A Diffusive Classification Loss for Learning Energy-based Generative Models

RuiKang OuYang*[1], Louis Grenioux*[2,3]
José Miguel Hernández-Lobato[1]

***Equal Contribution, order assigned randomly.**

1 UNIVERSITY OF CAMBRIDGE

2 CMAP

3 FLATIRON INSTITUTE

Learning energy-based generative models

Task

Learning energy-based generative models

Task

- We would like to train time-dependent energy-based models (EBMs)

Learning energy-based generative models

Task

- We would like to train time-dependent energy-based models (EBMs)
  - $p_t^\theta(y) = \exp(-U_t^\theta(y))/Z_t^\theta$

Learning energy-based generative models

Task

- We would like to train time-dependent energy-based models (EBMs)
  - $p_t^\theta(y) = \exp(-U_t^\theta(y))/Z_t^\theta$
  - intractable normalising constant: $Z_t^\theta = \int \exp(-U_t^\theta(y))dy$

Learning energy-based generative models

Task

- We would like to train time-dependent energy-based models (EBMs)
  - $p_t^\theta(y) = \exp(-U_t^\theta(y))/Z_t^\theta$
  - intractable normalising constant: $Z_t^\theta = \int \exp(-U_t^\theta(y))dy$
- Inference-time Application

Learning energy-based generative models

Task

- We would like to train time-dependent energy-based models (EBMs)
  - $p_t^\theta(y) = \exp(-U_t^\theta(y))/Z_t^\theta$
  - intractable normalising constant: $Z_t^\theta = \int \exp(-U_t^\theta(y))dy$
- Inference-time Application
  - Model composition [1, 2]

Learning energy-based generative models

Task

- We would like to train time-dependent energy-based models (EBMs)
  - $p_t^\theta(y) = \exp(-U_t^\theta(y))/Z_t^\theta$
  - intractable normalising constant: $Z_t^\theta = \int \exp(-U_t^\theta(y))dy$
- Inference-time Application
  - Model composition [1, 2]
  - Boltzmann Generation [3]

Learning energy-based generative models

Task

- We would like to train time-dependent energy-based models (EBMs)
  - $p_t^\theta(y) = \exp(-U_t^\theta(y))/Z_t^\theta$
  - intractable normalising constant: $Z_t^\theta = \int \exp(-U_t^\theta(y))dy$
- Inference-time Application
  - Model composition [1, 2]
  - Boltzmann Generation [3]
  - Free energy difference estimation [4, 5]

Learning energy-based generative models

Task

- We would like to train time-dependent energy-based models (EBMs)
  - $p_t^\theta(y) = \exp(-U_t^\theta(y))/Z_t^\theta$
  - intractable normalising constant: $Z_t^\theta = \int \exp(-U_t^\theta(y))dy$
- Inference-time Application
  - Model composition [1, 2]
  - Boltzmann Generation [3]
  - Free energy difference estimation [4, 5]

[1] Du, Yilun, et al. "Reduce, reuse, recycle: Compositional generation with energy-based diffusion models and mcmc." *International conference on machine learning*. PMLR, 2023.
[2] Thornton, James, et al. "Composition and control with distilled energy diffusion models and sequential monte carlo." *arXiv preprint arXiv:2502.12786* (2025).
[3] Phillips, Angus, et al. "Particle denoising diffusion sampler." *arXiv preprint arXiv:2402.06320* (2024).
[4] Fleuret, Fran, and Tristan Bereau. "Solvation Free Energies from Neural Thermodynamic Integration." *arXiv e-prints* (2024): arXiv-2410.
[5] Du, Yuanqi, et al. "FEAT: Free energy estimators with adaptive transport." *Advances in Neural Information Processing Systems* 38 (2026): 142130-142167.

Standard Methods

Maximum Likelihood Estimation

- Setup

Standard Methods

Maximum Likelihood Estimation

- Setup
  - $Y_t = X_t + \gamma(t)z$

Standard Methods

Maximum Likelihood Estimation

- Setup
  - $Y_t = X_t + \gamma(t)z$
  - X_t : $\alpha_t X_0 + \beta_t X_1$ for Stochastic Interpolants; or $\alpha_t X_0$ for Diffusion Models

Standard Methods

Maximum Likelihood Estimation

- Setup
  - $Y_t = X_t + \gamma(t)z$
  - X_t : $\alpha_t X_0 + \beta_t X_1$ for Stochastic Interpolants; or $\alpha_t X_0$ for Diffusion Models
  - $z \sim \mathcal{N}(0, I)$

Standard Methods

Maximum Likelihood Estimation

- Setup
  - $Y_t = X_t + \gamma(t)z$
  - X_t : $\alpha_t X_0 + \beta_t X_1$ for Stochastic Interpolants; or $\alpha_t X_0$ for Diffusion Models
  - $z \sim \mathcal{N}(0, I)$
- Contrastive Divergence

