$: Diffusion Model
$p(\mathbf{x})$: Data, $q(\mathbf{x})$: EBM

Generalized Contrastive Divergence

Inspired by Contrastive Divergence (Hinton, 2002)

$$\min_{q} \max_{\pi} KL(p||q) - KL(\pi||q)$$

Normalization constant cancelled out



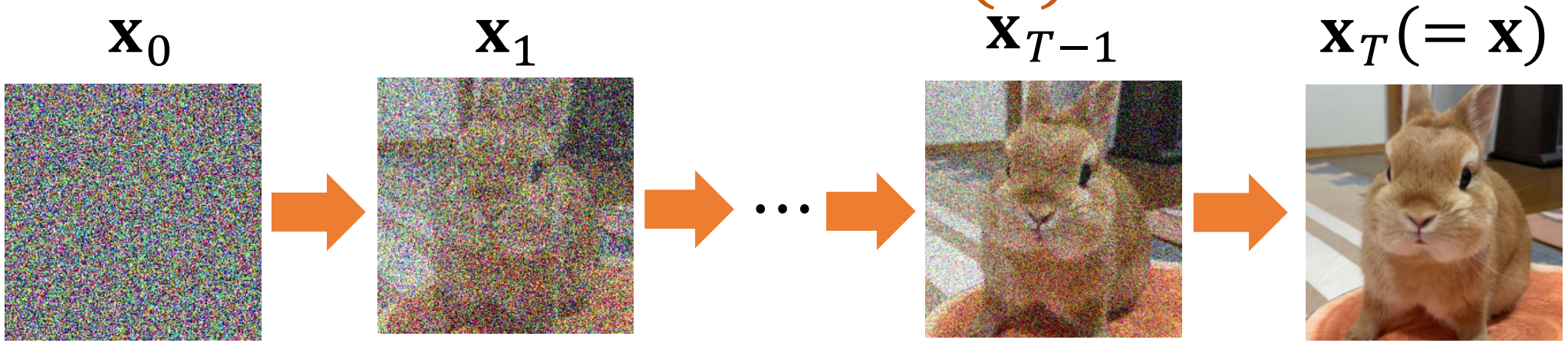
- Equilibrium: $p(\mathbf{x}) = q(\mathbf{x}) = \pi(\mathbf{x})$
- Equivalent to known objectives, e.g., adversarial EBM

"DxMI"

[Yoon et al., NeurIPS 2024 Oral (61/15,671, 0.4%)]

MaxEnt Inverse RL of Diffusion Models with EBMs

Diffusion Model $\pi(\mathbf{x})$



Training Data $p(\mathbf{x})$

Positive Sample

Negative Sample

Reward

Energy-Based Model $q(\mathbf{x})$

Diffusion Modeling is Behavioral Cloning (BC)

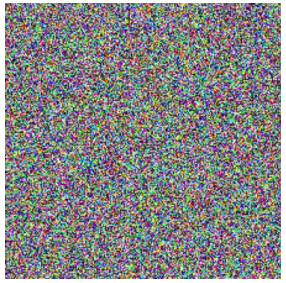
Expert Trajectory Demo

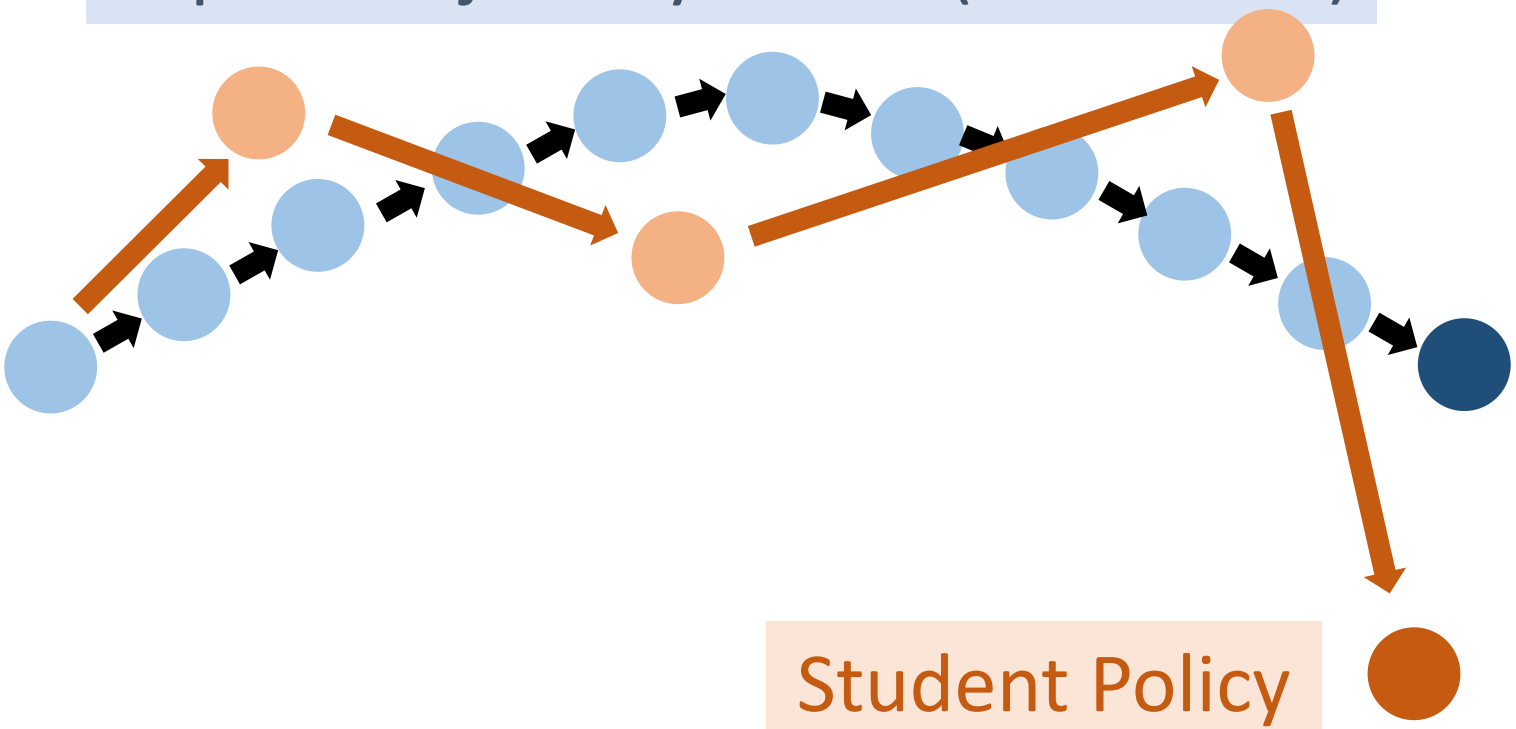
$\mathbf{x}_0$ → $\mathbf{x}_1$ → ⋯ → $\mathbf{x}_{T-1}$ → $\mathbf{x}_T (= \mathbf{x})$

