Discrete Neural Flow Samplers with Locally Equivariant Transformer

NeurIPS 2025

Zijing Ou[1], Ruixiang Zhang[2], Yingzhen Li[1]

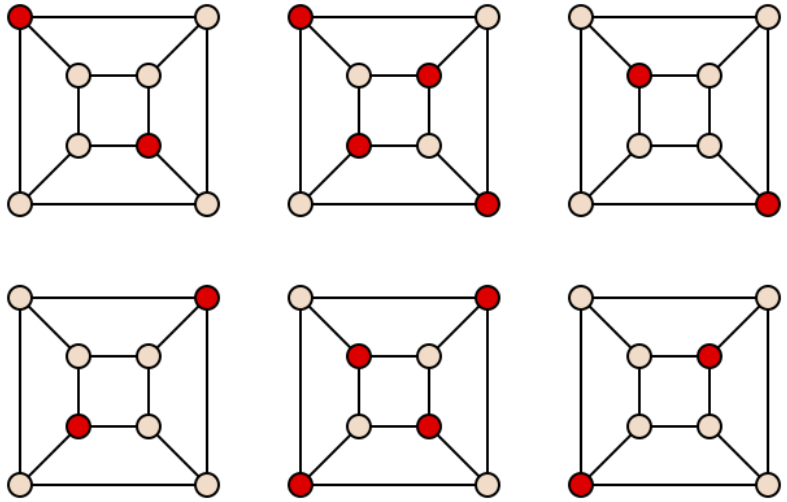
[1]Imperial College London, [2]Apple,

IMPERIAL

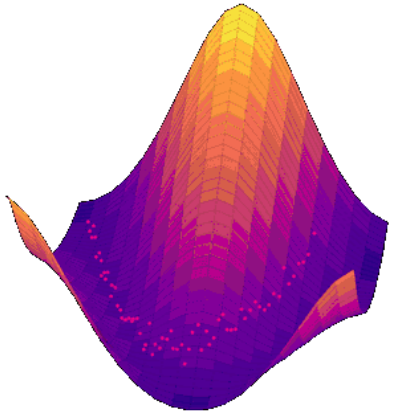
Sampling from Discrete Energy Distribution

Task: sample from $\pi(x) := \frac{1}{Z}\exp[-E(x)]$

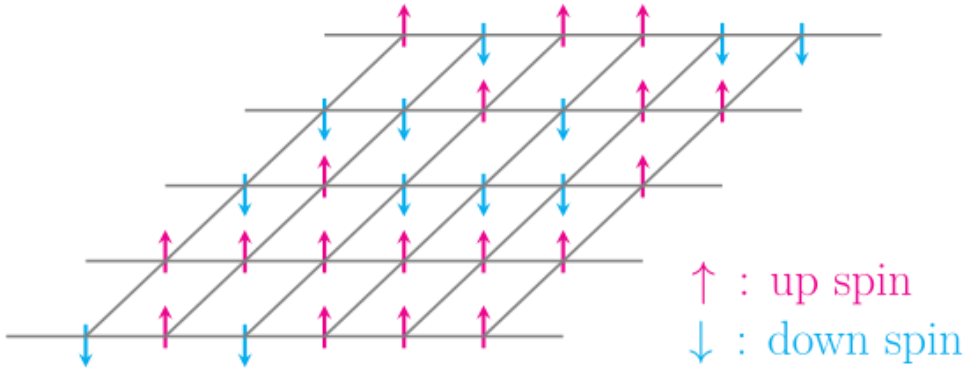
$$Z := \sum_x \exp[-E(x)] \qquad (x \in \{0,1,\dots\}^d)$$



Combinatorial Optimisation



Energy-based Models

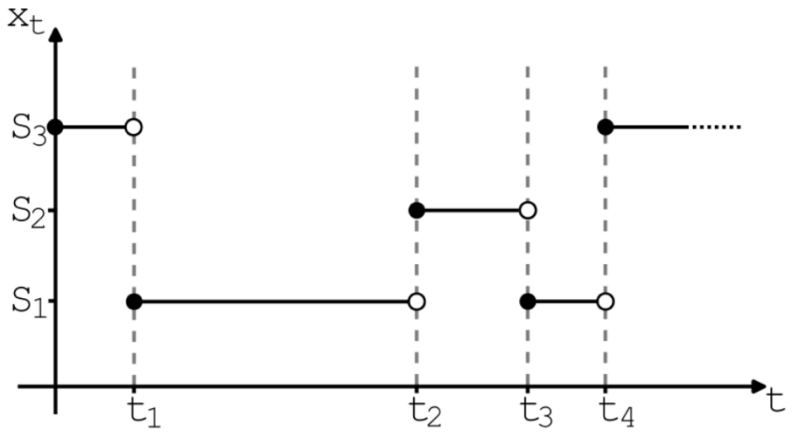


Ising Spin Models

Continuous Time Markov Chain (CTMC)