Standard Methods

Maximum Likelihood Estimation

- Setup
  - $Y_t = X_t + \gamma(t)z$
  - $X_t : \alpha_t X_0 + \beta_t X_1$ for Stochastic Interpolants; or $\alpha_t X_0$ for Diffusion Models
  - $z \sim \mathcal{N}(0, I)$
- Contrastive Divergence
  - $\nabla_\theta \mathcal{L}(\theta) = \mathbb{E}_t\left[\mathbb{E}_{p_t}[\nabla_\theta U_t^\theta(Y_t)] - \mathbb{E}_{p_t^\theta}[\nabla_\theta U_t^\theta(Y_t)]\right]$

Standard Methods

Maximum Likelihood Estimation

- Setup
  - $Y_t = X_t + \gamma(t)z$
  - $X_t : \alpha_t X_0 + \beta_t X_1$ for Stochastic Interpolants; or $\alpha_t X_0$ for Diffusion Models
  - $z \sim \mathcal{N}(0, I)$
- Contrastive Divergence
  - $\nabla_\theta \mathcal{L}(\theta) = \mathbb{E}_t\left[\mathbb{E}_{p_t}[\nabla_\theta U_t^\theta(Y_t)] - \mathbb{E}_{p_t^\theta}[\nabla_\theta U_t^\theta(Y_t)]\right]$
  - Requires sample from the p_t^θ ❌ —> Difficult [1, 2]

Standard Methods

Maximum Likelihood Estimation

- Setup
  - $Y_t = X_t + \gamma(t)z$
  - $X_t : \alpha_t X_0 + \beta_t X_1$ for Stochastic Interpolants; or $\alpha_t X_0$ for Diffusion Models
  - $z \sim \mathcal{N}(0, I)$
- Contrastive Divergence
  - $\nabla_\theta \mathcal{L}(\theta) = \mathbb{E}_t\left[\mathbb{E}_{p_t}[\nabla_\theta U_t^\theta(Y_t)] - \mathbb{E}_{p_t^\theta}[\nabla_\theta U_t^\theta(Y_t)]\right]$
  - Requires sample from the p_t^θ ❌ —> Difficult [1, 2]

[1] Du, Yilun, et al. "Improved contrastive divergence training of energy based models." *arXiv preprint arXiv:2012.01316* (2020).
[2] Gao, Ruiqi, et al. "Learning energy-based models by diffusion recovery likelihood." *arXiv preprint arXiv:2012.08125* (2020).

Standard Methods

Score Matching

- Setup
  - $Y_t = X_t + \gamma(t)z$
  - X_t : $\alpha_t X_0 + \beta_t X_1$ for Stochastic Interpolants; or $\alpha_t X_0$ for Diffusion Models
  - $z \sim \mathcal{N}(0, I)$
- Denoising Score Matching

Standard Methods

Score Matching

- Setup
  - $Y_t = X_t + \gamma(t)z$
  - $X_t : \alpha_t X_0 + \beta_t X_1$ for Stochastic Interpolants; or $\alpha_t X_0$ for Diffusion Models
  - $z \sim \mathcal{N}(0, I)$
- Denoising Score Matching
  - $\mathbb{E}\left[\left\| \nabla \log p_t^{\theta}(Y_t) - \nabla \log p_t(Y_t \mid X_t) \right\|^2 \right]$

Standard Methods

Score Matching

- Setup
  - $Y_t = X_t + \gamma(t)z$
  - $X_t : \alpha_t X_0 + \beta_t X_1$ for Stochastic Interpolants; or $\alpha_t X_0$ for Diffusion Models
  - $z \sim \mathcal{N}(0, I)$
- Denoising Score Matching
  - $\mathbb{E}\left[\left\| \nabla \log p_t^{\theta}(Y_t) - \nabla \log p_t(Y_t \mid X_t) \right\|^2 \right]$
  - High variance $\gamma(t) \to 0$ ❌

Standard Methods

Score Matching

- Setup
  - $Y_t = X_t + \gamma(t)z$
  - X_t : $\alpha_t X_0 + \beta_t X_1$ for Stochastic Interpolants; or $\alpha_t X_0$ for Diffusion Models
  - $z \sim \mathcal{N}(0, I)$
- Denoising Score Matching
  - $\mathbb{E}\left[\left\| \nabla \log p_t^{\theta}(Y_t) - \nabla \log p_t(Y_t \mid X_t) \right\|^2 \right]$
  - High variance $\gamma(t) \to 0$ ❌
  - blindness ❌

Standard Methods

Score Matching

- Setup
  - $Y_t = X_t + \gamma(t)z$
  - X_t : $\alpha_t X_0 + \beta_t X_1$ for Stochastic Interpolants; or $\alpha_t X_0$ for Diffusion Models
  - $z \sim \mathcal{N}(0, I)$
- Denoising Score Matching
  - $\mathbb{E}\left[\left\| \nabla \log p_t^\theta(Y_t) - \nabla \log p_t(Y_t \mid X_t) \right\|^2 \right]$
  - blindness ❌
  - High variance $\gamma(t) \to 0$ ❌

Blindness of score [1]

