Title: Machine Learning Renormalization Group (VIRTUAL)

Speakers: Yi-Zhuang You

Series: Machine Learning Initiative

Date: May 24, 2024 - 2:30 PM

URL: https://pirsa.org/24050091

Abstract: We develop a Machine-Learning Renormalization Group (MLRG) algorithm to explore and analyze many-body lattice models in statistical physics. Using the representation learning capability of generative modeling, MLRG automatically learns the optimal renormalization group (RG) transformations from self-generated spin configurations and formulates RG equations without human supervision. The algorithm does not focus on simulating any particular lattice model but broadly explores all possible models compatible with the internal and lattice symmetries given the on-site symmetry representation. It can uncover the RG monotone that governs the RG flow, assuming a strong form of the c-theorem. This enables several downstream tasks, including unsupervised classification of phases, automatic location of phase transitions or critical points, controlled estimation of critical exponents, and operator scaling dimensions. We demonstrate the MLRG method in two-dimensional lattice models with Ising symmetry and show that the algorithm correctly identifies and characterizes the Ising criticality.

Zoom link


Machine Learning
Renormalization Group
arXiv:2306.11054
Yi-Zhuang You 尤亦庄 (UCSD)
Perimeter - 2024.04
Art by Midjourney

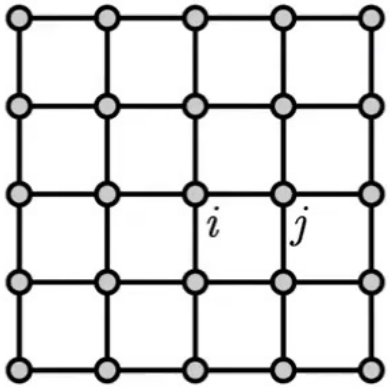
Motivation

- Renormalization Group (RG) is
  - An elegant conceptual framework to understand phase and phase transition
  - A powerful computational method to identify critical point and estimate critical exponents
- It iteratively coarsening the degrees of freedom in physical systems, extracting relevant features at every scale, and tracking the flow of effective theories.
  - How to identify relevant features?
  - What terms should be included in the effective theory?
  - How to explore the space of possible effective theories?
- **Machine Learning RG** (MLRG): let artificial intelligence (AI) to help us figure out these questions.

2D Ising Model

- Lattice: 2D square lattice
- Degrees of freedom: Ising spin $s_i = \pm 1$ (binary random variable) on each site
- Energy model:

$$E(s) = -J \sum_{\langle ij \rangle} s_i s_j$$



  - Model parameter: Ising coupling $J = 1/T$
- Probability distribution (Boltzmann distribution):

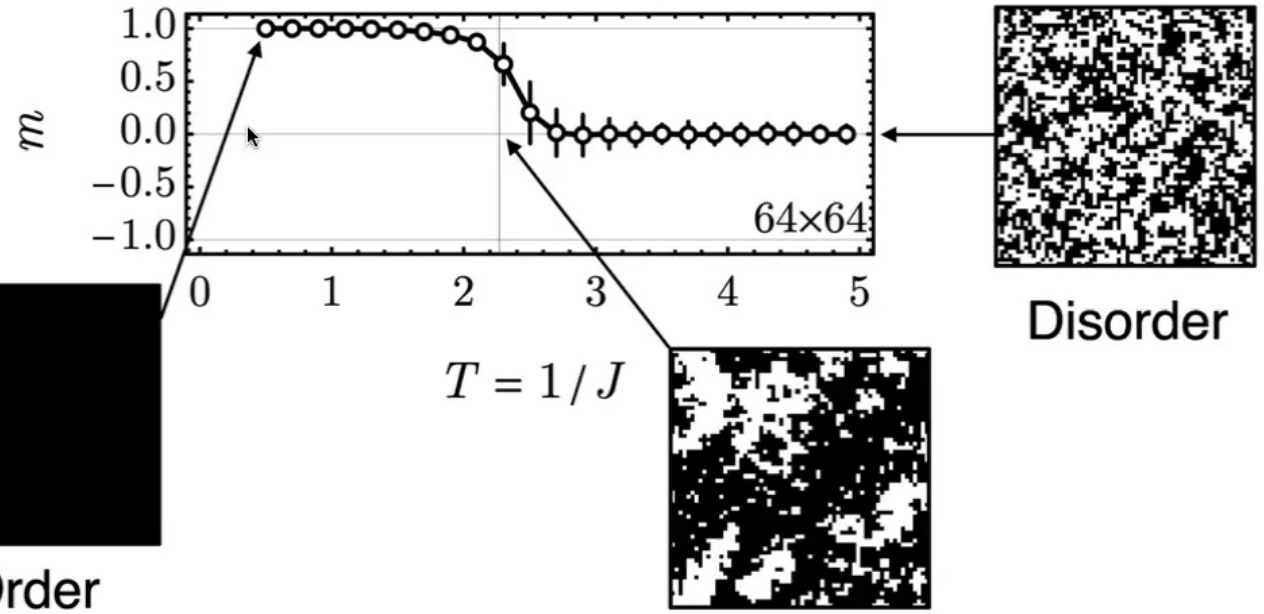
$$p(s) = \frac{1}{Z} \mathrm{e}^{-E(s)} = \frac{1}{Z} \prod_{\langle ij \rangle} \mathrm{e}^{J s_i s_j}$$

  - Partition function: $Z = \sum_{s} \mathrm{e}^{-E(s)}$

2D Ising Model

- Physical property: magnetization

$$m = \langle s_i \rangle = \sum_s s_i p(s)$$



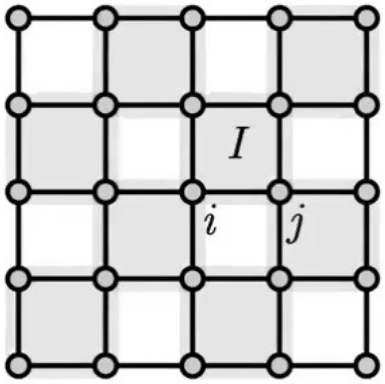
Real Space Renormalization Group

- Step 1: Block Partition
  (Partition the square lattice into 2x2 corner-sharing blocks)

$$E(s) = \sum_{\langle ij \rangle} (-J s_i s_j)$$

$$= \sum_I \sum_{\langle ij \rangle \in I} (-J s_i s_j)$$

$$= \sum_I E_I(s)$$



- Local energy model:

$$E_I(s) = -J \sum_{\langle ij \rangle \in I} s_i s_j$$

- Probability distribution can be decomposed as

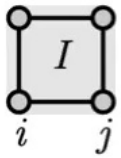
$$p(s) \propto \mathrm{e}^{-E(s)}$$

$$= \prod_I \mathrm{e}^{-E_I(s)}$$

$$\propto \prod_I p_I(s)$$

Real Space Renormalization Group

- Step 2: Feature Learning (Emergence of hidden variables)
  - The local energy model defines a local distribution

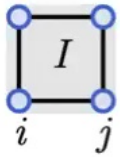


$$E_I(s) = -J \sum_{\langle ij \rangle \in I} s_i s_j \longrightarrow p_I(s) \propto \mathrm{e}^{-E_I(s)}$$

  - Generative modeling (with hidden/latent variables)
    - Unsupervised learning: learn to model probability distribution
    - Representation learning: learn to extract feature representations from data
    - Examples: Restricted Boltzmann Machine (RBM), Variational Auto-Encoder (VAE) …
  - Goal: model the local distribution, extract relevant features

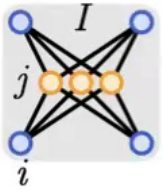
Real Space Renormalization Group

- Step 2: Feature Learning (Emergence of hidden variables)
  - The local energy model defines a local distribution



$$E_I(s) = -J \sum_{\langle ij \rangle \in I} s_i s_j \longrightarrow p_I(s) \propto \mathrm{e}^{-E_I(s)}$$

  - Restricted Boltzmann Machine
    - Introduce hidden variables $s'_j = \pm 1$ in each block
    - Model the local distribution by a new energy model defined on a bipartite graph between visible and hidden



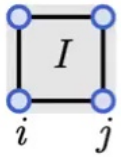
$$\tilde{E}_I(s, s') = - \sum_{i \in I_v, j \in I_h} w_{ij} s_i s'_j$$

$$\downarrow$$

$$\tilde{p}_I(s, s') \propto \mathrm{e}^{-\tilde{E}_I(s, s')}$$

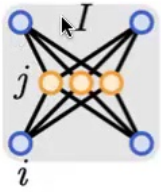
Real Space Renormalization Group

- Step 2: Feature Learning (Emergence of hidden variables)
  - The local energy model defines a local distribution



$$E_I(s) = -J \sum_{\langle ij \rangle \in I} s_i s_j \longrightarrow p_I(s) \propto \mathrm{e}^{-E_I(s)}$$

  - Restricted Boltzmann Machine



$$\tilde{E}_I(s, s') = -\sum_{i \in I_v, j \in I_h} w_{ij} s_i s'_j$$

$$\downarrow$$

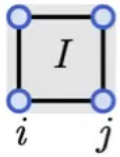
$$\tilde{p}_I(s, s') \propto \mathrm{e}^{-\tilde{E}_I(s, s')}$$