- Rate matrix characterises a discrete distribution path

$$R_t(y,x): \mathcal{X} \times \mathcal{X} \to \mathbb{R} \qquad \begin{array}{l} R_t(y,x) \geq 0 \text{ if } y \neq x \\ R_t(x,x) = -\sum_{y \neq x} R_t(y,x) \end{array}$$



Continuous Time Markov Chain (CTMC)

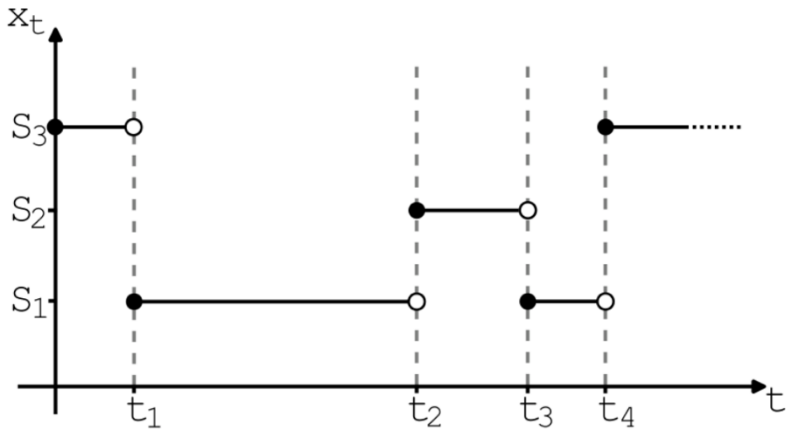
- Rate matrix characterises a discrete distribution path

$$R_t(y,x): \mathcal{X} \times \mathcal{X} \to \mathbb{R} \quad \begin{array}{l} R_t(y,x) \geq 0 \text{ if } y \neq x \\ R_t(x,x) = -\sum_{y \neq x} R_t(y,x) \end{array}$$

- Probability distribution evolves: $\{p_t(x)\}_{t\in[0,1]}$ satisfy

$$\partial_t p_t(x) = \sum_y R_t(x,y) p_t(y)$$

Kolmogorov equation



- Dividing both sides by $p_t\ (x)$ leads to:

$$\partial_t \log p_t(x_t) = R_t(x,y) \frac{p_t(y)}{p_t(x)} - \sum_{y \neq x} R_t(y,x)$$

$$\pi(x) = \frac{1}{Z}\exp[-E(x)]$$

Discrete Neural Flow Sampler

E.g., tempering:
$E_t(x) = \beta_t E_0(x) + (1-\beta_t)E(x)$,
$\beta_0 = 1, \beta_1 = 0$

- Specify a probabilistic path:

$\{p_t(x)\}_{t\in[0,1]}$: $\quad p_t(x) = \frac{1}{Z_t}\exp[-E_t(x)]$,

$p_0(x)$ easy to sample, $\quad p_1(x) = \pi(x)$, i.e., $E_1(x) \coloneqq E(x)$

$$\pi(x) = \frac{1}{Z}\exp[-E(x)]$$

Discrete Neural Flow Sampler

E.g., tempering:
$E_t(x) = \beta_t E_0(x) + (1-\beta_t)E(x)$,
$\beta_0 = 1, \beta_1 = 0$

- Specify a probabilistic path:

$$\{p_t(x)\}_{t\in[0,1]}: \qquad p_t(x) = \frac{1}{Z_t}\exp[-E_t(x)],$$

$$p_0(x) \text{ easy to sample}, \qquad p_1(x) = \pi(x), \text{ i.e., } E_1(x) \coloneqq E(x)$$