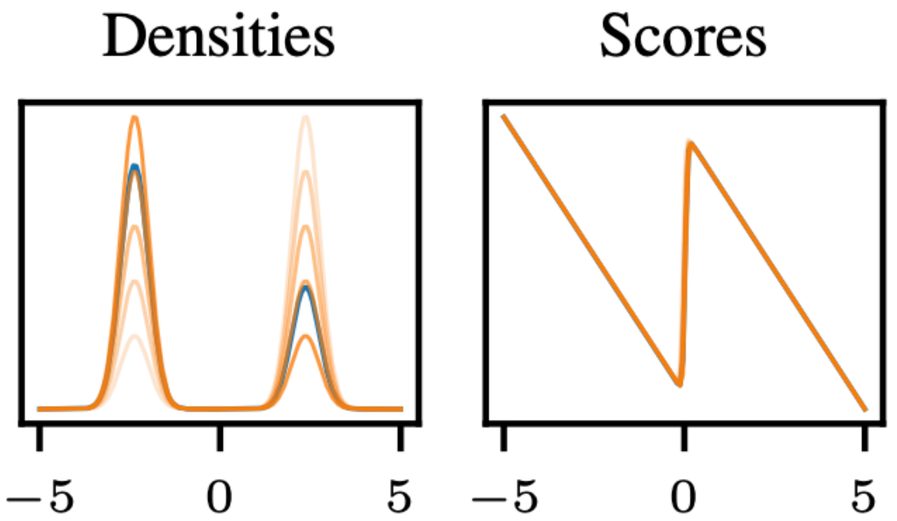
[1] Zhang, Mingtian, et al. "Towards healing the blindness of score matching." *arXiv preprint arXiv:2209.07396* (2022).

Standard Methods

Score Matching

- Setup
  - $Y_t = X_t + \gamma(t)z$
  - $(X_t)_t$ any stochastic process
  - $z \sim \mathcal{N}(0, I)$
- Denoising Score Matching
  - $\mathbb{E}\left[\left\| \nabla \log p_t^\theta(Y_t) - \nabla \log p_t(Y_t \mid X_t) \right\|^2 \right]$
  - blindness ❌
  - High variance $\gamma(t) \to 0$ ❌

Blindness of score [1]



Scores are indistinguishable among different mode weights

[1] Zhang, Mingtian, et al. "Towards healing the blindness of score matching." *arXiv preprint arXiv:2209.07396* (2022).

Diffusive CLassiFication

A general expression

- Setup
  - $Y_t = X_t + \gamma(t)z$
  - X_t :
    - $\alpha_t X_0 + \beta_t X_1$ for Stochastic Interpolants;
    - $\alpha_t X_0$ for Diffusion Models
  - $z \sim \mathcal{N}(0, I)$
- A classification loss

Diffusive CLassiFication

A general expression

- Setup
  - $Y_t = X_t + \gamma(t)z$
  - X_t :
    - $\alpha_t X_0 + \beta_t X_1$ for Stochastic Interpolants;
    - $\alpha_t X_0$ for Diffusion Models
  - $z \sim \mathcal{N}(0, I)$

- A classification loss
  - $\mathscr{L}_{\text{clf}}(\theta; N) = \mathbb{E}_{t_{1:N}}[\mathscr{L}_{\text{clf}}(\theta; t_{1:N})]$

Diffusive CLassiFication

A general expression

- Setup
  - $Y_t = X_t + \gamma(t)z$
  - X_t :
    - $\alpha_t X_0 + \beta_t X_1$ for Stochastic Interpolants;
    - $\alpha_t X_0$ for Diffusion Models
  - $z \sim \mathcal{N}(0, I)$

- A classification loss
  - $\mathscr{L}_{\text{clf}}(\theta; N) = \mathbb{E}_{t_{1:N}}[\mathscr{L}_{\text{clf}}(\theta; t_{1:N})]$
  - $$\mathscr{L}_{\text{clf}}(\theta; t_{1:N}) = -\frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{p_{t_i}}\left[\log \frac{p^{\theta}_{t_i}(Y_{t_i})}{\sum_{j=1}^{N} p^{\theta}_{t_j}(Y_{t_i})}\right]$$

Diffusive CLassiFication

A general expression

- Setup
  - $Y_t = X_t + \gamma(t)z$
  - X_t :
    - $\alpha_t X_0 + \beta_t X_1$ for Stochastic Interpolants;
    - $\alpha_t X_0$ for Diffusion Models
  - $z \sim \mathcal{N}(0, I)$

- A classification loss
  - $\mathscr{L}_{\text{clf}}(\theta; N) = \mathbb{E}_{t_{1:N}}[\mathscr{L}_{\text{clf}}(\theta; t_{1:N})]$
  - $$\mathscr{L}_{\text{clf}}(\theta; t_{1:N}) = -\frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{p_{t_i}}\left[\log \frac{p^{\theta}_{t_i}(Y_{t_i})}{\sum_{j=1}^{N} p^{\theta}_{t_j}(Y_{t_i})}\right]$$
    - Cross Entropy!!