    - Objective: the model distribution marginalized to visible variables should approximate the target distribution

$$\tilde{p}_I(s) = \sum_{s'} \tilde{p}_I(s, s') \to p_I(s)$$

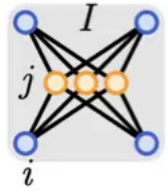
Real Space Renormalization Group

- Step 2: Feature Learning (Emergence of hidden variables)
  - The local energy model defines a local distribution



$$E_I(s) = -J \sum_{\langle ij \rangle \in I} s_i s_j \longrightarrow p_I(s) \propto \mathrm{e}^{-E_I(s)}$$

  - Restricted Boltzmann Machine



$$\tilde{E}_I(s, s') = -\sum_{i \in I_v, j \in I_h} w_{ij} s_i s'_j$$

$$\downarrow$$

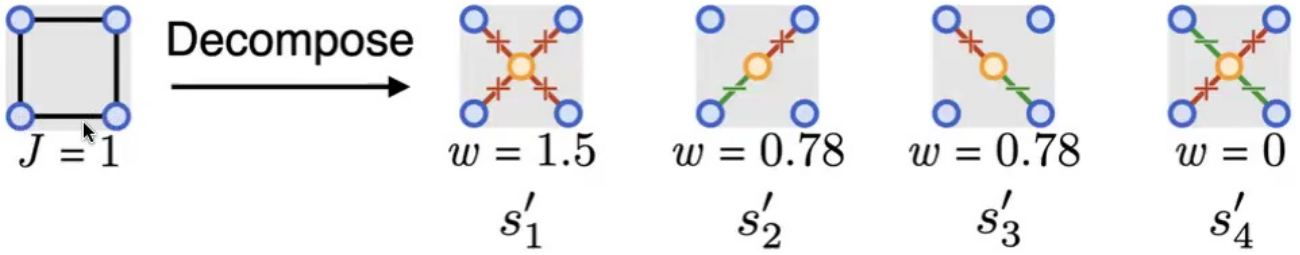
$$\tilde{p}_I(s, s') \propto \mathrm{e}^{-\tilde{E}_I(s,s')} \qquad \tilde{p}_I(s) = \sum_{s'} \tilde{p}_I(s, s') \to p_I(s)$$

  - Find the optimal w_{ij} to minimize the KL divergence

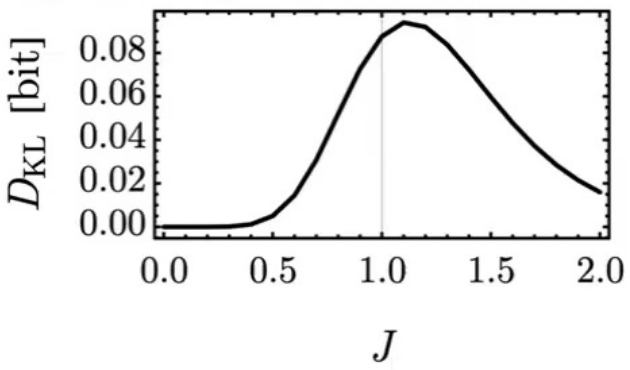
$$\mathcal{L} = D_{\mathrm{KL}}[p_I(s) \| \tilde{p}_I(s)] = \sum_s p_I(s) \log \frac{p_I(s)}{\tilde{p}_I(s)}$$

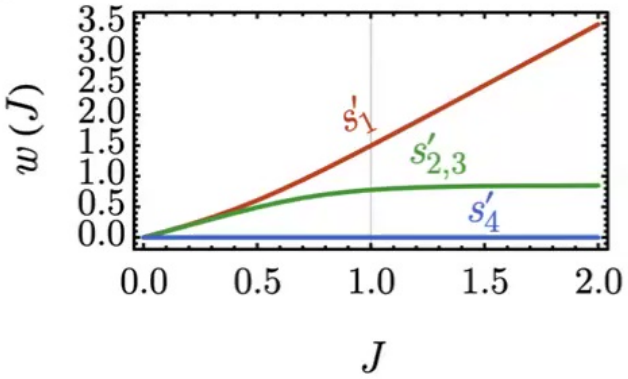
Real Space Renormalization Group

- Step 2: Feature Learning (Emergence of hidden variables)
  - After training, each hidden variable represents a feature (collective pattern) of visible variables



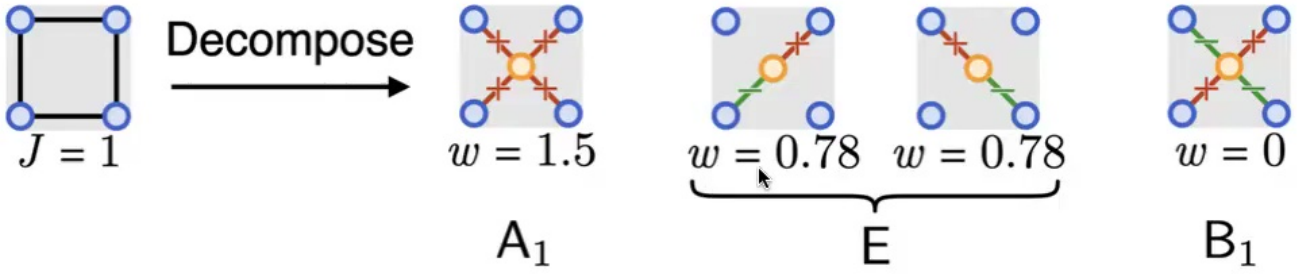
KL divergence: $D_{\mathrm{KL}} = 0.088\,\mathrm{bit}$





Real Space Renormalization Group

- Step 2: Feature Learning (Emergence of hidden variables)
  - After training, each hidden variable represents a feature (collective pattern) of visible variables

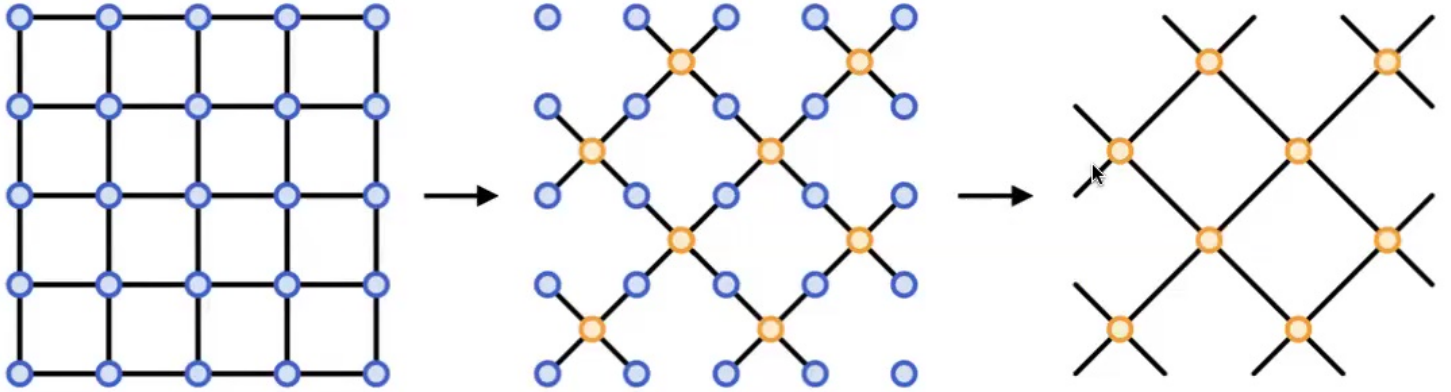


  - Classified by representations of the C_{4v} point group

irrep	dim	transforms as	T_{C_4}	T_σ
A_1	1	1	$\begin{bmatrix} 1 \end{bmatrix}$	$\begin{bmatrix} 1 \end{bmatrix}$
A_2	1	$xy(x^2 - y^2)$	$\begin{bmatrix} 1 \end{bmatrix}$	$\begin{bmatrix} -1 \end{bmatrix}$
B_1	1	$x^2 - y^2$	$\begin{bmatrix} -1 \end{bmatrix}$	$\begin{bmatrix} 1 \end{bmatrix}$
B_2	1	xy	$\begin{bmatrix} -1 \end{bmatrix}$	$\begin{bmatrix} -1 \end{bmatrix}$
E	2	(x, y)	$\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$

Real Space Renormalization Group

- Step 3: Marginalize visible variables (Extract effective model for hidden variables)



$p(s)$ → $\tilde{p}(s, s')$ → $\tilde{p}(s') = \sum_s \tilde{p}(s, s')$

Marginalize

Generate

$E(s)$ $\tilde{E}(s, s')$ $\tilde{E}(s')$

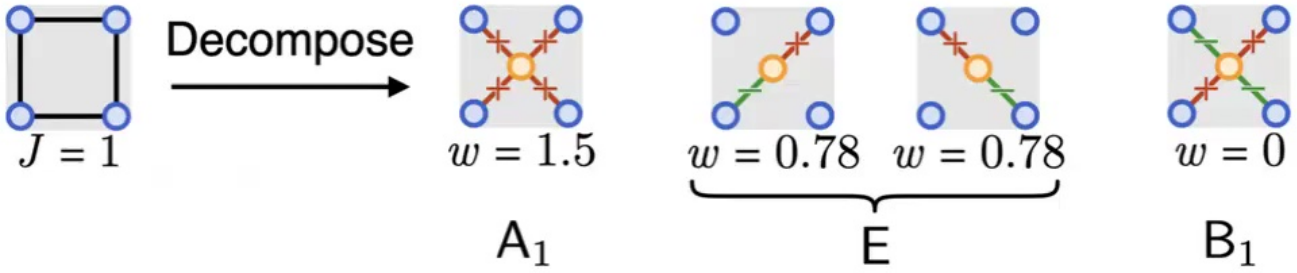
Extract effective model

Block partition $= \sum_I E_I(s)$ → $= \sum_I \tilde{E}_I(s, s')$

Feature learning

Real Space Renormalization Group

- Step 2: Feature Learning (Emergence of hidden variables)
  - After training, each hidden variable represents a feature (collective pattern) of visible variables