- Learn a Rate matrix $R_t^\theta(y,x)$ by minimising L2 error:

$$L(R_t^\theta) \coloneqq E_{q_t(x)}[\| \partial_t \log p_t(x) + \sum_{y\neq x} R_t^\theta(y,x) - R_t^\theta(x,y)\frac{p_t(y)}{p_t(x)} \|_2^2]$$

Ensuring Kolmogorov equation to hold for every $x \sim q_t(x)$

$\pi(x) = \frac{1}{Z}\exp[-E(x)]$

Discrete Neural Flow Sampler

E.g., tempering:
$E_t(x) = \beta_t E_0(x) + (1-\beta_t)E(x)$,
$\beta_0 = 1, \beta_1 = 0$

- Specify a probabilistic path:

$$\{p_t(x)\}_{t\in[0,1]}: \qquad p_t(x) = \frac{1}{Z_t}\exp[-E_t(x)],$$

$$p_0(x) \text{ easy to sample}, \qquad p_1(x) = \pi(x), \text{ i.e., } E_1(x) \coloneqq E(x)$$

- Learn a Rate matrix $R_t^\theta(y,x)$ by minimising L2 error:

$$L(R_t^\theta) \coloneqq E_{q_t(x)}[\| \partial_t \log p_t(x) + \sum_{y\neq x} R_t^\theta(y,x) - R_t^\theta(x,y)\frac{p_t(y)}{p_t(x)} \|_2^2]$$

Ensuring Kolmogorov equation to hold for every $x \sim q_t(x)$

- Simulate samples from $\pi(x)$ (approximately) by:

$$x_0 \sim p_0(x), \qquad x_{t+\Delta t} \sim Cat(x; 1_{x_{t+\Delta t}=x_t} + R_{t+\Delta t}(x_{t+\Delta t}, x)\Delta t)$$

$\pi(x) = \frac{1}{Z}\exp[-E(x)]$

Discrete Neural Flow Sampler

$= \exp(E_t(x) - E_t(y))$

Training: $L(R_t^\theta) \coloneqq E_{q_t(x)}[\| \partial_t \log p_t(x) + \sum_{y \neq x} R_t^\theta(y,x) - R_t^\theta(x,y)\frac{p_t(y)}{p_t(x)}\|_2^2]$

Sampling: $x_0 \sim p_0(x), \qquad x_{t+\Delta t} \sim Cat(x; 1_{x_{t+\Delta t}=x_t} + R_{t+\Delta t}(x_{t+\Delta t}, x)\Delta t)$

- Challenges:
  - Estimating $\partial_t \log p_t(x)$:

    $p_t(x) \coloneqq \frac{1}{Z_t}\exp[-E_t(x)] \quad \Rightarrow \quad \partial_t \log p_t(x) = -\partial_t E_t(x) - \partial_t \log Z_t$

    (intractable)
  - Intractable sum $\sum_{y \neq x} R_t^\theta(y,x) - R_t^\theta(x,y)$

    $R_t^\theta : \mathcal{X} \times \mathbb{R} \to \mathbb{R}^{|\mathcal{X}|}$ requires $\mathcal{O}(|\mathcal{X}|)$ NN forward to compute the sum

$$\pi(x) = \frac{1}{Z}\exp[-E(x)]$$

Estimating $\partial_t \log Z_t$

Monte Carlo estimate can have high variance!

$$\partial_t \log Z_t = \frac{1}{Z_t}\partial_t \sum_x \exp[-E_t(x)] = -\frac{1}{Z_t}\sum_x \exp[-E_t(x)]\,\partial_t E_t(x) = -E_{p_t(x)}[\partial_t E_t(x)]$$

- Solution: Discrete Stein control variate with $x^k \sim p_t(x)$

(Discrete-Stein Operator)

$$-E_{p_t(x)}[\partial_t E(x)] \approx \frac{1}{K}\sum_{k=1}^{K} -\partial_t E_t(x^k) + \beta\left[\sum_{y\neq x} R_t^\theta(y,x) - R_t^\theta(x,y)\frac{p_t(y)}{p_t(x)}\right]$$

$$\pi(x) = \frac{1}{Z}\exp[-E(x)]$$

Estimating $\partial_t \log Z_t$

Monte Carlo estimate can have high variance!

$$\partial_t \log Z_t = \frac{1}{Z_t}\partial_t \sum_x \exp[-E_t(x)] = -\frac{1}{Z_t}\sum_x \exp[-E_t(x)]\,\partial_t E_t(x) = -E_{p_t(x)}[\partial_t E_t(x)]$$