Diffusive CLassiFication

A general expression

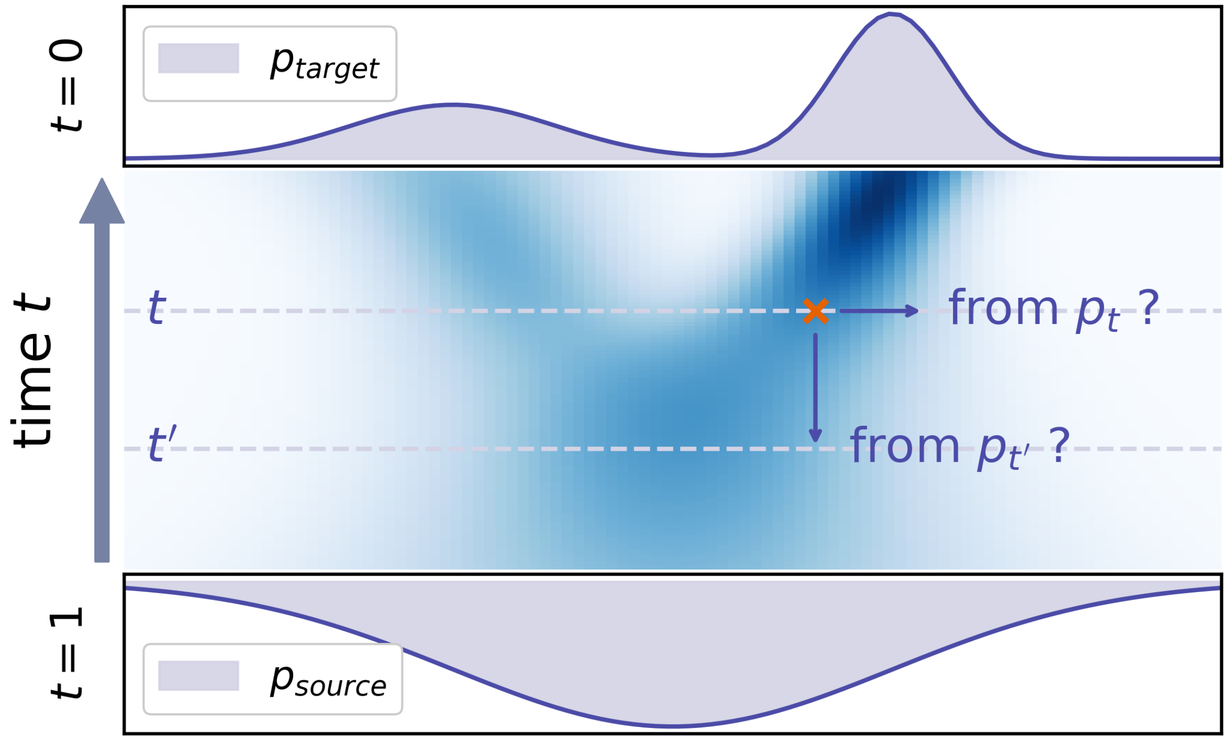
- Setup
  - $Y_t = X_t + \gamma(t)z$
  - X_t :
    - $\alpha_t X_0 + \beta_t X_1$ for Stochastic Interpolants;
    - $\alpha_t X_0$ for Diffusion Models
  - $z \sim \mathcal{N}(0, I)$

- A classification loss
  - $\mathscr{L}_{\text{clf}}(\theta; N) = \mathbb{E}_{t_{1:N}}[\mathscr{L}_{\text{clf}}(\theta; t_{1:N})]$
  - $$\mathscr{L}_{\text{clf}}(\theta; t_{1:N}) = -\frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{p_{t_i}}\left[\log \frac{p^{\theta}_{t_i}(Y_{t_i})}{\sum_{j=1}^{N} p^{\theta}_{t_j}(Y_{t_i})}\right]$$
    - Cross Entropy!!
  - Optimum: $$\frac{p^{\theta}_{t_i}(y)}{\sum_{j=1}^{N} p^{\theta}_{t_J}(y)} = \frac{p_{t_i}(y)}{\sum_{j=1}^{N} p_{t_J}(y)}$$

