Introduction to Machine Learning Methods in Condensed Matter Physics

LECTURE 7, FALL 2021

Yi Zhang (张亿)

International Center for Quantum Materials, School of Physics
Peking University, Beijing, 100871, China

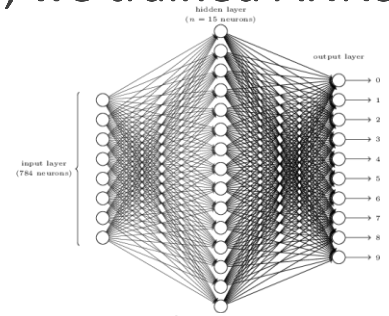
Email: frankzhangyi@pku.edu.cn

Generative models and graphic models

- We want more naturally looking hand-written digits!
- Previously, we trained ANNs to recognize hand-written digits:

Input

$\boldsymbol{x}$

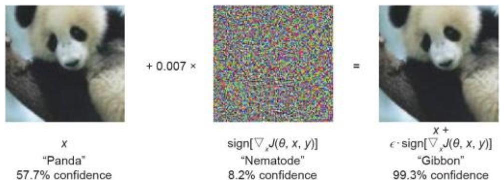


Judgement output

$f(\boldsymbol{x})$

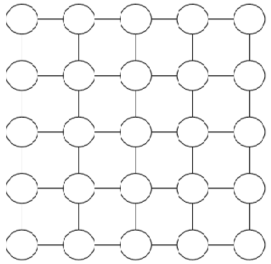
- **Generative model** versus **discriminative model**:

Input $\boldsymbol{x}$ → More inputs following the same rules: $\boldsymbol{x}', \boldsymbol{x}'', \boldsymbol{x}''', \cdots$



We can sample $\boldsymbol{x}$ with respect to $f(\boldsymbol{x})$, but it is inefficient and expansive.

- **Graphic models**: probability distribution with statistical mechanics:



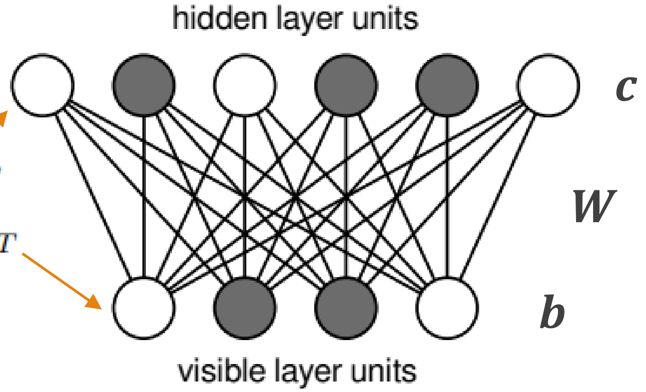
$$\mu(v) = \frac{1}{Z} \exp\left\{ \sum_i \theta_i v_i + \sum_{(i,j)\in E} \theta_{ij} v_i v_j \right\}$$

Restricted Boltzmann Machine

- Equivalent to a fully-connected feed-forward ANN with two layers:
- A binary graphic model with a visible layer and hidden layer:

"restrict": no intra-layer connections

$$\begin{cases} \boldsymbol{H} = (H_1, ..., H_J)^T \\ \boldsymbol{X} = (X_1, ..., X_I)^T \end{cases}$$



- The probability of a configuration: Boltzmann distribution

$\boldsymbol{W}$, $\boldsymbol{b}$, and $\boldsymbol{c}$ as model parameters

$$\begin{cases} P(\boldsymbol{X}, \boldsymbol{H}) = \frac{1}{Z} \exp(-E(\boldsymbol{X}, \boldsymbol{H})) \\ E(\boldsymbol{X}, \boldsymbol{H}) = -\boldsymbol{X}^T \boldsymbol{b} - \boldsymbol{c}^T \boldsymbol{H} - \boldsymbol{X}^T \boldsymbol{W} \boldsymbol{H} \end{cases}$$