$\pi(\mathbf{x})$

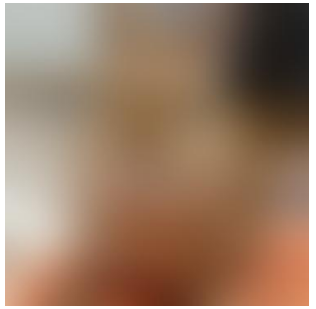
BC Fails Under Distribution Shift

Expert Trajectory Demo ($T = 1000$)

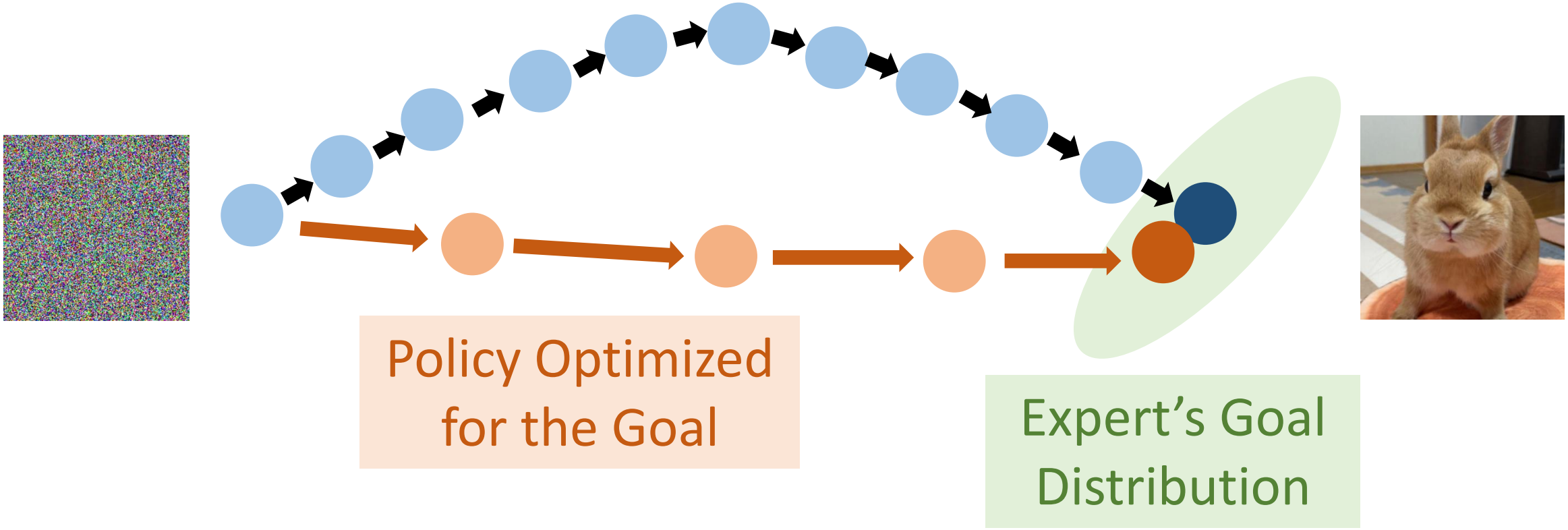




Student Policy
($T = 4$)



Inverse Reinforcement Learning Approach



Challenges in Generative Modeling

1. Training short-run **diffusion models** ($T \leq 10$)

2. Training **energy-based models (EBM)** without MCMC

$$q(x) = \frac{1}{Z} e^{-E(x)/\tau}$$

τ: temperature
Z: normalization constant

MaxEnt Inverse RL can address these challenges simultaneously.

Application: Few-Step Image Generation

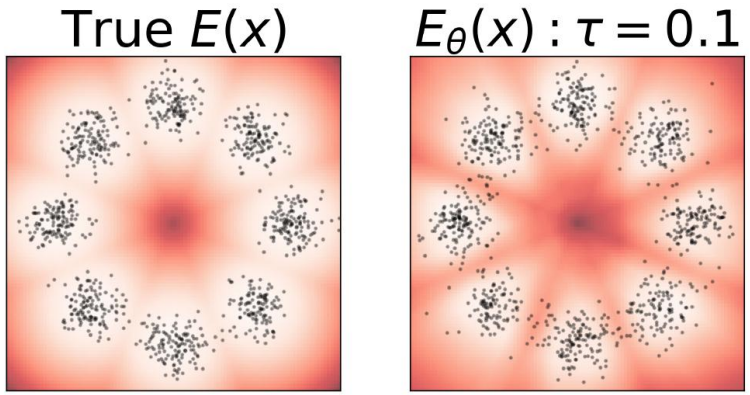
Before fine-tuning (EDM, T=10)