Diffusive CLassiFication

A binary case illustration

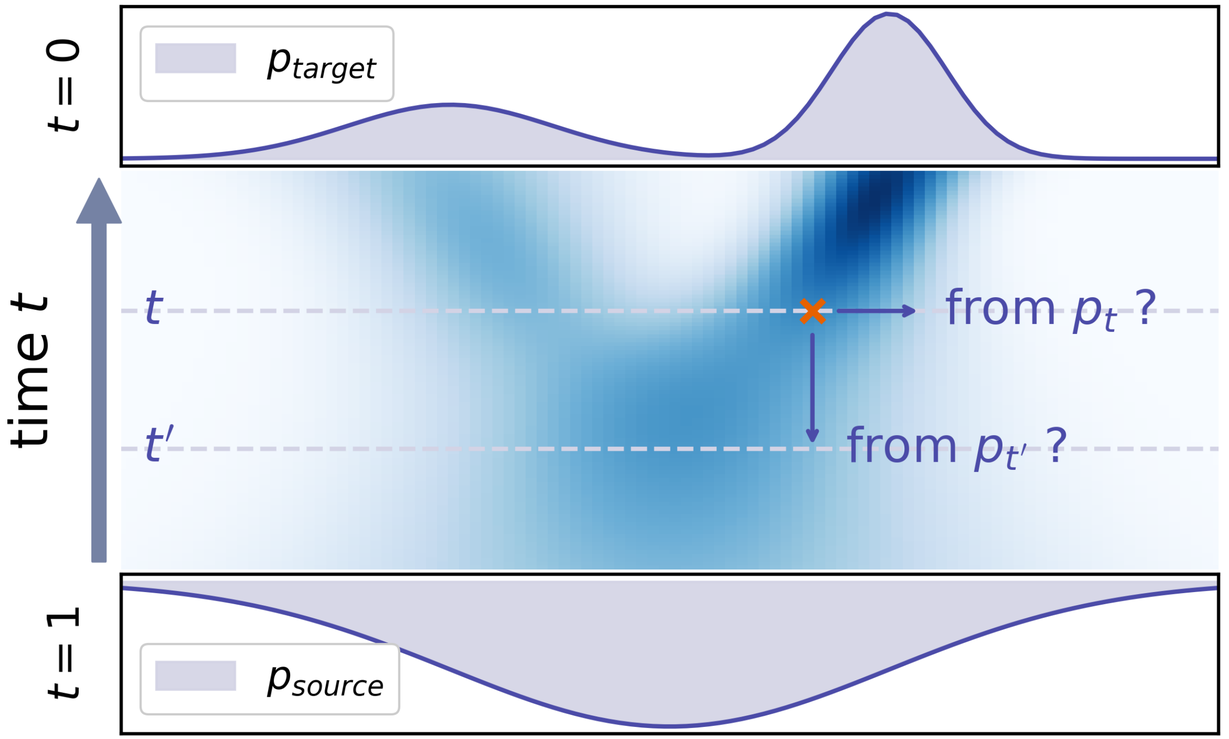
- The binary case



Diffusive CLassiFication

A binary case illustration

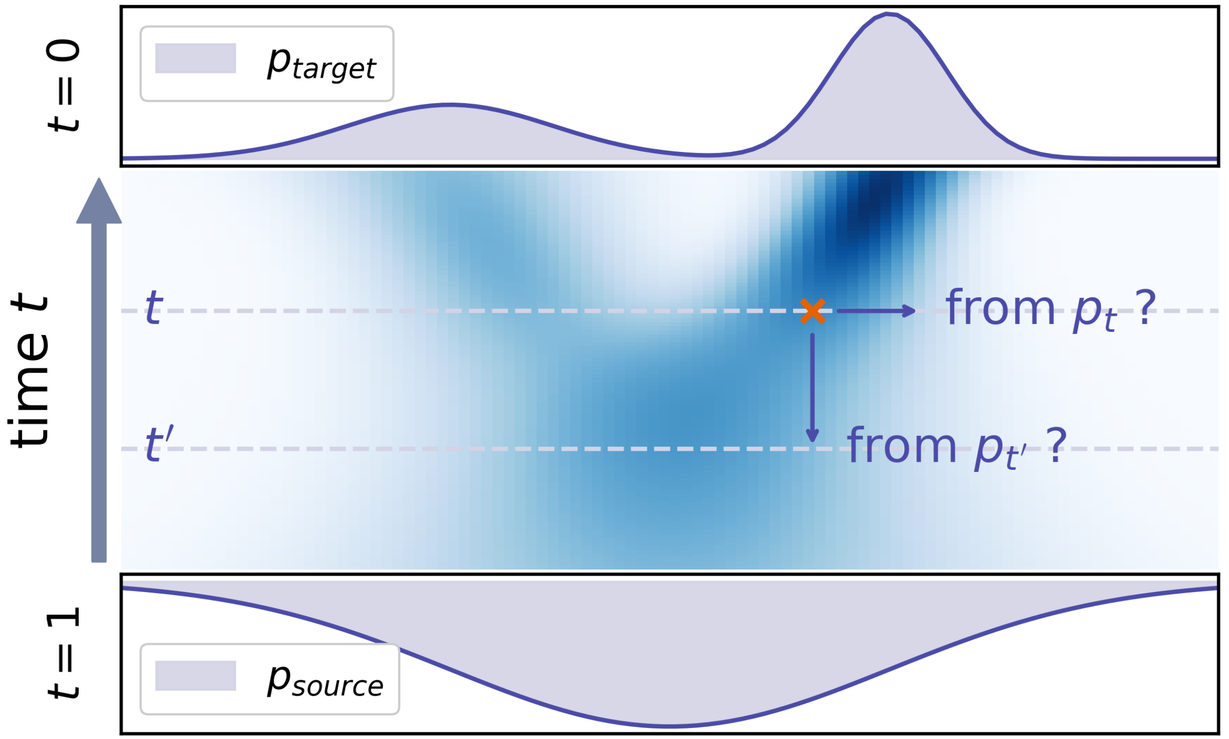
- The binary case
  - randomly sample (t, t')