- Solution: Discrete Stein control variate with $x^k \sim p_t(x)$

(Discrete-Stein Operator)

$$-E_{p_t(x)}[\partial_t E(x)] \approx \frac{1}{K}\sum_{k=1}^{K} -\partial_t E_t(x^k) + \beta\left[\sum_{y\neq x} R_t^\theta(y,x) - R_t^\theta(x,y)\frac{p_t(y)}{p_t(x)}\right]$$

- Stein's Identity ensures unbiasedness: for any R_t^θ

$$E_{p_t(x)}\left[\sum_{y\neq x} R_t^\theta(y,x) - R_t^\theta(x,y)\frac{p_t(y)}{p_t(x)}\right] = 0$$

- Kolmogorov equation for *globally optimal* $R_t^\theta(x,t)$

$$\underbrace{-\partial_t E_t(x) + \left[\sum_{y\neq x} R_t^\theta(y,x) - R_t^\theta(x,y)\frac{p_t(y)}{p_t(x)}\right]}_{:=\,\xi_t(x;R_t^\theta)} = \partial_t \log Z_t$$

⇒ Variance is 0 when $\beta = 1$ and $R_t^\theta(x,t)$ is optimal

$$\pi(x) = \frac{1}{Z}\exp[-E(x)]$$

Estimating $\partial_t \log Z_t$

Monte Carlo estimate can have high variance!

$$\partial_t \log Z_t = \frac{1}{Z_t}\partial_t \sum_x \exp[-E_t(x)] = -\frac{1}{Z_t}\sum_x \exp[-E_t(x)]\,\partial_t E_t(x) = -E_{p_t(x)}[\partial_t E_t(x)]$$

- Solution: Discrete Stein control variate with $x^k \sim p_t(x)$

(Discrete-Stein Operator)

$$-E_{p_t(x)}[\partial_t E(x)] \approx \frac{1}{K}\sum_{k=1}^{K} -\partial_t E_t(x^k) + \beta\left[\sum_{y\neq x} R_t^\theta(y,x) - R_t^\theta(x,y)\frac{p_t(y)}{p_t(x)}\right]$$

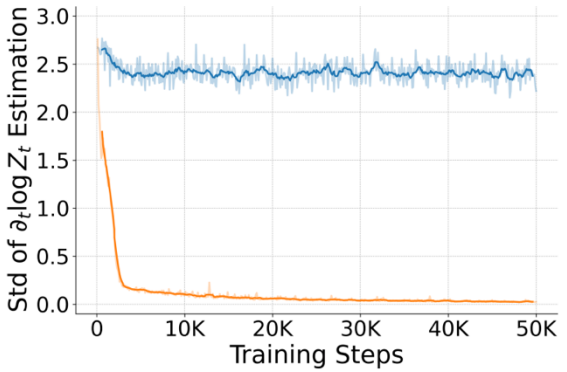
- Stein's Identity ensures unbiasedness: for any R_t^θ

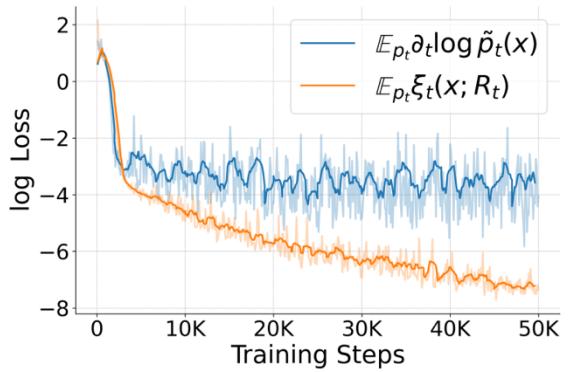
$$E_{p_t(x)}\left[\sum_{y\neq x} R_t^\theta(y,x) - R_t^\theta(x,y)\frac{p_t(y)}{p_t(x)}\right] = 0$$

- Kolmogorov equation for *globally optimal* $R_t^\theta(x,t)$

$$\underbrace{-\partial_t E_t(x) + \left[\sum_{y\neq x} R_t^\theta(y,x) - R_t^\theta(x,y)\frac{p_t(y)}{p_t(x)}\right]}_{:=\,\xi_t(x;R_t^\theta)} = \partial_t \log Z_t$$

⇒ Variance is 0 when $\beta = 1$ and $R_t^\theta(x,t)$ is optimal





Efficient Training with LeNets

Naïve summation requires $\mathcal{O}(|\mathcal{X}|)$ network evaluation!