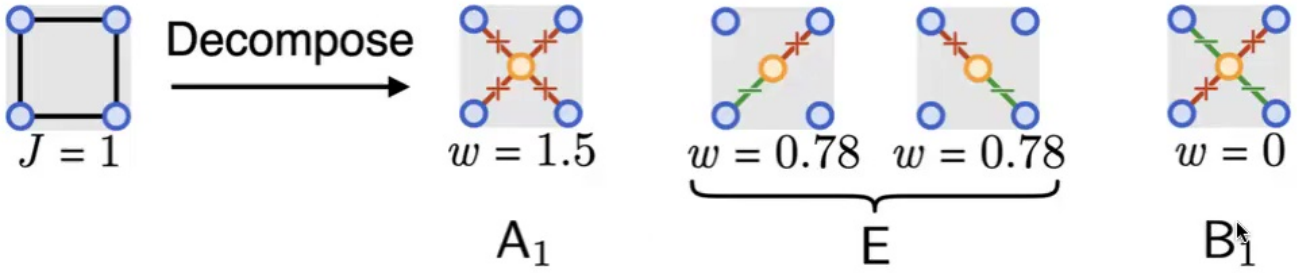


  - Classified by representations of the C_{4v} point group

irrep	dim	transforms as	T_{C_4}	T_σ
A_1	1	1	$\begin{bmatrix}1\end{bmatrix}$	$\begin{bmatrix}1\end{bmatrix}$
A_2	1	$xy(x^2-y^2)$	$\begin{bmatrix}1\end{bmatrix}$	$\begin{bmatrix}-1\end{bmatrix}$
B_1	1	x^2-y^2	$\begin{bmatrix}-1\end{bmatrix}$	$\begin{bmatrix}1\end{bmatrix}$
B_2	1	xy	$\begin{bmatrix}-1\end{bmatrix}$	$\begin{bmatrix}-1\end{bmatrix}$
E	2	(x, y)	$\begin{bmatrix}0 & -1\\ 1 & 0\end{bmatrix}$	$\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}$

Real Space Renormalization Group

- Step 2: Feature Learning (Emergence of hidden variables)
  - After training, each hidden variable represents a feature (collective pattern) of visible variables

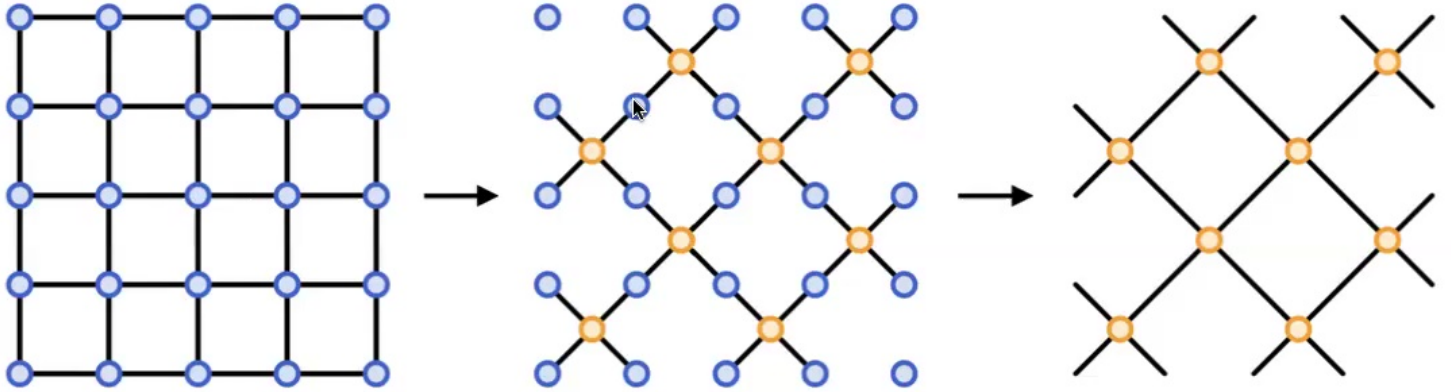


  - Classified by representations of the C_{4v} point group

irrep	dim	transforms as	T_{C_4}	T_σ
A_1	1	1	$\begin{bmatrix}1\end{bmatrix}$	$\begin{bmatrix}1\end{bmatrix}$
A_2	1	$xy(x^2-y^2)$	$\begin{bmatrix}1\end{bmatrix}$	$\begin{bmatrix}-1\end{bmatrix}$
B_1	1	x^2-y^2	$\begin{bmatrix}-1\end{bmatrix}$	$\begin{bmatrix}1\end{bmatrix}$
B_2	1	xy	$\begin{bmatrix}-1\end{bmatrix}$	$\begin{bmatrix}-1\end{bmatrix}$
E	2	(x, y)	$\begin{bmatrix}0 & -1\\ 1 & 0\end{bmatrix}$	$\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}$

Real Space Renormalization Group

- Step 3: Marginalize visible variables (Extract effective model for hidden variables)



$p(s)$ $\quad$ $\tilde{p}(s, s')$ $\longrightarrow$ $\tilde{p}(s') = \sum_s \tilde{p}(s, s')$

Marginalize

Generate

$E(s)$ $\quad$ $\tilde{E}(s, s')$ $\quad$ $\tilde{E}(s')$

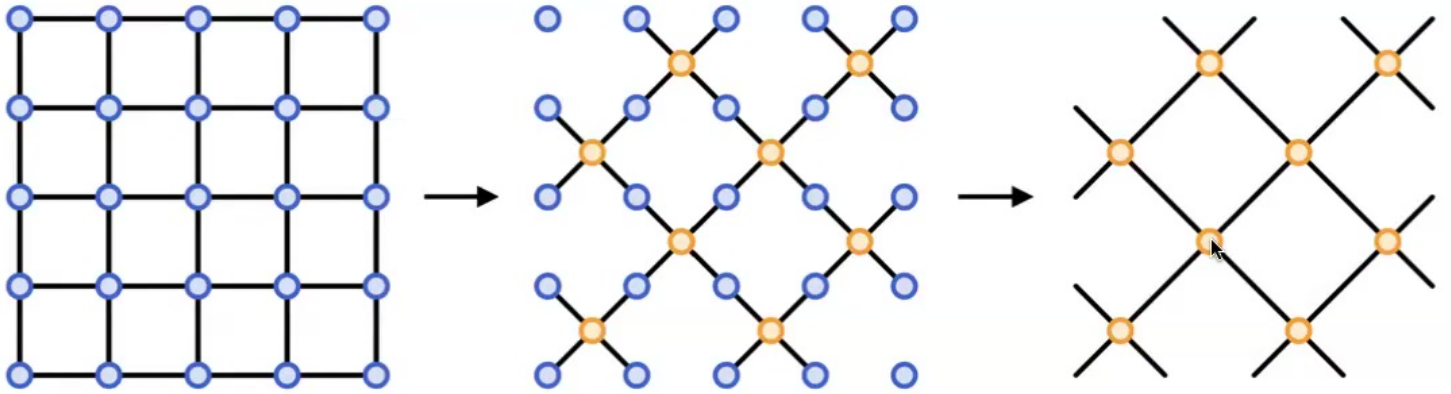
Extract effective model

Block partition $= \sum_I E_I(s) \longrightarrow = \sum_I \tilde{E}_I(s, s')$

Feature learning

Real Space Renormalization Group

- Summary



$$E(s) = -J \sum_{\langle ij \rangle} s_i s_j \xrightarrow{\text{Effective approx.}} \tilde{E}(s') = -J' \sum_{\langle ij \rangle} s'_i s'_j$$