After fine-tuning with DxMI (T=10)

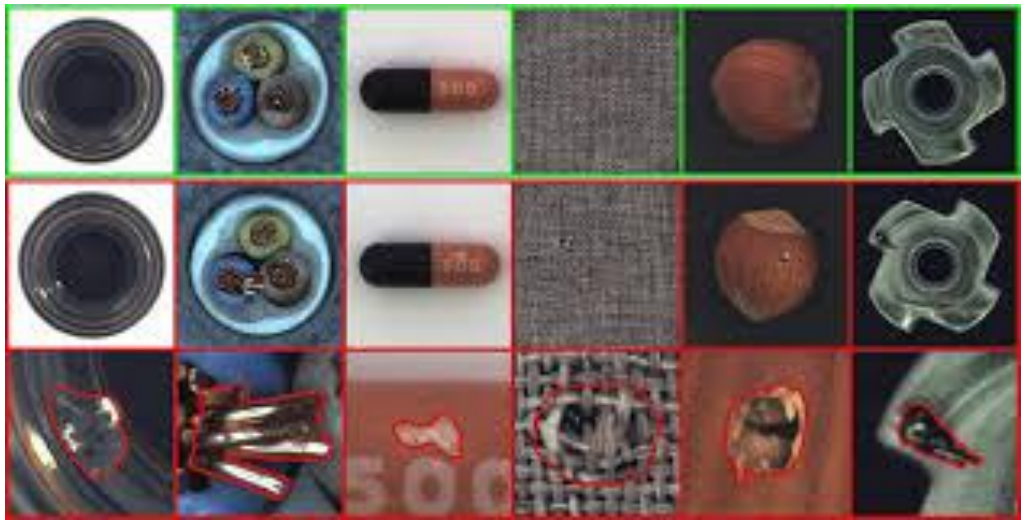
CIFAR-10	T	FID (↓)	Recall (↑)
DDPM (Ho et al., 2020)	1000	3.21	0.57
DxMI (Ours)	10	**3.19**	**0.63**
DDGAN (Xiao et al., 2021)	4	3.93	0.52
DxMI (Ours)	4	**3.22**	0.52

ImageNet 64x64	T	FID (↓)
EDM (Karras et al., 2022)	79	2.44
Consistency Model (Song et al., 2023)	2	4.70
	1	6.20
DxMI (Ours)	10	2.68
DxMI (Ours)	4	3.21

Application: Energy-Based Anomaly Detection



High $E(x)$ → Low $p(x)$ → Anomaly



MVTec-AD (Unified)	Det.	Loc.
DRAEM (Zavrtanik et al., 2021)	88.1	87.2
MPDR (Yoon et al., 2023)	96.0	96.7
UniAD (You et al., 2022)	96.5	96.8
DxMI (Ours)	**97.0**	**97.1**

Diffusion by Maximum Entropy IRL (DxMI)

$\pi(x)$: Diffusion Model
$p(\mathbf{x})$: Data, $q(\mathbf{x})$: EBM

Repeat:

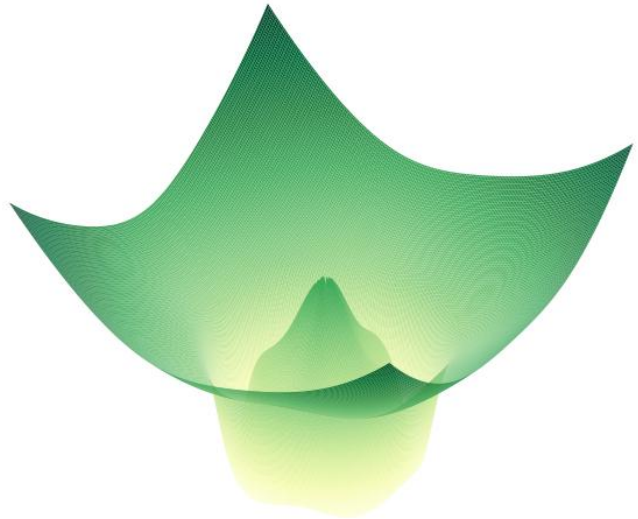
Sample $\mathbf{x} \sim p(\mathbf{x})$ and $\mathbf{x}_T \sim \pi(\mathbf{x})$

Reward learning: $\min_E \mathbb{E}_{p(\mathbf{x})}[E(\mathbf{x})] - \mathbb{E}_{\pi(\mathbf{x})}[E(\mathbf{x})]$

Policy learning: $\min_\pi KL(\pi(\mathbf{x})||q(\mathbf{x}))$

$\min_\pi KL(\pi||q)$ requires backpropagation through diffusion steps.