Training: $$L(R_t^\theta) \coloneqq E_{q_t(x)}[\| \partial_t \log p_t(x) + \sum_{y \neq x} R_t^\theta(y,x) - R_t^\theta(x,y)\frac{p_t(y)}{p_t(x)} \|_2^2]$$

Locally Equivariant Networks (LeNets)

$$G_t^\theta(\tau, i|x) = -G_t^\theta(x_i, i|\mathrm{Swap}(x,i,\tau))$$

$\tau \in \{1, \dots, S\}$

$\mathrm{Swap}(x,i,\tau) = (x_1, \dots, x_i = \tau, \dots, x_d)$

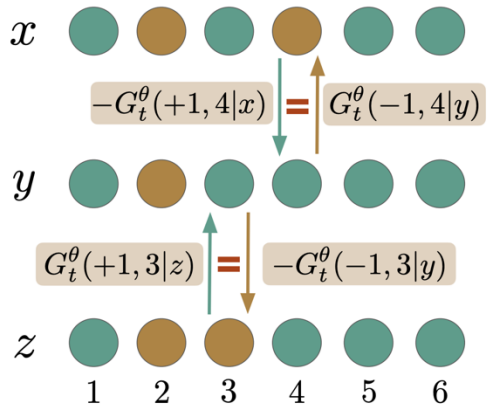


Image source: Holderrieth et al. 2025

Efficient Training with LeNets

Naïve summation requires $\mathcal{O}(|\mathcal{X}|)$ network evaluation!

Training: $$L(R_t^\theta) \coloneqq E_{q_t(x)}[\| \partial_t \log p_t(x) + \sum_{y \neq x} R_t^\theta(y,x) - R_t^\theta(x,y) \frac{p_t(y)}{p_t(x)} \|_2^2]$$

Locally Equivariant Networks (LeNets)

$$G_t^\theta(\tau, i|x) = -G_t^\theta(x_i, i|\mathrm{Swap}(x,i,\tau))$$

$\tau \in \{1, \dots, S\}$

$\mathrm{Swap}(x,i,\tau) = (x_1, \dots, x_i = \tau, \dots, x_d)$

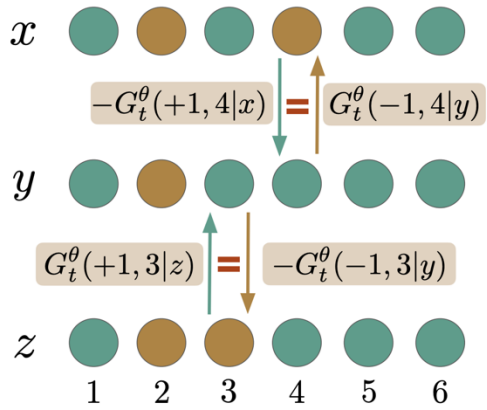


Image source: Holderrieth et al. 2025

$$L(R_t^\theta) \coloneqq E_{q_t(x)}[\| \partial_t \log p_t(x) + \sum_{i, y_i \neq x_i} [G_t^\theta(y_i, i|x)]_+ - [-G_t^\theta(y_i, i|x)]_+ \frac{p_t(y)}{p_t(x)} \|_2^2]$$

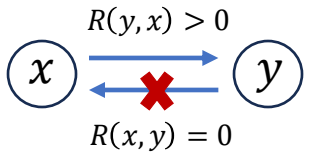
Summation requires only $\mathcal{O}(1)$ network evaluation!

Efficient Training with LeNets

$$L(R_t^\theta) \coloneqq E_{q_t(x)}[\| \partial_t \log p_t(x) + \sum_{i, y_i \neq x_i} \left[G_t^\theta(y_i, i|x)\right]_+ - \left[-G_t^\theta(y_i, i|x)\right]_+ \frac{p_t(y)}{p_t(x)} \|_2^2]$$

One-way Rate Matrix

- A rate matrix R is one-way if $R(y, x) > 0$ implies $R(x, y) = 0$.