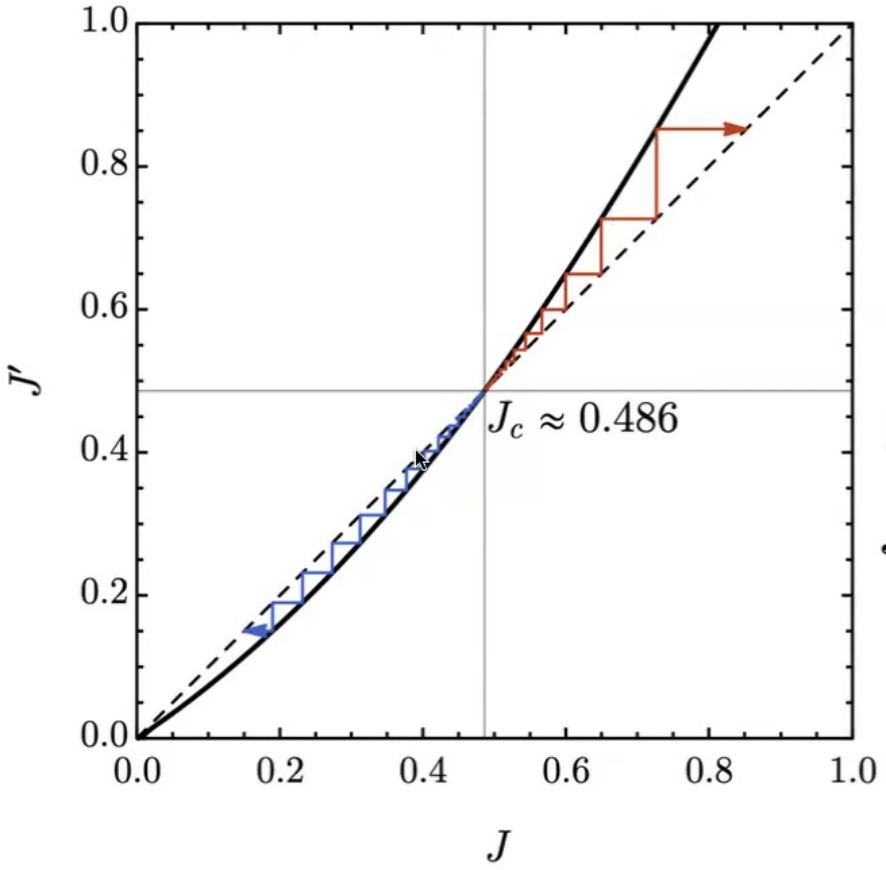
RG equation: $J' = \frac{1}{2} \ln \cosh(2w(J))$

Part of the equation is machine learned, not derived by human

- Lattice constant enlarged by $\sqrt{2}$
- Coupling flows as $J \to J'$

RG Fixed Points

- RG Flow: model parameter runs with RG iterations



- RG fixed points:

$J = 0$ (Disorder)

$J \approx 0.486$ (Critical)

$J = \infty$ (Ordered)

2D Ising exact solution

$$J_c = \frac{1}{2}\ln(1+\sqrt{2}) \approx 0.4407$$

Ising Criticality

- Scaling Law and Critical Exponents
  - Linearize the RG equation around the critical point

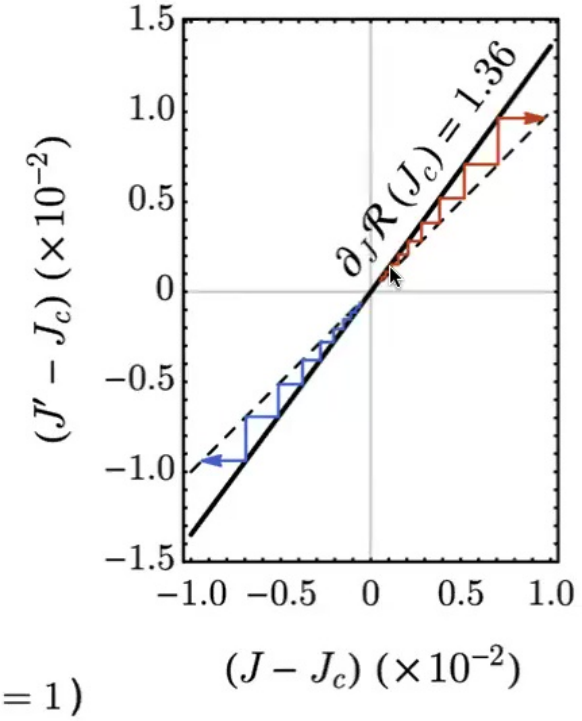
$$J' = \mathcal{R}(J) \xrightarrow{J', J \to J_c} (J' - J_c) = \partial_J \mathcal{R}(J_c)(J - J_c) + \cdots$$
$$= \tfrac{1}{2} \ln \cosh(2w(J))$$

  - After n steps of iteration

$$(J - J_c) = (\partial_J \mathcal{R}(J_c))^n (J_0 - J_c)$$
$$\xi = \xi_0 / (\sqrt{2})^n \quad \text{(Correlation length)}$$

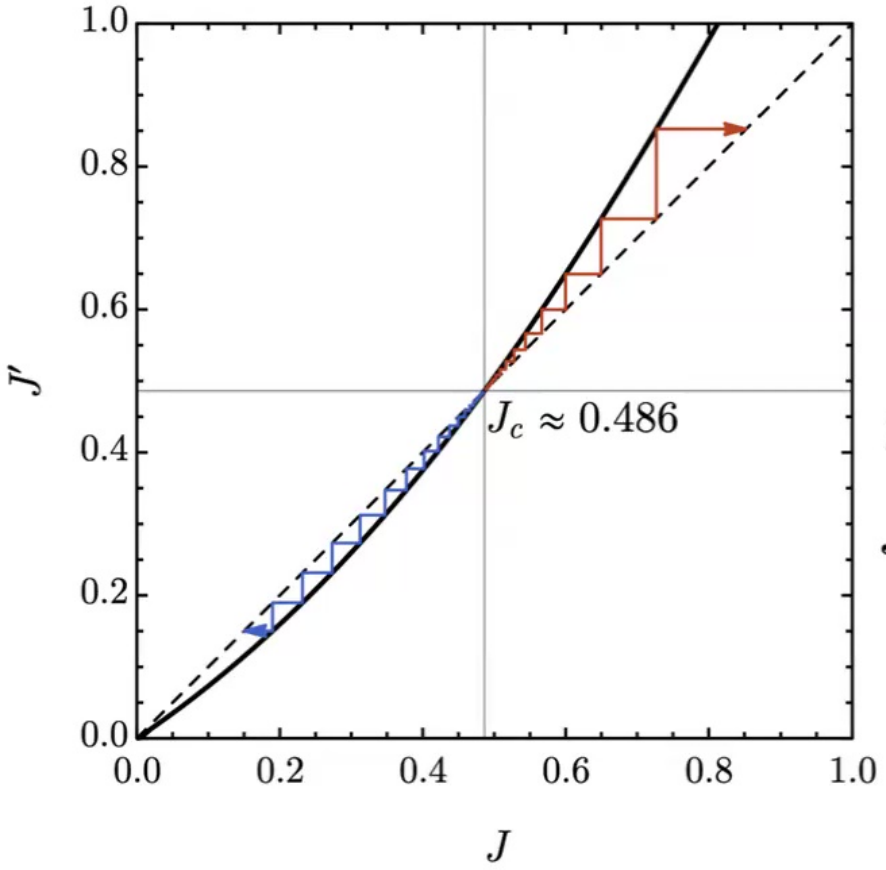
  - Cancel *n* from the equation
    Scaling law $\xi \propto (J - J_c)^{-\nu}$

Exponent $\nu = \dfrac{\ln \sqrt{2}}{\ln \partial_J \mathcal{R}(J_c)} \approx 1.1$ (Exact $\nu = 1$)



RG Fixed Points

- RG Flow: model parameter runs with RG iterations



- RG fixed points:

$J = 0$ (Disorder)

$J \approx 0.486$ (Critical)

$J = \infty$ (Ordered)

2D Ising exact solution

$$J_c = \frac{1}{2}\ln(1+\sqrt{2}) \approx 0.4407$$

Ising Criticality

- Scaling Law and Critical Exponents
  - Linearize the RG equation around the critical point

$$J' = \mathcal{R}(J) \xrightarrow{J', J \to J_c} (J' - J_c) = \partial_J \mathcal{R}(J_c)(J - J_c) + \cdots$$
$$= \tfrac{1}{2} \ln \cosh(2w(J))$$

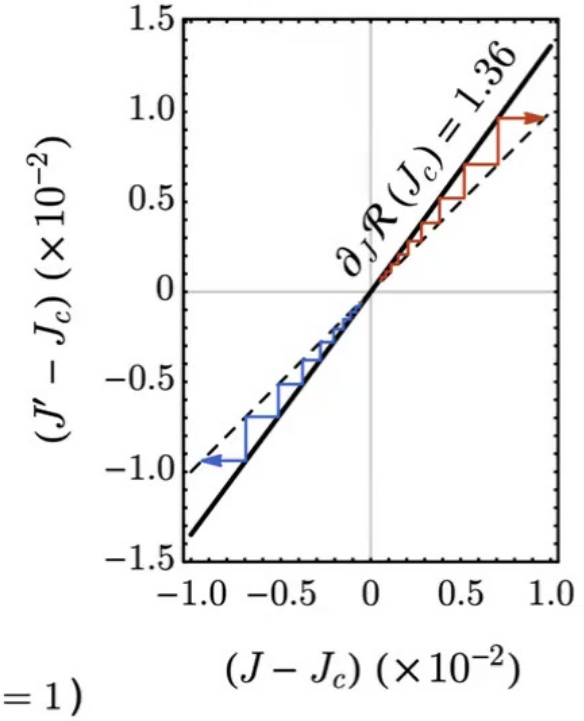
  - After n steps of iteration

$$(J - J_c) = (\partial_J \mathcal{R}(J_c))^n (J_0 - J_c)$$
$$\xi = \xi_0 / (\sqrt{2})^n \quad \text{(Correlation length)}$$