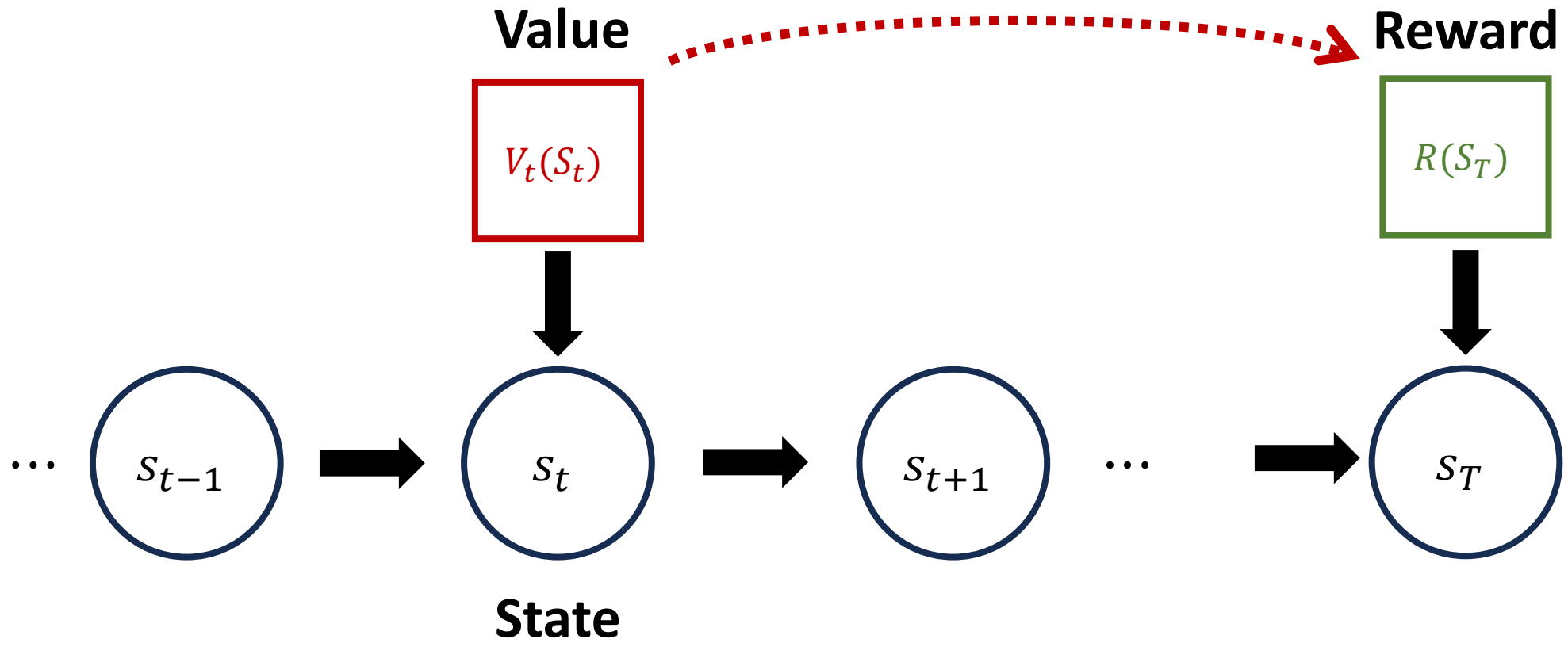
Value



Value Function $V_t(x)$

"How good is the current state? "

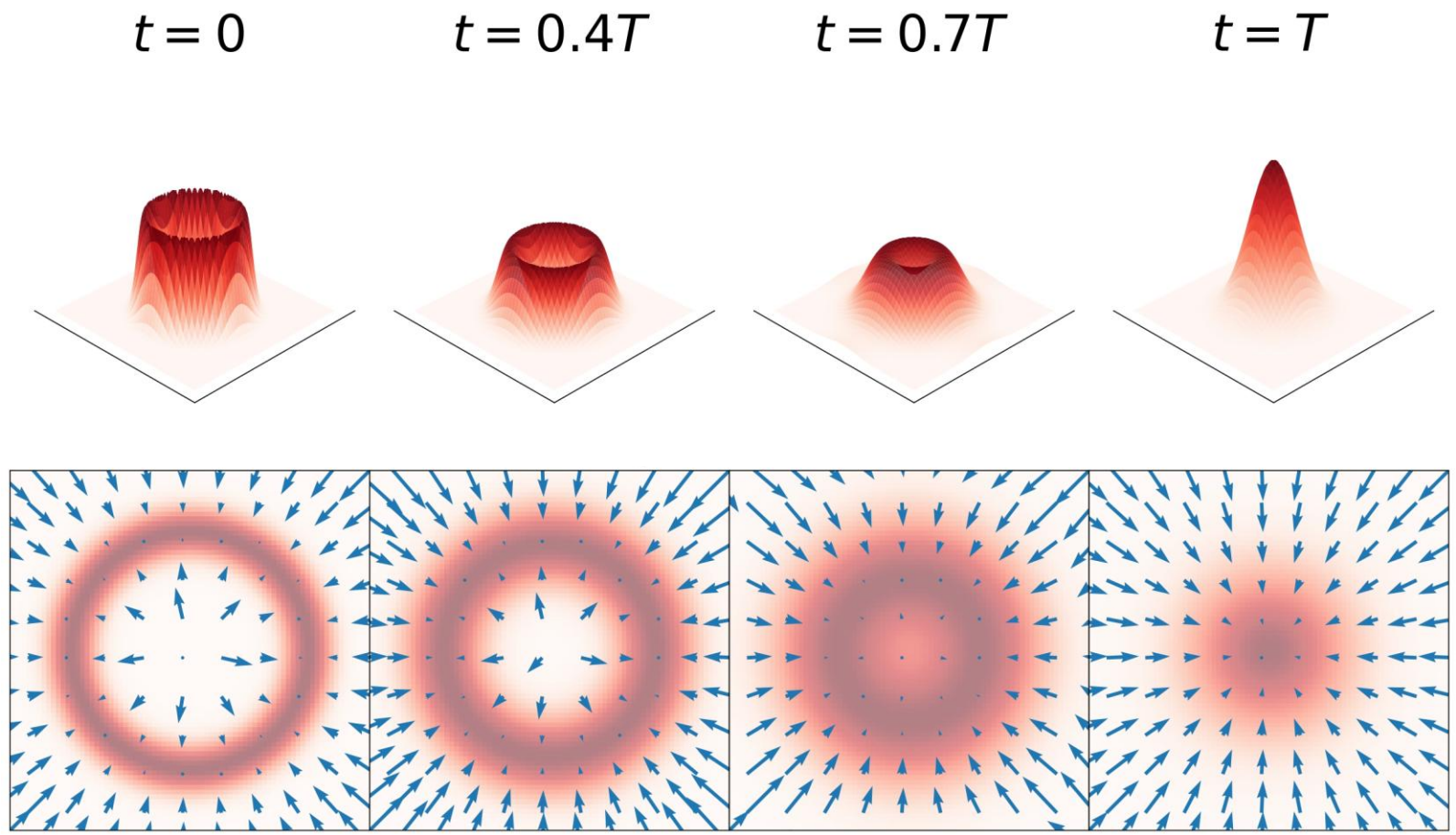
Value: Look-ahead Estimate of Future Reward