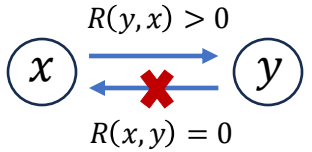


Efficient Training with LeNets

$$L(R_t^\theta) \coloneqq E_{q_t(x)}[\| \partial_t \log p_t(x) + \sum_{i, y_i \neq x_i} \left[G_t^\theta(y_i, i|x)\right]_+ - \left[-G_t^\theta(y_i, i|x)\right]_+ \frac{p_t(y)}{p_t(x)} \|_2^2]$$

One-way Rate Matrix

- A rate matrix R is one-way if $R(y, x) > 0$ implies $R(x, y) = 0$.



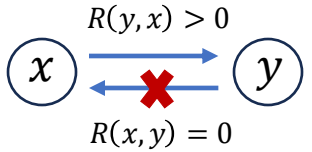
For a rate matrix R_t that generates the probabilistic path p_t, there exists a one-way rate matrix Q_t that generates the same path.

Efficient Training with LeNets

$$L(R_t^\theta) \coloneqq E_{q_t(x)}[\| \partial_t \log p_t(x) + \sum_{i, y_i \neq x_i} \left[G_t^\theta(y_i, i|x)\right]_+ - \left[-G_t^\theta(y_i, i|x)\right]_+ \frac{p_t(y)}{p_t(x)} \|_2^2]$$

One-way Rate Matrix

- A rate matrix R is one-way if $R(y, x) > 0$ implies $R(x, y) = 0$.



For a rate matrix R_t that generates the probabilistic path p_t, there exists a one-way rate matrix Q_t that generates the same path.

One-way Parametrisation: The matrix $R_t^\theta(\tau, i|x) = \left[G_t^\theta(\tau, i|x)\right]_+$ is a one-way rate matrix, where $[z]_+ = \max(z, 0)$ denotes the ReLU operation.

Locally Equivariant Transformer (LeTF)

$$L(R_t^\theta) \coloneqq E_{q_t(x)}[\| \partial_t \log p_t(x) + \sum_{i,y_i \neq x_i} \left[G_t^\theta(y_i, i|x)\right]_+ - \left[-G_t^\theta(y_i, i|x)\right]_+ \frac{p_t(y)}{p_t(x)} \|_2^2]$$

Instantiation of LeNets

$$G(\tau, i|x) = \left(\omega_\tau - \omega_{x_i}\right)^T H(x)_{i,:}$$

Token embedding

Hollow network: the i-th output is independent from the i-th input

$H(x_{i\leftarrow\tau})_{i,:} = H(x_{i\leftarrow\tau'})_{i,:}, \forall \tau, \tau'$

$G(\tau, i|x)$ is locally equivariant

$$\begin{aligned} G(\tau, i|x) &= -\left(\omega_{x_i} - \omega_\tau\right)^T H(\mathrm{Swap}(x, i, \tau))_{i,:} \\ &= -G(x_i, i|\mathrm{Swap}(x, i, \tau)) \end{aligned}$$

Locally Equivariant Transformer (LeTF)

$$L(R_t^\theta) \coloneqq E_{q_t(x)}[\| \partial_t \log p_t(x) + \sum_{i,y_i \neq x_i} [G_t^\theta(y_i, i|x)]_+ - [-G_t^\theta(y_i, i|x)]_+ \frac{p_t(y)}{p_t(x)} \|_2^2]$$