Diffusive CLassiFication

A binary case illustration

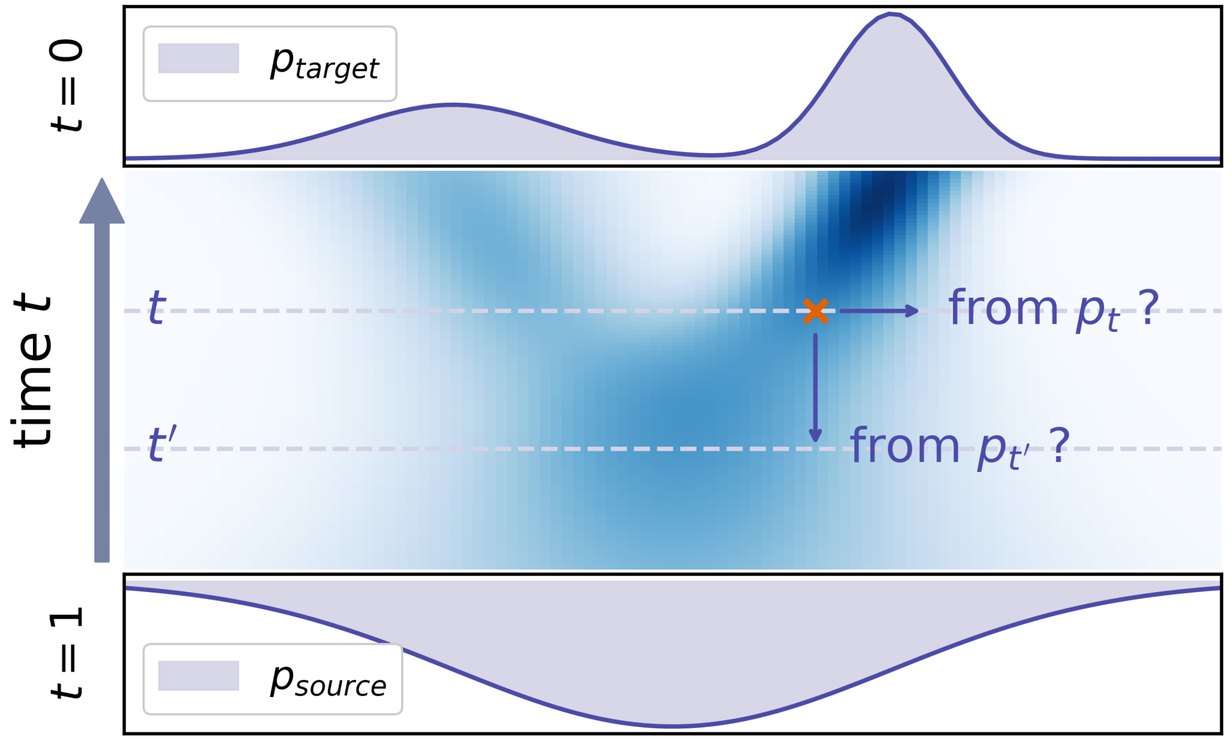
- The binary case
  - randomly sample (t, t')
  - sample $x_t \sim p_t$ and classify if it comes from p_t^θ or $p_{t'}^\theta$



Diffusive CLassiFication

A binary case illustration

- The binary case
  - randomly sample (t, t')
  - sample $x_t \sim p_t$ and classify if it comes from p_t^θ or $p_{t'}^\theta$
  - sample $x_{t'} \sim p_{t'}$ and classify if it comes from p_t^θ or $p_{t'}^\theta$



DiffCLF + DSM

Improved energy-based generative model training

DiffCLF + DSM

Improved energy-based generative model training

- We train jointly with DiffCLF and DSM

DiffCLF + DSM

Improved energy-based generative model training

- We train jointly with DiffCLF and DSM
- DiffCLF+DSM —> unique optimum: $p_t^{\theta*}(y) = p_t(y), \forall t$

DiffCLF + DSM

Improved energy-based generative model training

- We train jointly with DiffCLF and DSM
- DiffCLF+DSM —> unique optimum: $p_t^{\theta*}(y) = p_t(y), \forall t$
- DiffCLF+DSM —> consistent optimisation problem (convergence guarantee)