Temporal Difference Learning & Monte Carlo Estimation:
Propagation of signal without Backpropagation Through Time

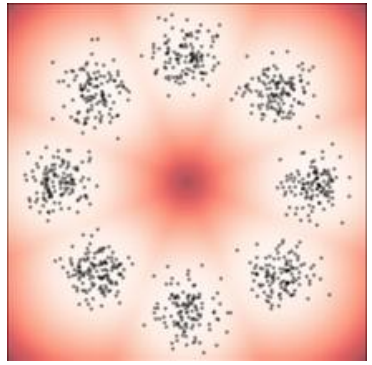
Value Function of Diffusion Process

Energy of marginal distribution p_t in a diffusion process



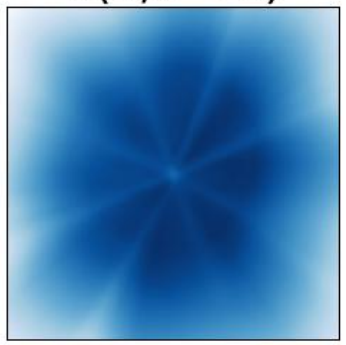
Proposal:Value Gradient Sampling

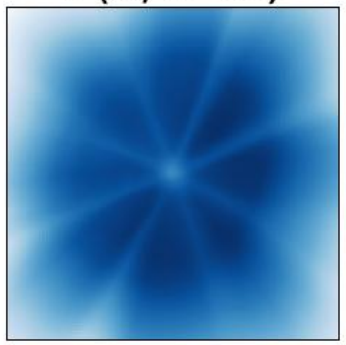
Target Density

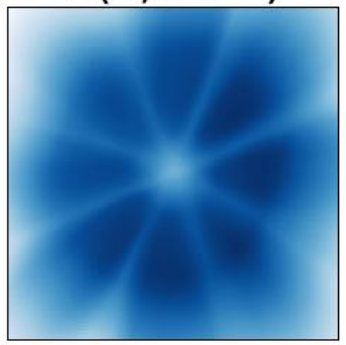


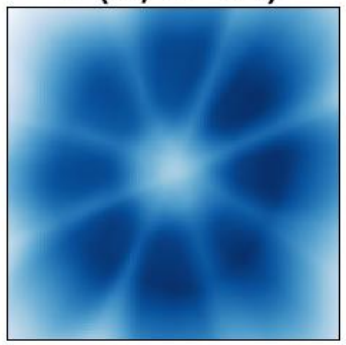
$$x_0 \sim \mathcal{N}(0, I)$$

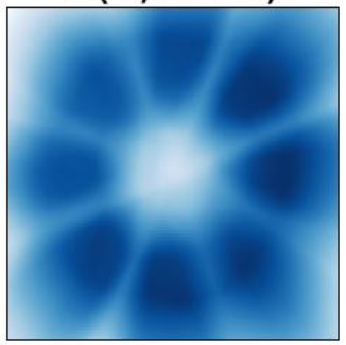
$$x_{t+1} = a_t x_t - \frac{\sigma_t^2}{\tau} \nabla_{x_{t+1}} V_{t+1}\left(x_{t+1}\right) + \sigma_t \epsilon_t$$

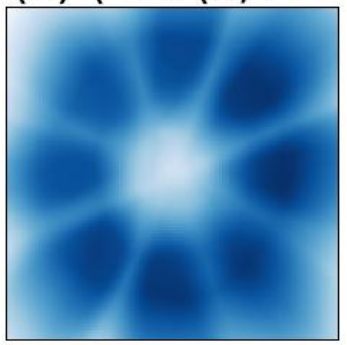
$V(x, t=0)$ 

$V(x, t=1)$ 

$V(x, t=2)$ 

$V(x, t=3)$ 

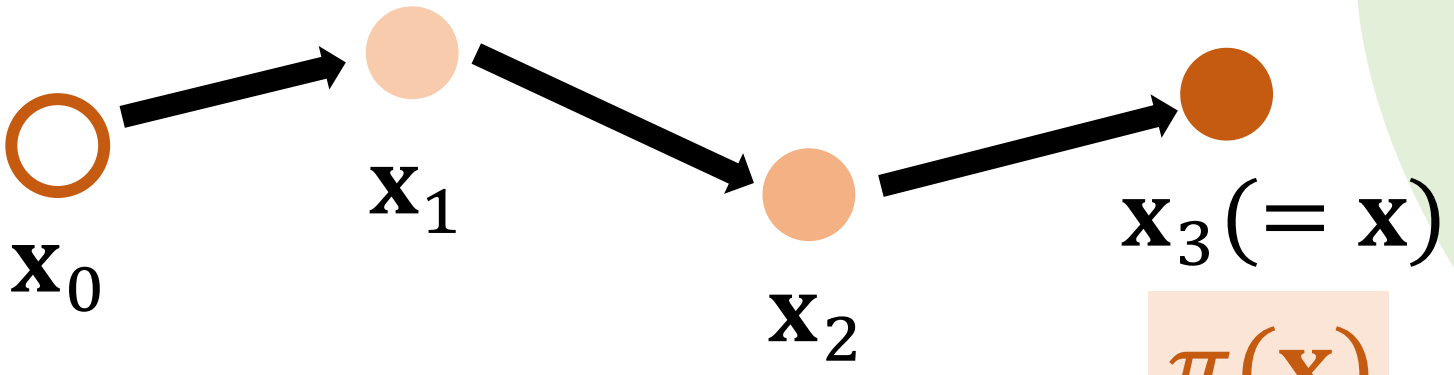
$V(x, t=4)$ 

$E(x)\ (= V(x, t=5))$ 

Value Gradient Sampler

Problem: Training a Diffusion Sampler
Given an unnormalized energy function $E(x)$,
generate independent samples drawn from $q(x) \sim \exp(-E(x))$