Instantiation of LeNets

$$G(\tau, i|x) = (\omega_\tau - \omega_{x_i})^T H(x)_{i,:}$$

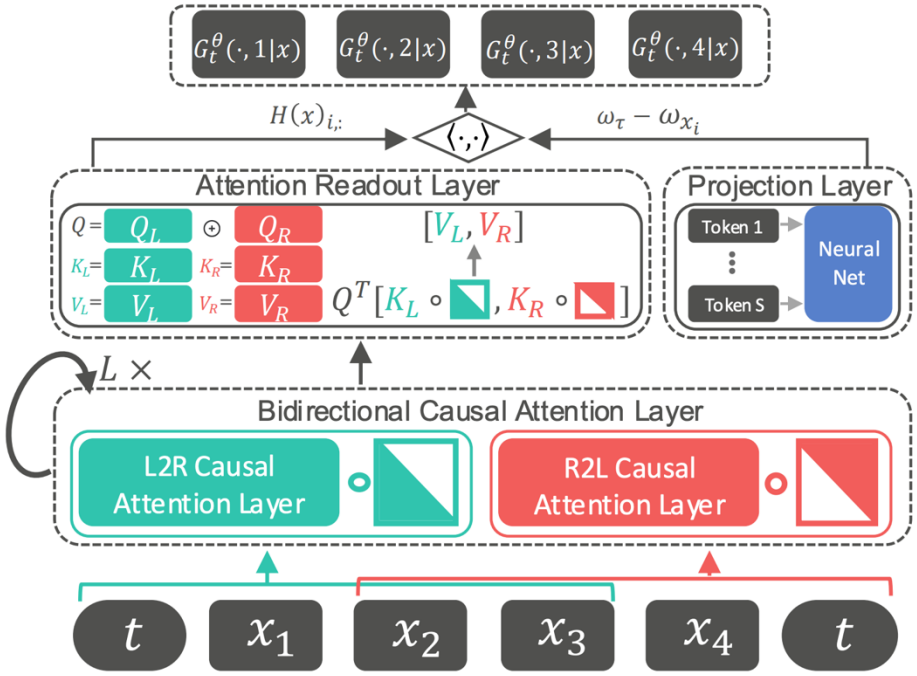
Token embedding

Hollow network: the i-th output is independent from the i-th input

$H(x_{i\leftarrow\tau})_{i,:} = H(x_{i\leftarrow\tau'})_{i,:}, \forall \tau, \tau'$

$G(\tau, i|x)$ is locally equivariant

$$\begin{aligned} G(\tau, i|x) &= -(\omega_{x_i} - \omega_\tau)^T H(\mathrm{Swap}(x, i, \tau))_{i,:} \\ &= -G(x_i, i|\mathrm{Swap}(x, i, \tau)) \end{aligned}$$



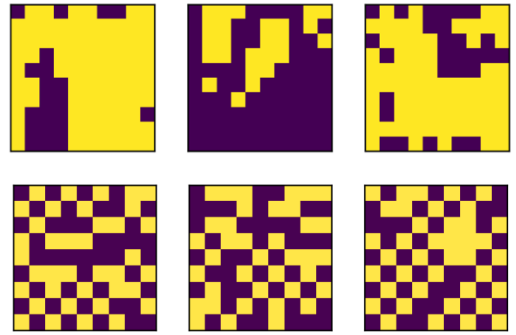
Sampling from Ising Models

$$\pi(x) \propto \exp(-\sigma x^T A x), \; x \in \{-1, 1\}^D,$$

Adjacency matrix of the lattice grid

$D = 10 \times 10$
$\sigma = 0.2$

$D = 10 \times 10$
$\sigma = -0.2$



$\sigma = 0.1$

	ESS	Free Energy	Internal Energy	Entropy
Optimal Value	1	-2.12417	-1.47632	0.28550
DNFS	0.9685 ± 0.0010	-2.11201 ± 6.6258e-05	-1.47425 ± 0.0068	0.28106 ± 0.0030
LEAPS	0.3631 ± 0.1184	-2.10111 ± 0.0002	-1.44926 ± 0.0130	0.28727 ± 0.0057

$\sigma = 0.22305$

	ESS	Free Energy	Internal Energy	Entropy
Optimal Value	1	-2.12417	-1.47632	0.28550
DNFS	0.9685 ± 0.0010	-2.11201 ± 6.6258e-05	-1.47425 ± 0.0068	0.28106 ± 0.0030
LEAPS	0.3631 ± 0.1184	-2.10111 ± 0.0002	-1.44926 ± 0.0130	0.28727 ± 0.0057