  - Cancel *n* from the equation

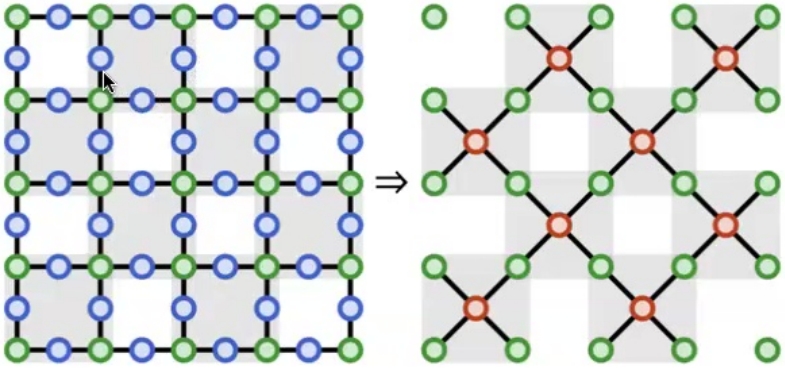
Scaling law $\xi \propto (J - J_c)^{-\nu}$

Exponent $\nu = \dfrac{\ln \sqrt{2}}{\ln \partial_J \mathcal{R}(J_c)} \approx 1.1$ (Exact $\nu = 1$)

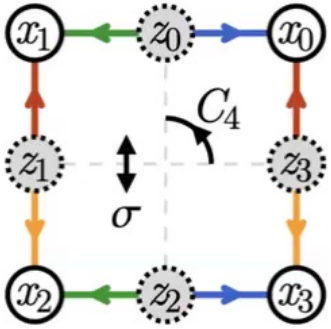


Teacher-Student Learning Paradigm

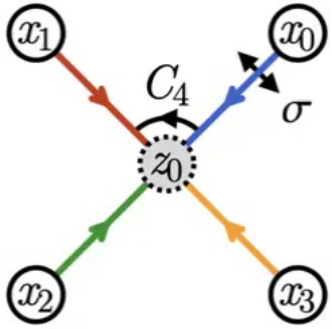
- It is more convenient to define our model on the Lieb lattice



(Teacher)



$\Rightarrow$



(Student)

Then RG iteration can be viewed as a fine-grained teacher RBM teaching a coarse-grained student RBM

Equivariant RBM

- Take $\epsilon_J(s_i, s_j) = -J_{ab} s_i^a s_j^b$ on representative bond:
  - A_1 (1-dim): 1 parameter

$$J = \begin{bmatrix} J_0 \end{bmatrix}$$

  - $A_1 \oplus E$ (3-dim): 5 parameters

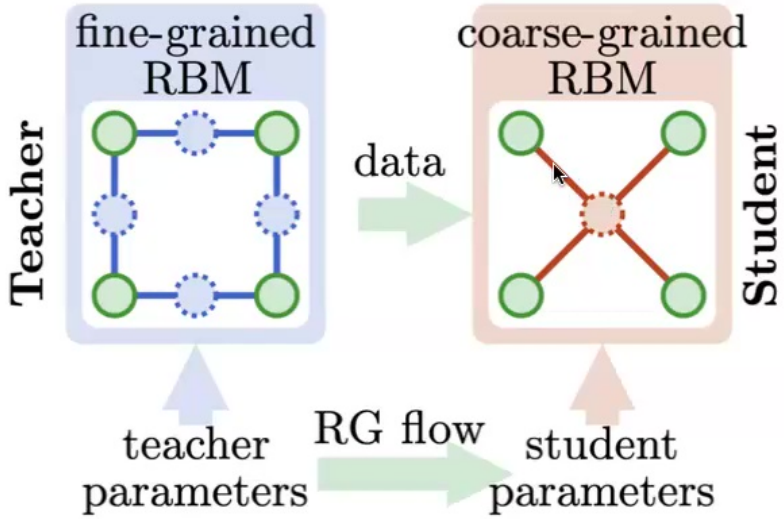
$$J = \left[\begin{array}{c|cc} J_0 & J_1 & 0 \\ \hline J_2 & J_3 & 0 \\ 0 & 0 & J_4 \end{array}\right]$$

  - $2A_1 \oplus E$ (4-dim): 10 parameters

$$J = \left[\begin{array}{c|c|cc} J_0 & J_1 & J_2 & 0 \\ \hline J_3 & J_4 & J_5 & 0 \\ \hline J_6 & J_7 & J_8 & 0 \\ 0 & 0 & 0 & J_9 \end{array}\right]$$

Teacher-Student Learning Paradigm

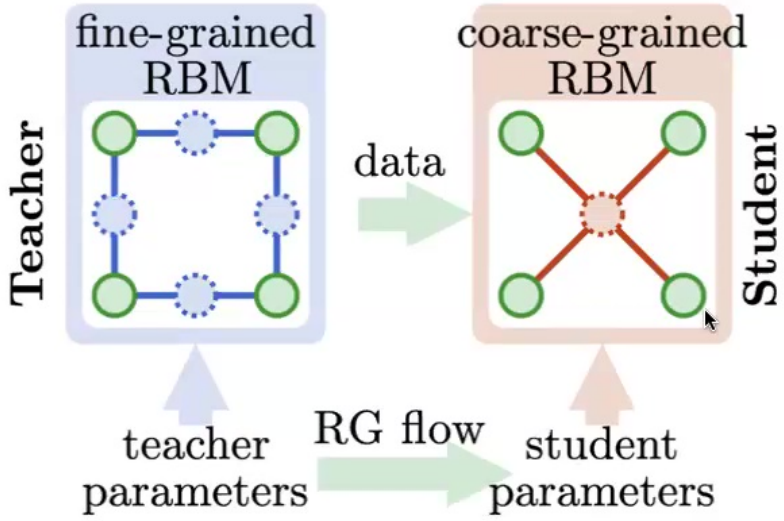
- Machine Learning RG (v0)



- Assign the initial parameter to the teacher RBM
- Teacher RBM generates data to train student RBM
- Replace the teacher with the trained student (copy student parameters to teacher) — parameters will flow under RG.

Teacher-Student Learning Paradigm

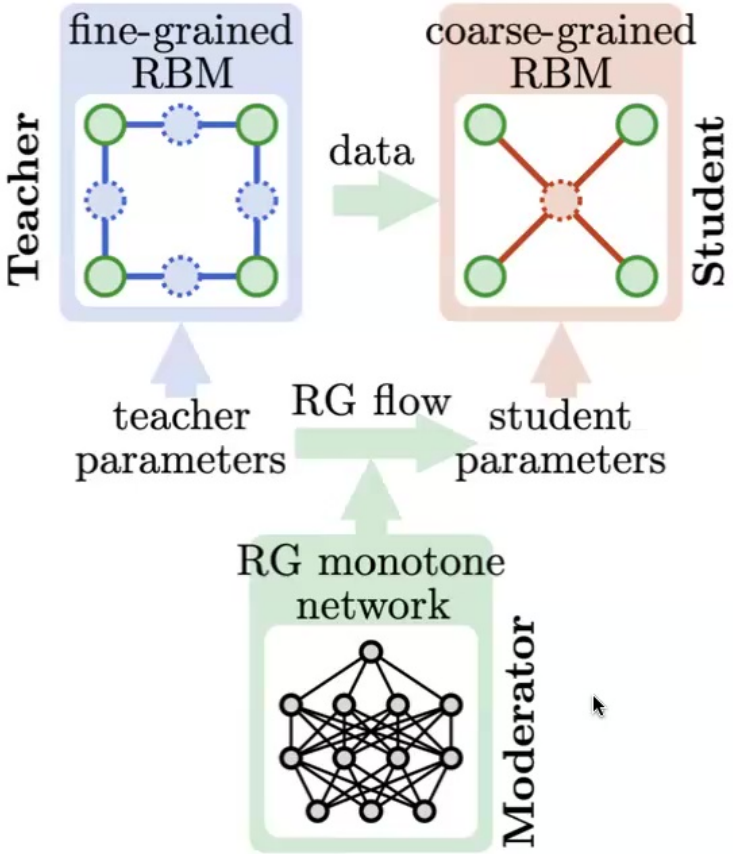
- Machine Learning RG (v0)



- Advantage: establish the optimal RG flow by unsupervised learning, automatically identify stable phases
- Challenge: almost always miss the critical point due to the stochastic nature of training RBM

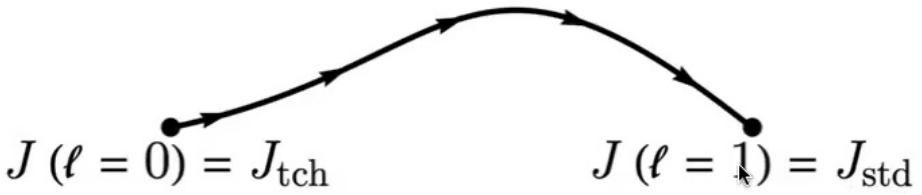
Introducing Moderator

- Introduce a third machine — the moderator — to learn the learning process between teacher and student



Introducing Moderator

- The moderator monitors the teach-student learning, learning to predict student parameters from teacher parameters
  - It models the parameter flow as a differential equation