e.g., Path Integral Sampler, GFlowNet, etc



"Amortized" Parametric Sampler

$q(\mathbf{x})$

$\pi(\mathbf{x})$

Temporal Difference Learning of Diffusion Sampler

$$\min_{\pi} KL(\pi(\mathbf{x}_T)||q(\mathbf{x}_T))$$

Temporal Difference Learning

$V_1(\mathbf{x}_1)$ ⋯ $V_{t-2}(\mathbf{x}_{t-2})$ $V_{t-1}(\mathbf{x}_{t-1})$ $q(\mathbf{x})$

$\mathbf{x}_0$ $\mathbf{x}_1$ ⋯ $\mathbf{x}_{t-2}$ $\mathbf{x}_{t-1}$ $\mathbf{x}_T(=\mathbf{x})$

No backpropagation through time

Optimal Control Formulation

Objective

Data Processing Inequality

$$KL\left(\pi(\mathbf{x}_T)||q(\mathbf{x}_T)\right) \leq KL(\pi(\mathbf{x}_{0:T})||q(\mathbf{x}_T)\tilde{q}(\mathbf{x}_{0:T-1}|\mathbf{x}_T))$$

Optimal Control

Value-based Dynamic Programming

$$\min_{\pi} \mathbb{E}_{\pi}\left[\underbrace{E(\mathbf{x}_T)}_{\text{Terminal Cost}} + \tau \underbrace{\sum_{t=0}^{T-1} \log \pi(\mathbf{x}_{t+1}|\mathbf{x}_t) + \sum_{t=0}^{T-1} \frac{\tau}{2s_t^2}\|\mathbf{x}_{t+1}-\mathbf{x}_t\|^2}_{\text{Running Cost } R(\mathbf{x}_t,\mathbf{x}_{t+1})}\right]$$

$R(\mathbf{x}_t, \mathbf{x}_{t+1})$: Running cost

Value Learning (Update $V_t(\mathbf{x}_t)$)

Temporal Difference (TD) learning

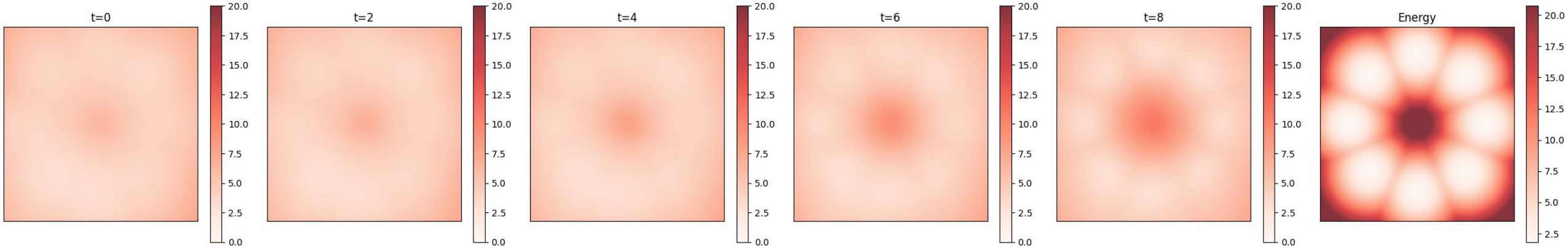
$$\min_{V_t} \mathbb{E}_{\mathbf{x}_t, \mathbf{x}_{t+1} \sim \pi} \left[\left(\underbrace{V_{t+1}(\mathbf{x}_{t+1}) + R(\mathbf{x}_t, \mathbf{x}_{t+1})}_{\text{TD Target}} - V_t(\mathbf{x}_t) \right)^2 \right]$$

Policy Improvement (Update $\pi(\mathbf{x})$)

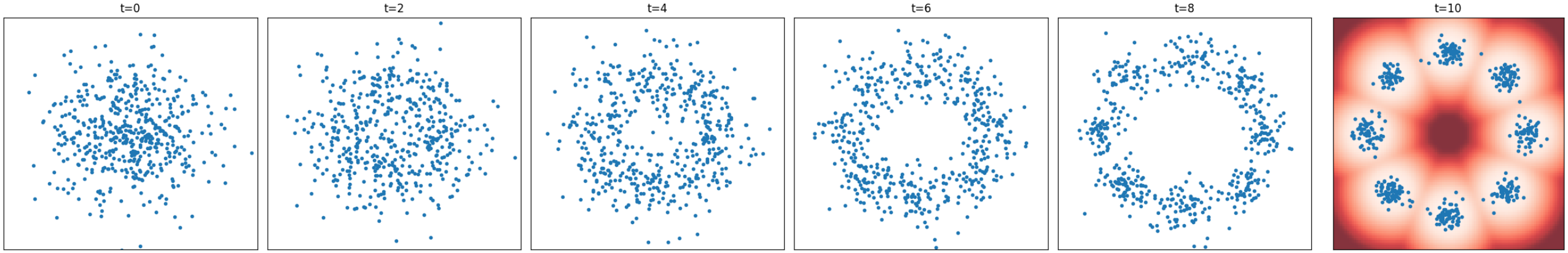
Transition-wise diffusion model update using $V_t(\mathbf{x}_t)$

$$\min_{\pi(\mathbf{x}_{t+1}|\mathbf{x}_t)} \mathbb{E}_{\mathbf{x}_t, \mathbf{x}_{t+1} \sim \pi} [V_{t+1}(\mathbf{x}_{t+1}) + R(\mathbf{x}_t, \mathbf{x}_{t+1})]$$