DiffCLF + DSM

Improved energy-based generative model training

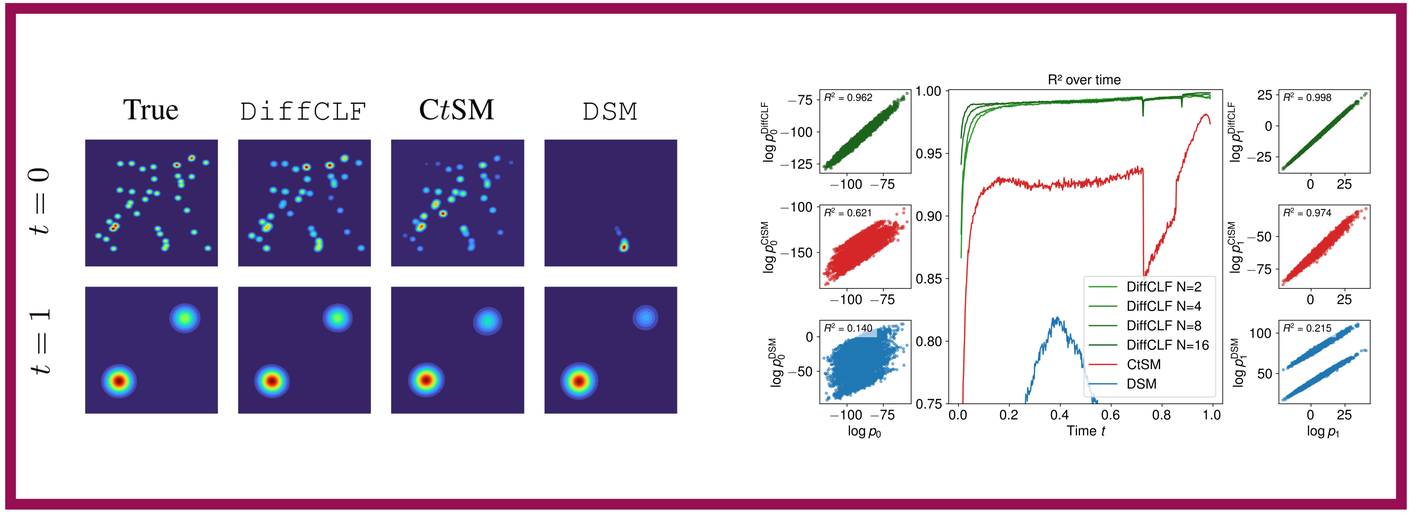
- We train jointly with DiffCLF and DSM
- DiffCLF+DSM —> unique optimum: $p_t^{\theta*}(y) = p_t(y), \forall t$
- DiffCLF+DSM —> consistent optimisation problem (convergence guarantee)
- Infinitesimal behaviour of binary DiffCLF

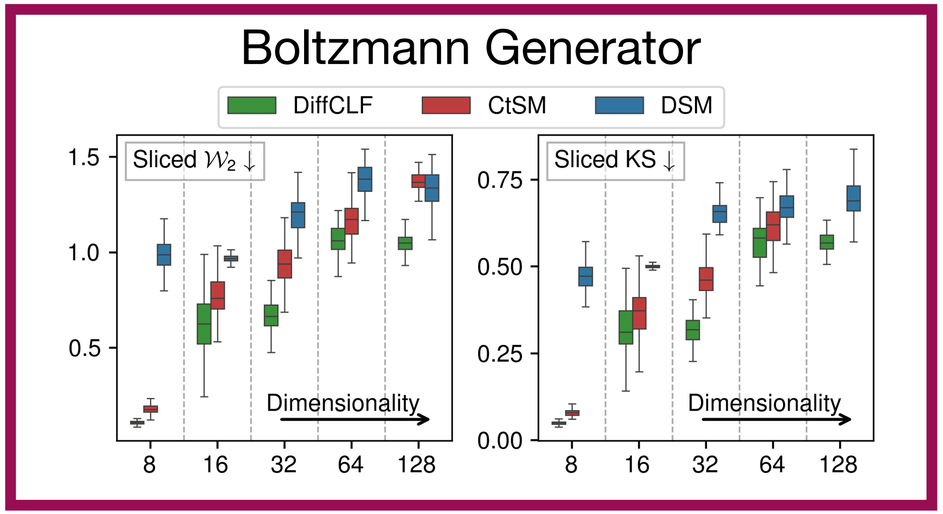
DiffCLF + DSM

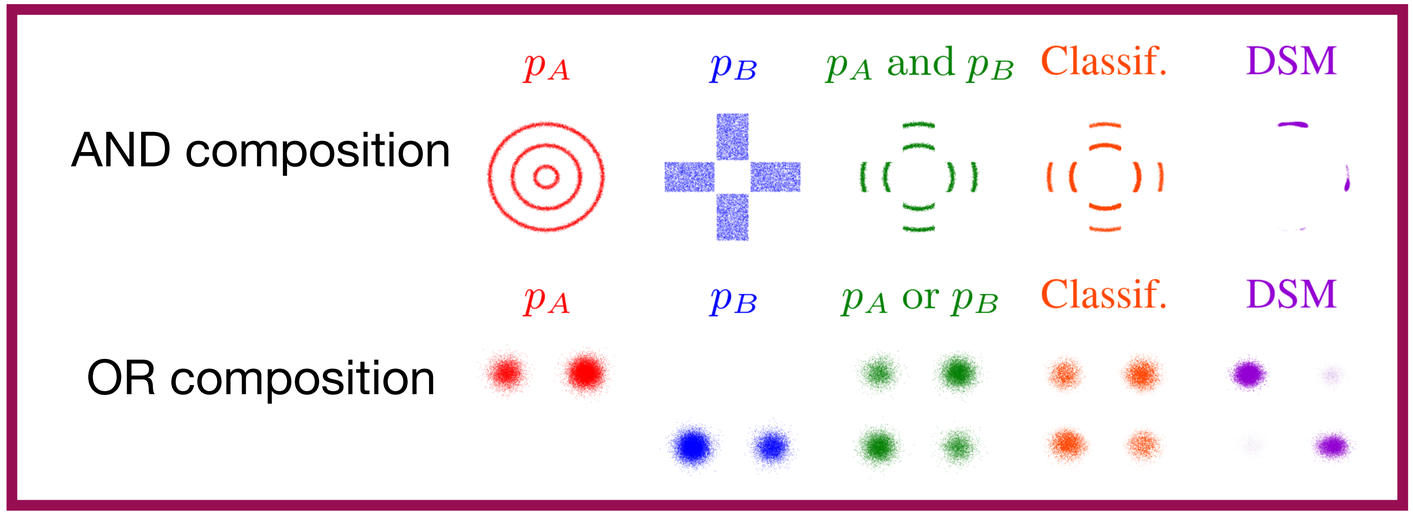
Improved energy-based generative model training