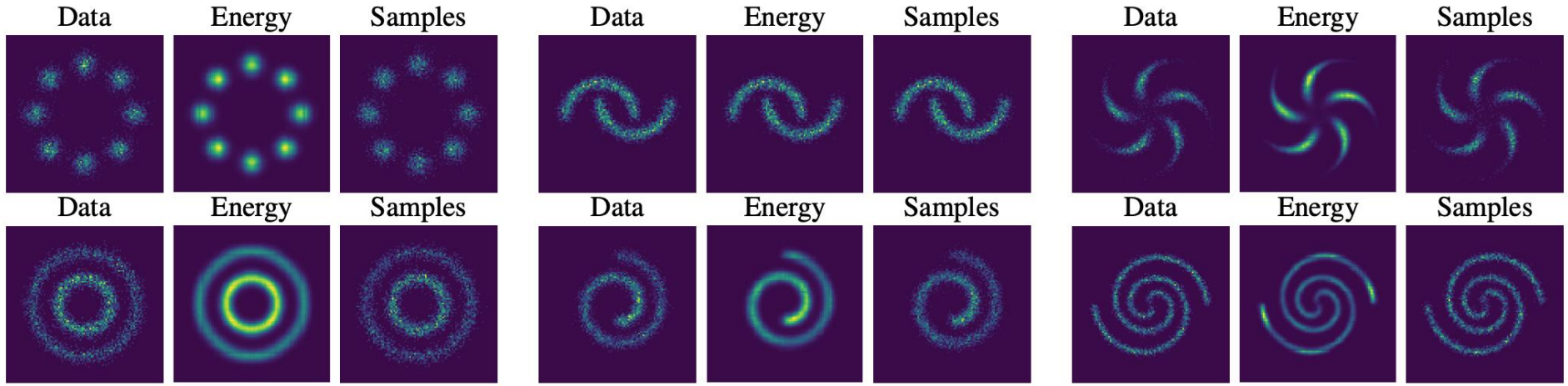
	LEAPS	DNFS (ours)
Different Perspective	Log-variance Importance Weight	Kolmogorov Forward Equation
$\partial_t \log Z_t$ Estimation	Learnable Neural Network	Monte Carlo Control Variates
Architecture	Locally Equivariant Convolutional Network	Locally Equivariant Transformer

Training Discrete EBMs

$$\nabla_\phi \log p_\phi(x) = E_{p_\phi(y)}\left[\nabla_\phi E(y)\right] - \nabla_\phi E(x),$$

$$E_{p_\phi(x)}\left[\nabla_\phi E(x)\right] = \sum_{k=1}^{K} \frac{\exp(w^{(k)})}{\sum_{j=1}^{K} \exp(w^{(j)})} \nabla_\phi E(x), \; w^{(k)} = \int_0^1 \xi_t(x_t; R_t^\theta)\, dt$$

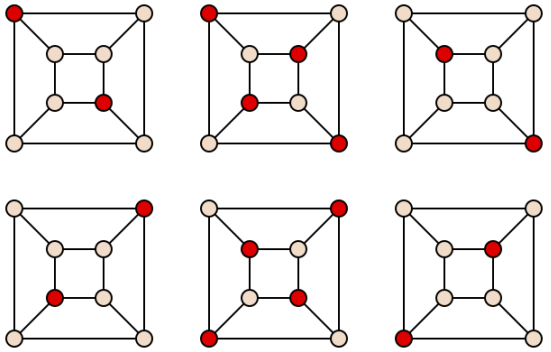
$$\xi_t\left(x_t; R_t^\theta\right) = -\partial_t E_t(x) + \left[\sum_{y \neq x} R_t^\theta(y, x) - R_t^\theta(x, y) \frac{p_t(y)}{p_t(x)}\right]$$



Solving Combinatorial Optimisation Problems

Target of Maximum Independent Set

$$\pi(x) \propto \exp\left(\frac{1}{T}\left(\sum_{i=1}^{|V|} x_i - \lambda \sum_{(i,j\in E)} x_i x_j\right)\right),\ T \to 0$$



Method	ER16-20			ER32-40			ER64-75		
	Size ↑	Drop ↓	Time ↓	Size ↑	Drop ↓	Time ↓	Size ↑	Drop ↓	Time ↓
Gurobi	8.92	0.00%	4:00	14.62	0.00%	4:03	20.55	0.00%	4:10
Random	5.21	41.6%	0:03	6.31	56.8%	0:06	8.63	58.0%	0:09
DMALA	8.81	1.23%	0:21	14.02	4.10%	0:22	19.54	4.91%	0:24
GFlowNet	8.75	1.91%	0:02	13.93	4.72%	0:04	19.13	6.91%	0:07
DNFS	8.28	7.17%	0:03	13.18	9.85%	0:06	18.12	11.8%	0:09
DNFS+DMALA	8.91	0.11%	0:10	14.31	2.12%	0:15	20.06	2.38%	0:22

Thank you for watching

- **Contact:** Zijing Ou (z.ou22@imperial.ac.uk)
- **Code:** https://github.com/J-zin/DNFS
- **Poster time:** 2025/12/03, 4:30pm-7:30pm

IMPERIAL