Value Functions

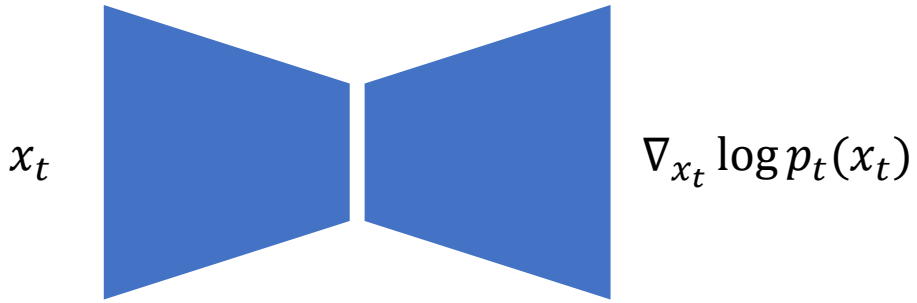


Samples

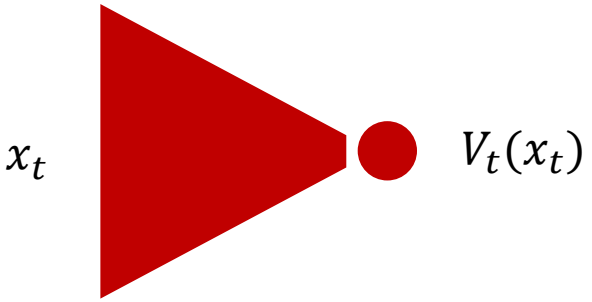


VGS vs Conventional Diffusion Samplers

Conventional Diffusion Samplers
Directly parametrize the drift (i.e., score)



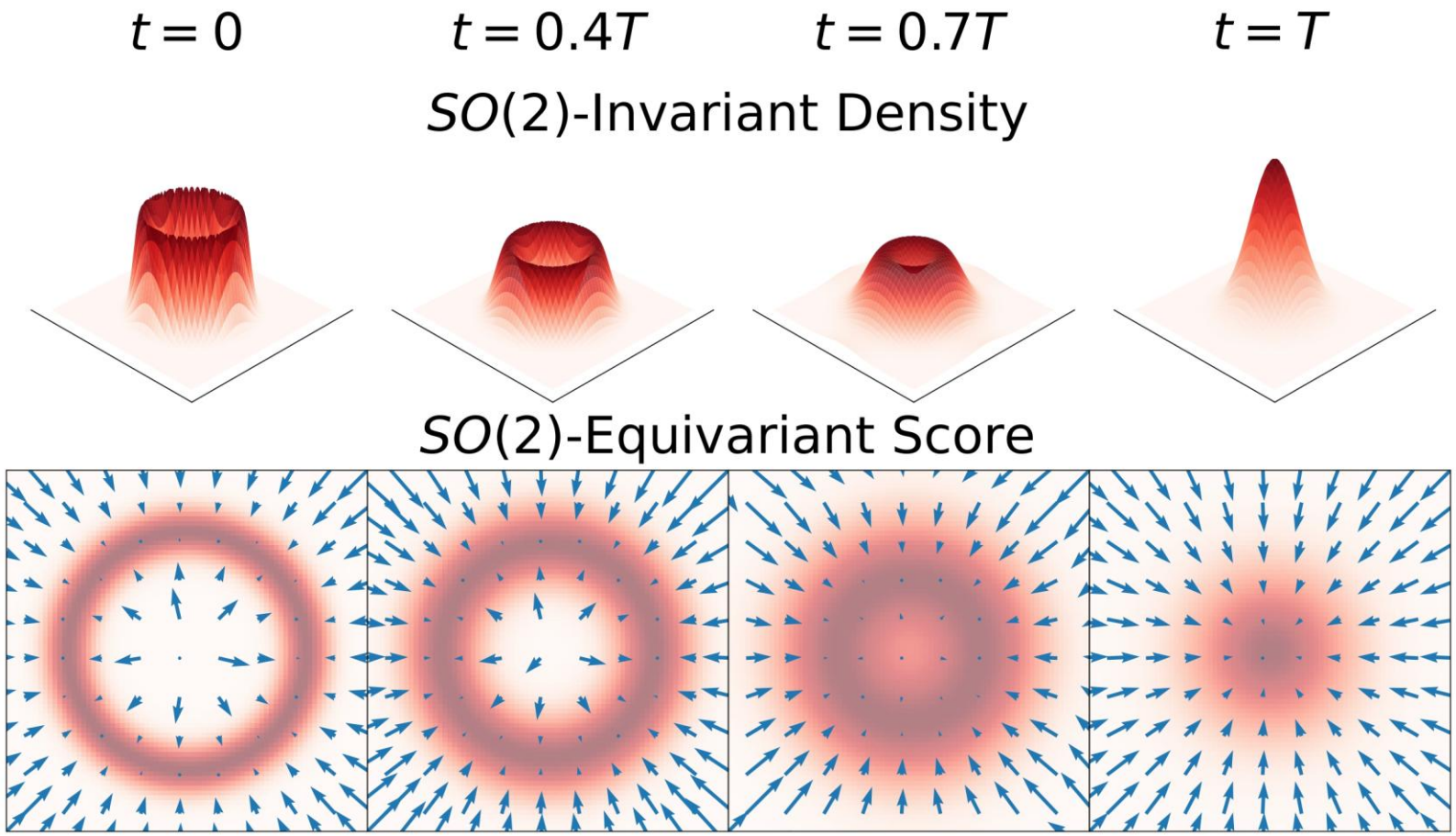
Value Gradient Samplers
Drift is parametrized as the gradient of the value function