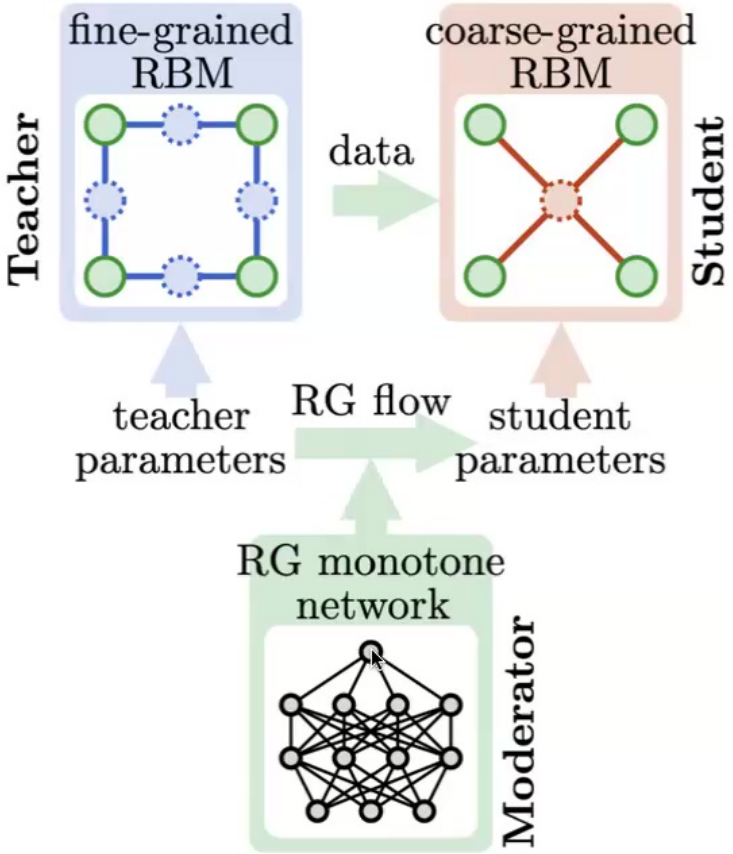


$$\frac{\mathrm{d}J}{\mathrm{d}\ell} = -\nabla_J C_\theta(J) \qquad \text{(RG equation)}$$

  - More specifically, the RG equation describes a gradient flow, induced by the RG monotone function $C_\theta(J) \in \mathbb{R}$ — modeled by a neural network with network parameters θ
  - The model can be trained by neural-ODE techniques

Introducing Moderator

- Introduce a third machine — the moderator — to learn the learning process between teacher and student



Training the Moderator

- Models are trained jointly
  - Start with J_{tch}
  - Call moderator to predict J_{std} by solving RG eq.

$$\frac{\mathrm{d}}{\mathrm{d}\ell} J = -\nabla_J C_\theta(J)$$

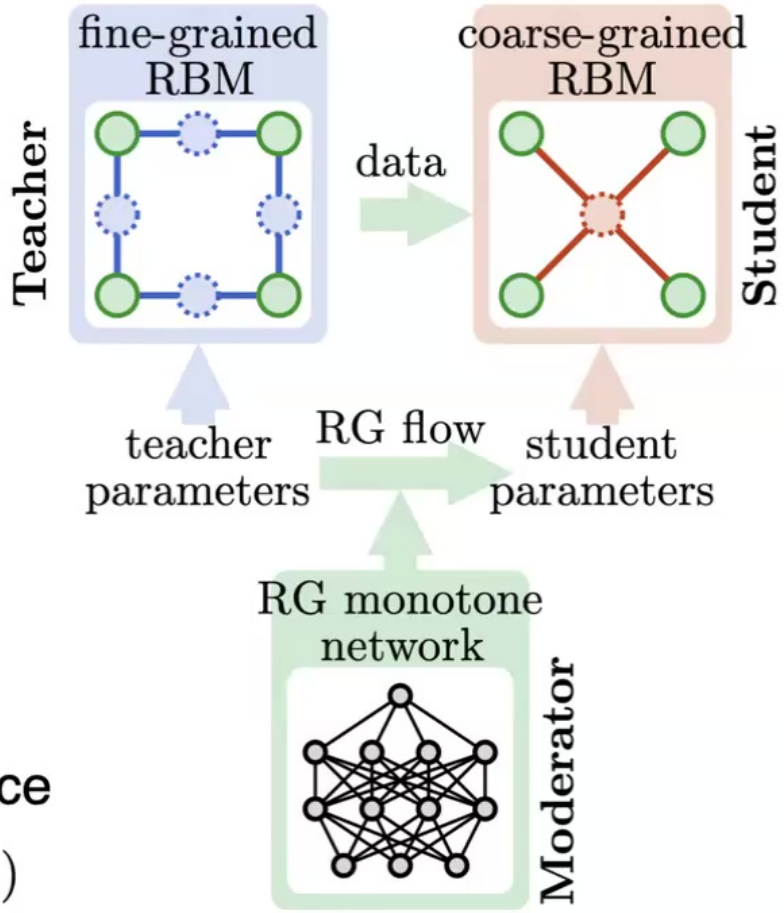
  - Mount $J_{\text{tch}}, J_{\text{std}}$ to the RBM energy models

$$p_{\text{tch}}(x) \propto \mathrm{e}^{-E_{J_{\text{tch}}}(x)}$$

$$p_{\text{std}}(x) \propto \mathrm{e}^{-E_{J_{\text{std}}}(x)}$$

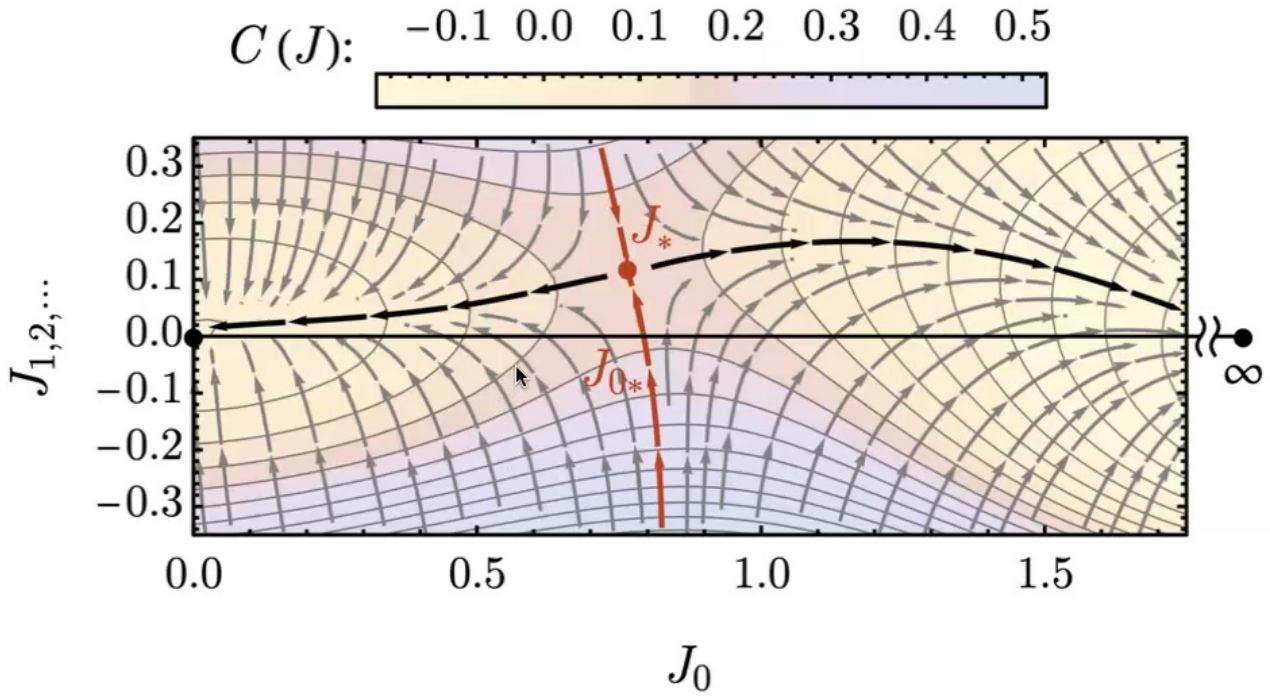
  - Minimize the KL divergence

$$\mathcal{L} = D_{\text{KL}}(p_{\text{tch}}(x) \| p_{\text{std}}(x))$$



Training the Moderator

- All the trainable parameters are in the moderator — after training, the main result is an optimal RG monotone $C(J)$
  - Pick $A_1 \oplus E$ representation, the result looks like this



Sampling in the Parameter Space

- The model should be trained for every possible J_{tch} — how to explore such a large parameter space more efficiently?
  - Some parameter regimes are more important — for example, near RG fixed points (especially critical points)

$$\nabla_J C(J) \simeq 0$$

  - Using importance sampling to guide the model to learn harder near RG fixed points