- We train jointly with DiffCLF and DSM
- DiffCLF+DSM —> unique optimum: $p_t^{\theta*}(y) = p_t(y), \forall t$
- DiffCLF+DSM —> consistent optimisation problem (convergence guarantee)
- Infinitesimal behaviour of binary DiffCLF
  - $$\lim_{\delta \to 0^+} \frac{8}{\delta^2} \left(\mathscr{L}_{\text{clf}}(\theta; t, t+\delta) - \log 2 \right) = \mathbb{E}\left[\left(\partial_t \log p_t^\theta(Y_t) - \partial_t \log p_t(Y_t) \right)^2 \right] + C$$

Experiments



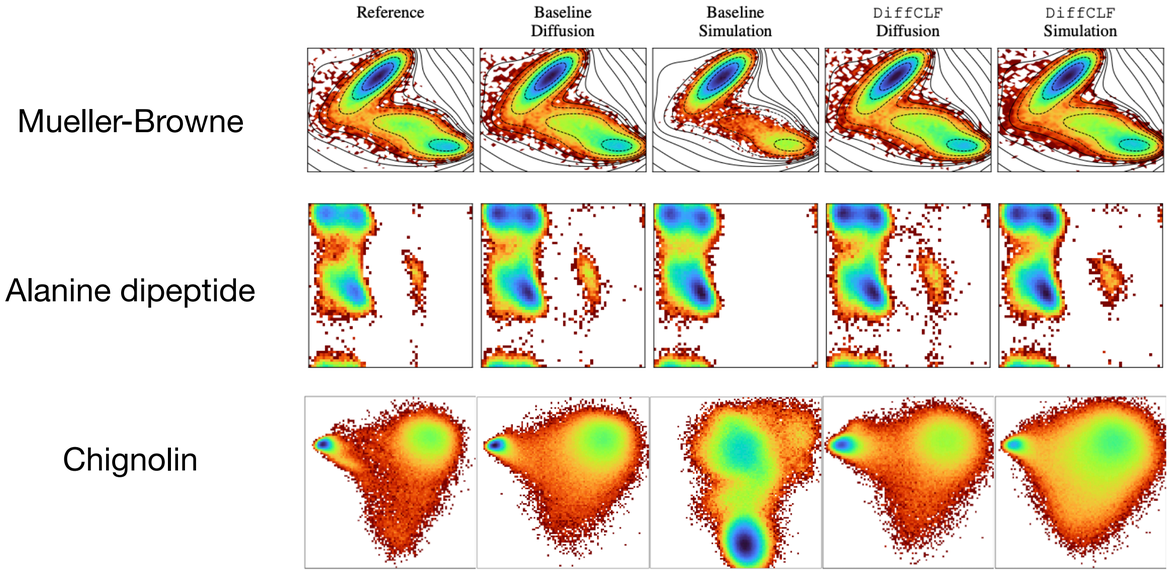




Experiments

4. Molecules estimation

System	Method	IID JS	Langevin JS	IID PMF	Langevin PMF	Train time (GPU hrs)
ALDP	DSM	0.0081±0.0003	0.0695±0.0517	0.095±0.003	1.047±0.924	3.3
	FPE	0.0082±0.0002	0.0090±0.0006	0.098±0.003	0.104±0.004	8.1
	DiffCLF	0.0068±0.0001	0.0092±0.0002	0.070±0.002	0.094±0.001	5.6
Chignolin	DSM	0.0036±0.0001	0.4351±0.0141	0.027±0.000	63.804±0.372	8.7
	FPE	0.0048±0.0001	0.0050±0.0001	0.037±0.000	0.039±0.001	49.6
	DiffCLF	0.0073±0.0019	0.0181±0.0055	0.060±0.015	0.162±0.053	18.9



Free energy difference estimation

Table 2. ALDP solvation free energy estimated with thermodynamic integration.

Method	Estimation
Reference	29.43±0.01
TI w/ $\mathcal{L}_{\text{base}}$ (Máté et al., 2025)	27.30±0.45
TI w/ $\mathcal{L}_{\text{base}} + \mathcal{L}_{\text{clf}}$ (ours)	**29.02**±0.41

Conclusion

Conclusion

DiffCLF

- Efficient ✅
- Unique optimum ✅
- Consistent optimization ✅
- Non-blind to mode weights ✅
- Improvement across different applications ✅

Conclusion

DiffCLF

Future works

- Large scale experiments
  - Images
  - proteins
- Other data space
  - discrete
  - Riemanian manifold