$$\nabla_{x_t} V_t(x_t) = \nabla_{x_t} \log p_t(x_t)$$

Value function $V_t(x_t)$ is the energy of a diffusion density $p_t(x_t)$

Invariant Value Induces Equivariant Gradients



VGS is Effective in Equivariant Sampling

LJ-55 Benchmark

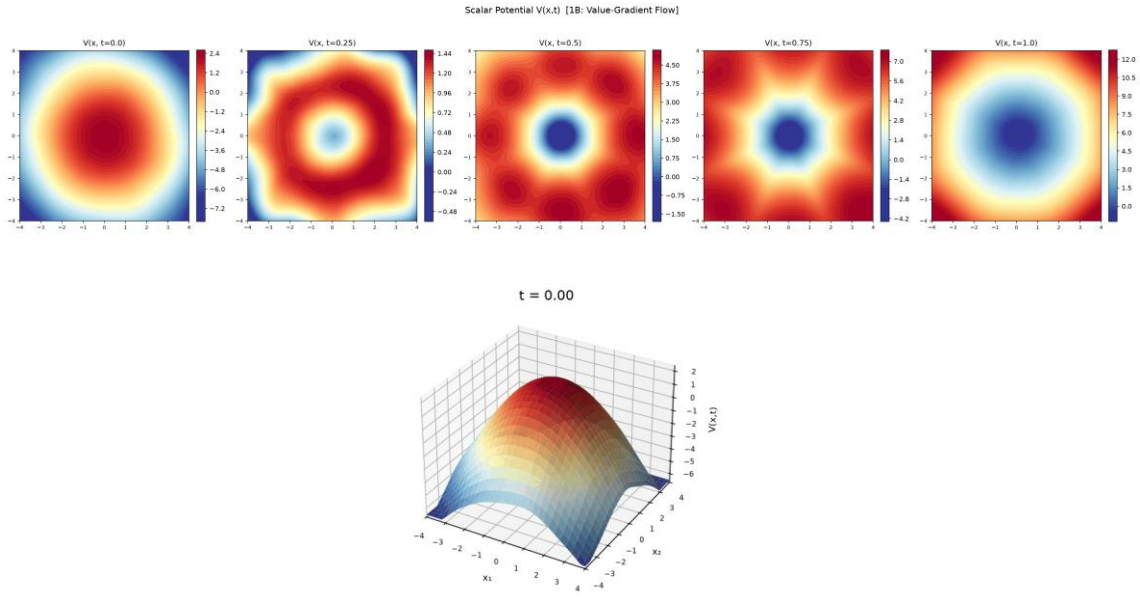
Method	Wasserstein Distance (↓)	Generation Time (s/512 samples)	# Params.
FAB	17.217	58	5.4M
iDEM	16.460	135	1.2M
DiKL	18.050	0.02	2M
PIS	Diverged	27.4	0.6M
DDS	Diverged	27.4	0.6M
VGS	**15.906**	**0.13**	**10M**

- Other diffusion samplers use EGNN to implement $SE(3)$ equivariant score.
- EGNN scales $O(n^2)$ and is very slow.

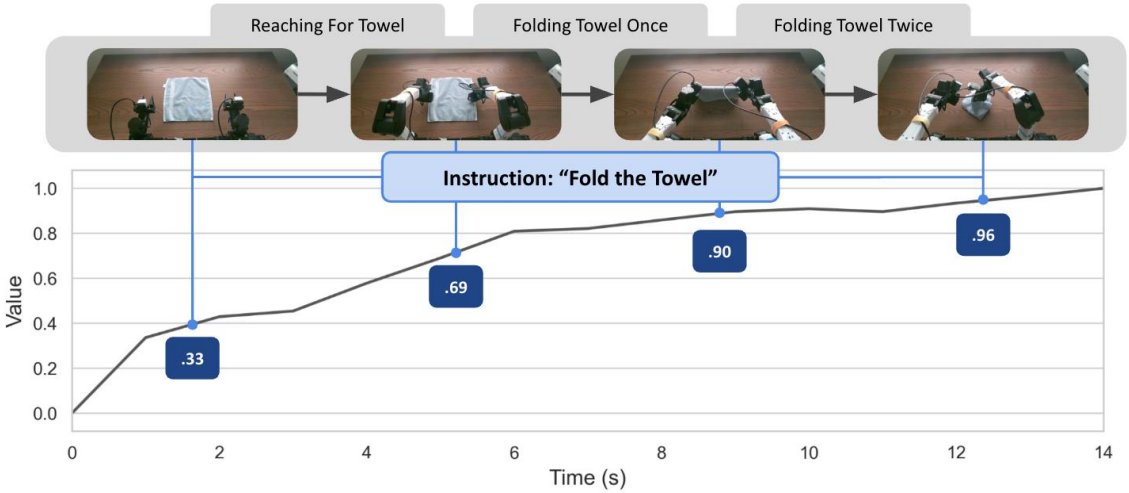
Future Directions

Future Directions

Value Gradient Flow



Robot Reward Models / Value Models



Thank you for listening!

Contact: swyoon@unist.ac.kr

UNIST Active Generative Agents (AGA) Laboratory

Generative Modeling
+ Reinforcement Learning
+ Applications (Robotics, Science & Engineering, Adtech, …)

Hiring interns and PhD students

Open for academic and industrial collaborations