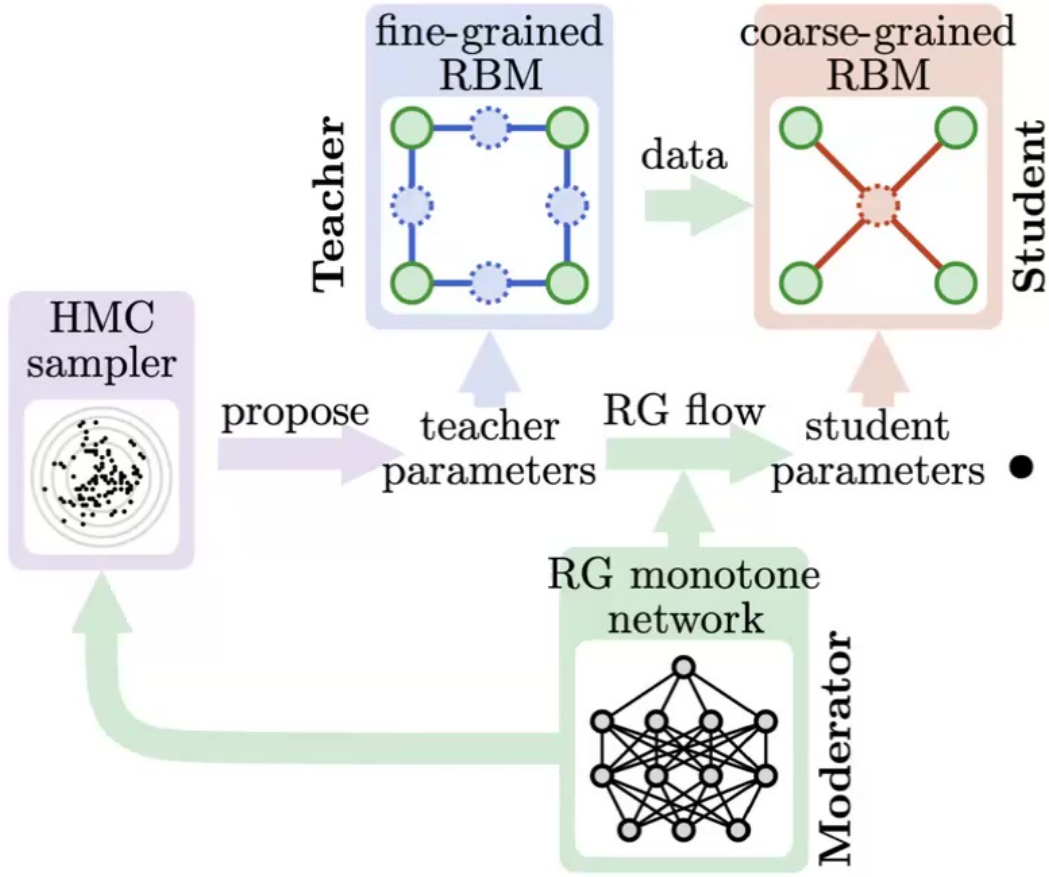
$$p(J) \propto \mathrm{e}^{-\beta \|\nabla_J C(J)\|^2}$$

    β is a hyper-parameter

  - Since J is continuous, a natural choice is the Hamiltonian Monte Carlo (HMC) approach

Algorithm Overview

- Machine Learning RG (v1)

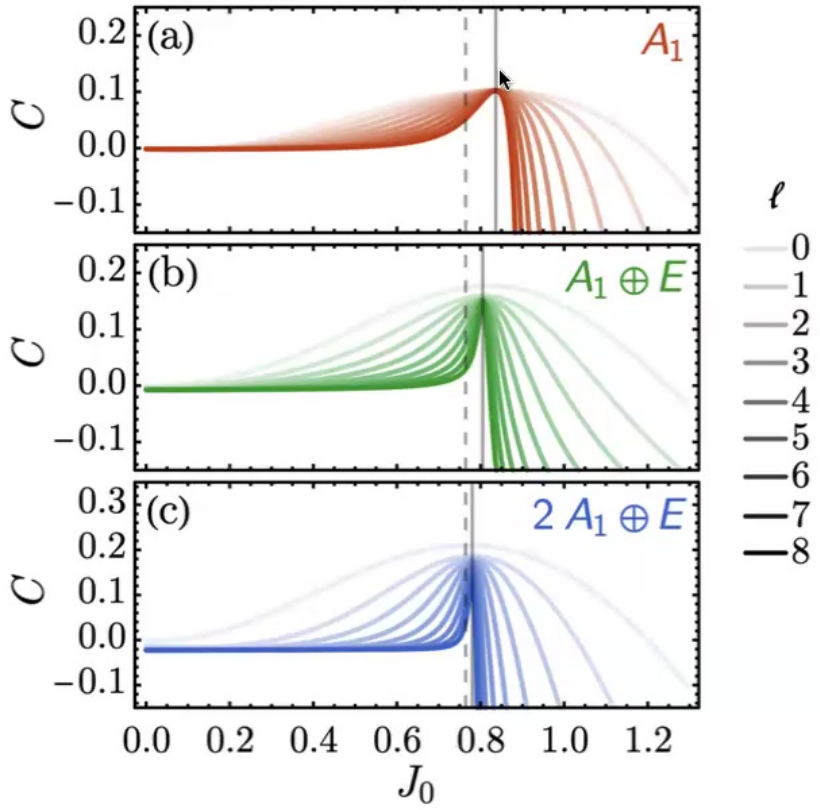


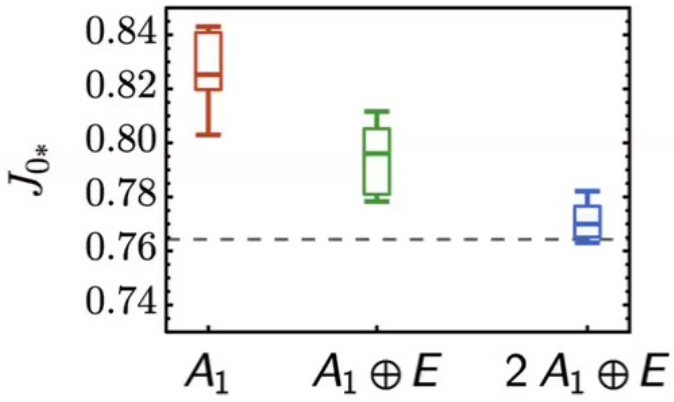
- Input:
  - Symmetry group $G = C_{4v} \times \mathbb{Z}_2$
  - Choice of on-site rep.: $\mathsf{A}_1, \mathsf{A}_1 \oplus \mathsf{E}, \cdots$
- Output:
  - RG monotone $C(J)$ and RG equation

$$\frac{\mathrm{d}}{\mathrm{d}\ell} J = -\nabla_J C(J)$$

Applications of MLRG

- Task 1: Automatic identification of critical points

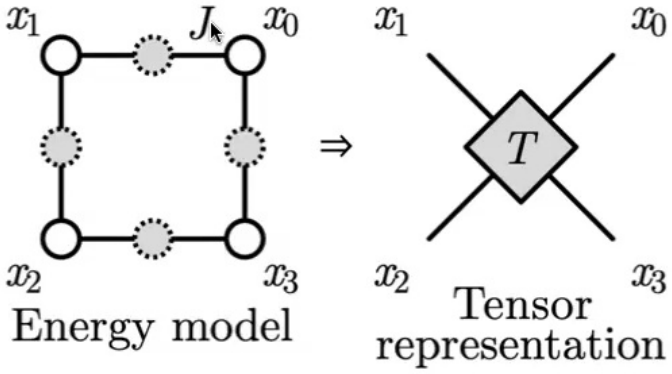




$$J_{0*} = \underset{J_0}{\operatorname{argmax}}\, C(J(J_0, \ell))$$

Applications of MLRG

- Task 2: Unsupervised classification of SSB phases
  - Quantization: translate energy models to tensor networks



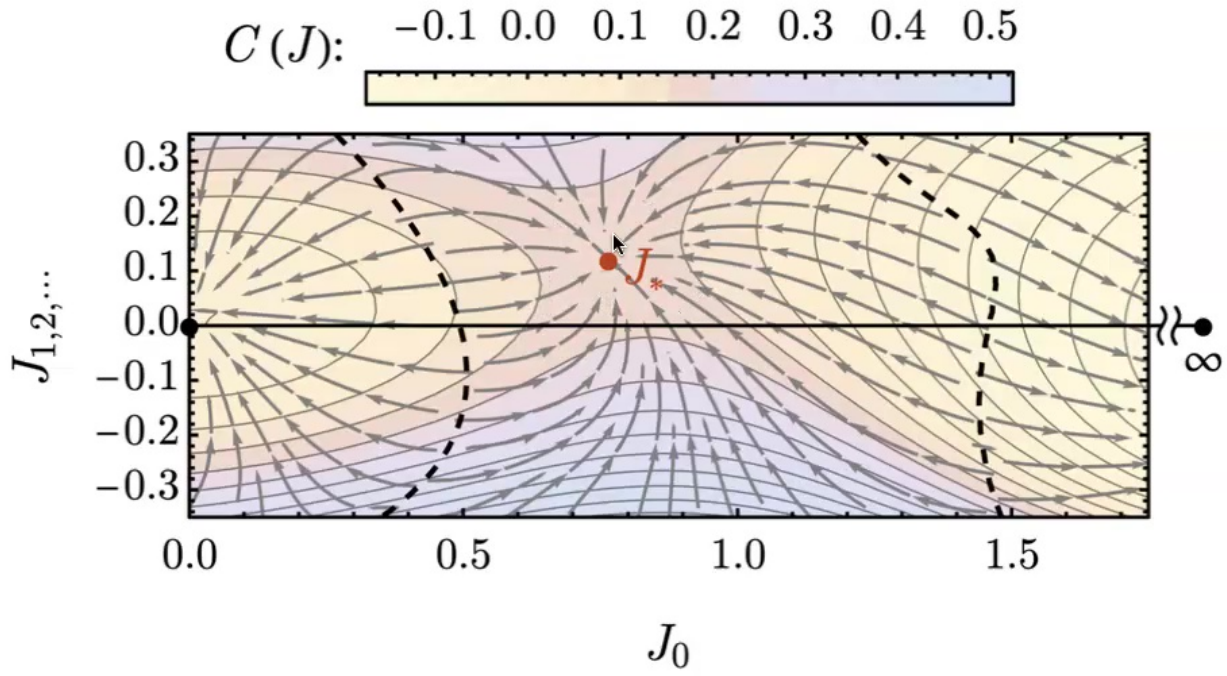
  - Estimate ground state degeneracy

$$Z = \left(\quad T \quad \right)^2 \Big/ \quad T \quad T$$

Application of MLRG

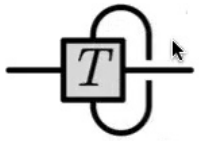
- Task 3: Extract critical exponents
  - Local RG fixed points (using Newton's method to solve $\nabla_J C = 0$)

$$J \to J - (\nabla_J^2 C)^{-1} \nabla_J C$$

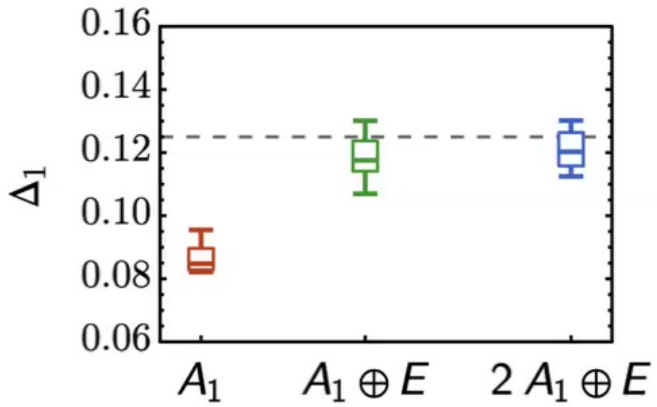


Application of MLRG

- Task 3: Extract critical exponents
  - Construct transfer matrix from fixed-point tensor

$M =$ 

  - Obtain operator scaling dimensions from eigenvalues of the transfer matrix $\operatorname{Tr} M^n \propto \sum_k \mathrm{e}^{-2\pi n(\Delta_k - c/12)}$



Summary

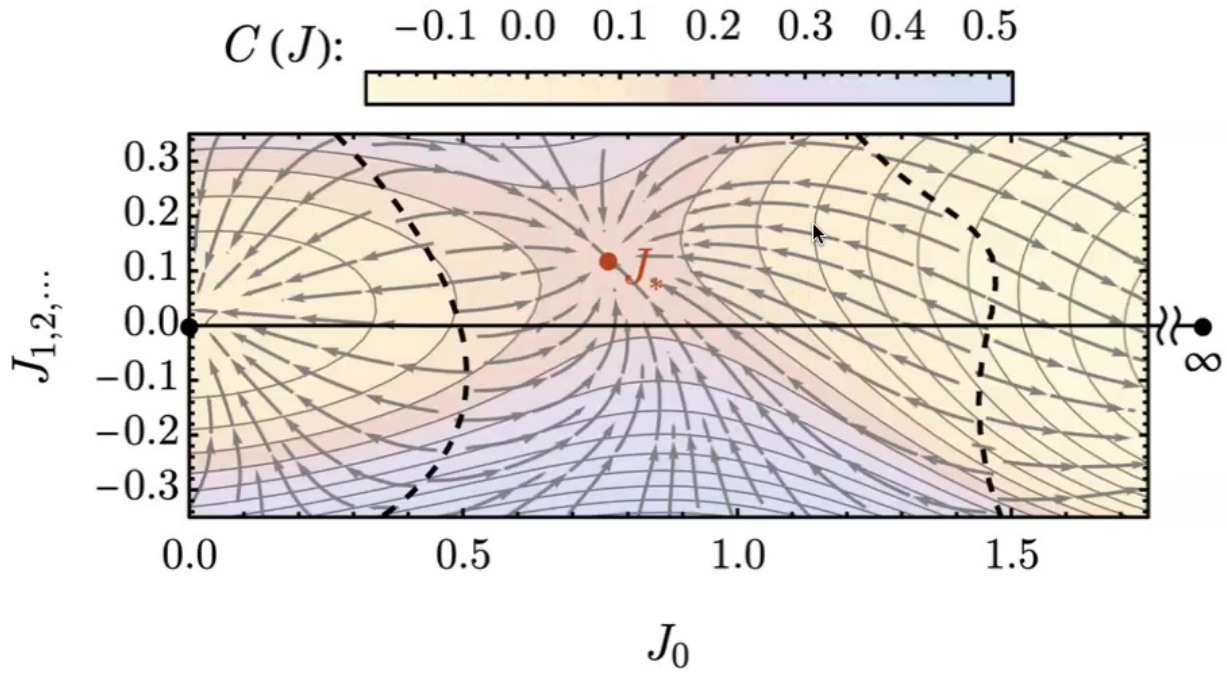
- MLRG method can automatically discover RG monotones, identify critical points, and estimate critical exponents, all starting from the symmetry and dimension of the physical system.
- Features:
  - Efficiency in Small Cluster Sampling
  - Exploration of the Full Parameter Space
  - Discovery and Analysis of Critical Points
  - Controlled Convergence of the Algorithm
  - Interpretability

github.com/
EverettYou/
MLRG

Application of MLRG

- Task 3: Extract critical exponents
  - Local RG fixed points (using Newton's method to solve $\nabla_J C = 0$)

$$J \to J - (\nabla_J^2 C)^{-1} \nabla_J C$$



Summary

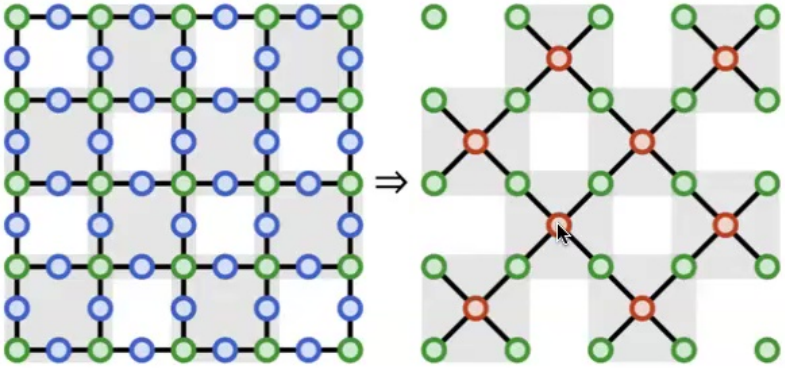
- MLRG method can automatically discover RG monotones, identify critical points, and estimate critical exponents, all starting from the symmetry and dimension of the physical system.
- Features:
  - Efficiency in Small Cluster Sampling
  - Exploration of the Full Parameter Space
  - Discovery and Analysis of Critical Points
  - Controlled Convergence of the Algorithm
  - Interpretability

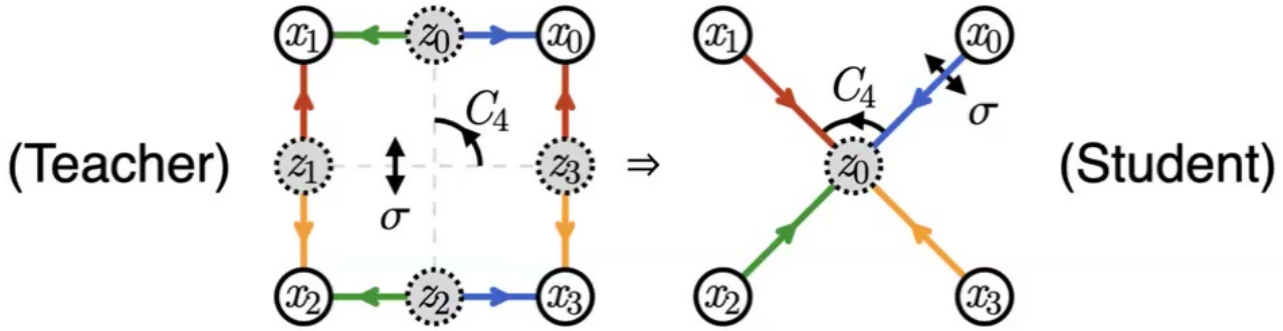
github.com/
EverettYou/
MLRG

Thanks!

Teacher-Student Learning Paradigm

- It is more convenient to define our model on the Lieb lattice





Then RG iteration can be viewed as a fine-grained teacher RBM teaching a coarse-grained student RBM

Training the Moderator

- Models are trained jointly
  - Start with J_{tch}
  - Call moderator to predict J_{std} by solving RG eq.

$$\frac{\mathrm{d}}{\mathrm{d}\ell} J = -\nabla_J C_\theta(J)$$

  - Mount $J_{\mathrm{tch}}, J_{\mathrm{std}}$ to the RBM energy models

$$p_{\mathrm{tch}}(x) \propto \mathrm{e}^{-E_{J_{\mathrm{tch}}}(x)}$$

$$p_{\mathrm{std}}(x) \propto \mathrm{e}^{-E_{J_{\mathrm{std}}}(x)}$$

  - Minimize the KL divergence

$$\mathcal{L} = D_{\mathrm{KL}}(p_{\mathrm{tch}}(x) \| p_{\mathrm{std}}(x))$$