- Partition function:

$$Z = \sum_{\boldsymbol{X}, \boldsymbol{H}} \exp(-E(\boldsymbol{X}, \boldsymbol{H}))$$

- Free energy w.r.t visible layer:

$$F(\boldsymbol{X}) = -\ln\left(\sum_{\boldsymbol{h}} \exp(-E(\boldsymbol{X}, \boldsymbol{h}))\right)$$

$$P(\boldsymbol{X}) = \frac{1}{Z} \exp(-F(\boldsymbol{X}))$$

Restricted Boltzmann Machine

- Some probabilities and conditional probabilities:

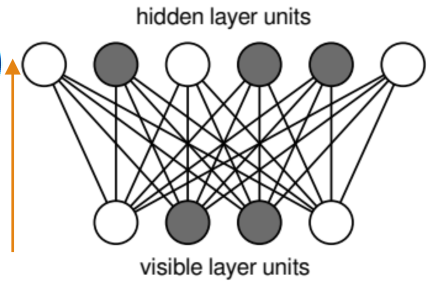
$$P(\boldsymbol{X},\boldsymbol{H}) = \frac{1}{Z}\exp\left(\boldsymbol{X}^T\boldsymbol{b}+\sum_j\left(c_j+\boldsymbol{X}^T\boldsymbol{w}_j\right)H_j\right) = \frac{1}{Z}\exp\left(\boldsymbol{X}^T\boldsymbol{b}\right)\prod_j\exp\left(\left(c_j+\boldsymbol{X}^T\boldsymbol{w}_j\right)H_j\right)$$

$$P(\boldsymbol{X}) = \sum_{\boldsymbol{h}}P(\boldsymbol{X},\boldsymbol{h}) = \frac{1}{Z}\exp\left(\boldsymbol{X}^T\boldsymbol{b}\right)\prod_j\sum_{h_j}\exp\left(\left(c_j+\boldsymbol{X}^T\boldsymbol{w}_j\right)h_j\right) = \frac{1}{Z}\exp\left(\boldsymbol{X}^T\boldsymbol{b}\right)\prod_j\left(1+\exp\left(c_j+\boldsymbol{X}^T\boldsymbol{w}_j\right)\right)$$

easy to evaluate for given $\boldsymbol{X}$

- Given the visible variables, the hidden variables are conditionally independent (*and vice versa*):

$$P(\boldsymbol{H}|\boldsymbol{X}) = \frac{P(\boldsymbol{X},\boldsymbol{H})}{P(\boldsymbol{X})} = \prod_j\frac{\exp\left(\left(c_j+\boldsymbol{X}^T\boldsymbol{w}_j\right)H_j\right)}{1+\exp\left(c_j+\boldsymbol{X}^T\boldsymbol{w}_j\right)} = \prod_j P(H_j|\boldsymbol{X})$$



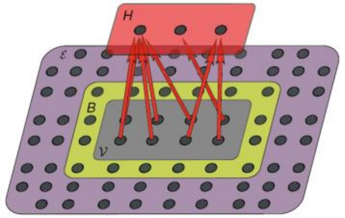
- Similarity and connections to ANN (of weights $\boldsymbol{W}$ and biases $\boldsymbol{c}$):

$$P(h_j=1|\boldsymbol{X}) = \frac{\exp\left(c_j+\boldsymbol{X}^T\boldsymbol{w}_j\right)}{1+\exp\left(c_j+\boldsymbol{X}^T\boldsymbol{w}_j\right)} = \sigma\left(c_j+\boldsymbol{X}^T\boldsymbol{w}_j\right)$$

https://www.ini.rub.de/PEOPLE/wiskott/Teaching/Material/RestrictedBoltzmannMachines-LectureNotesPublic.pdf

- Size of $\boldsymbol{H}$ controls and describes the effective degrees of freedom in $\boldsymbol{X}$

e.g. previous example of RG with mutual information:



Training the Restricted Boltzmann Machine

- Training the model by adapting the parameters $\boldsymbol{b}$, $\boldsymbol{c}$, and $\boldsymbol{W}$ with gradient descent:

- Example: maximize the likelihood of the given data:

η: learning rate

(given a distribution → sample for $\boldsymbol{X}$)
Other options are discussed later.

$$\ln(P(\boldsymbol{X})) = -F(\boldsymbol{X}) - \ln(Z)$$

$$\Delta\theta = \eta \frac{\partial \ln(P(\boldsymbol{X}))}{\partial\theta}$$

Contrastive divergence (CD):

$$\frac{\partial \ln(P(\boldsymbol{X}))}{\partial\theta} = -\frac{\partial F(\boldsymbol{X})}{\partial\theta} - \frac{1}{Z}\frac{\partial Z}{\partial\theta} = -\frac{\partial F(\boldsymbol{X})}{\partial\theta} + \sum_{\boldsymbol{x}'} P(\boldsymbol{x}') \cdot \frac{\partial F(\boldsymbol{x}')}{\partial\theta} = -\left\langle \frac{\partial F(\tilde{\boldsymbol{x}})}{\partial\theta} \right\rangle_{\tilde{\boldsymbol{x}}} + \left\langle \frac{\partial F(\boldsymbol{x}')}{\partial\theta} \right\rangle_{\boldsymbol{x}'}$$

Average over the training dataset

Sample over all configurations with Markov Chain Monte Carlo

$$\frac{\partial F(\boldsymbol{X})}{\partial b_i} = -X_i$$

$$\frac{\partial F(\boldsymbol{X})}{\partial c_j} = -\sigma(c_j + \boldsymbol{X}^T \boldsymbol{w}_j) = -y_j(\boldsymbol{X})$$

$$\frac{\partial F(\boldsymbol{X})}{\partial w_{ij}} = -X_i\, y_j(\boldsymbol{X})$$

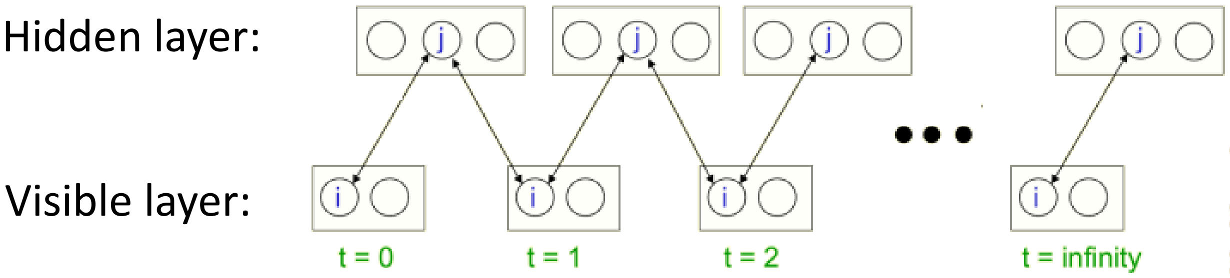
$$\begin{aligned}\frac{\partial F(\boldsymbol{X})}{\partial\theta} &= \frac{\partial}{\partial\theta}\left(-\ln\left(\sum_{\boldsymbol{h}} \exp\left(-E(\boldsymbol{X},\boldsymbol{h})\right)\right)\right) \\ &= \left(\sum_{\boldsymbol{h}'} P(\boldsymbol{X},\boldsymbol{h}')\right)^{-1} \left(\sum_{\boldsymbol{h}} P(\boldsymbol{X},\boldsymbol{h}) \cdot \frac{\partial E(\boldsymbol{X},\boldsymbol{h})}{\partial\theta}\right) \\ &= \sum_{\boldsymbol{h}} P(\boldsymbol{h}|\boldsymbol{X}) \cdot \frac{\partial E(\boldsymbol{X},\boldsymbol{h})}{\partial\theta} = \sum_{\boldsymbol{h}} \left(\prod_j P(h_j|\boldsymbol{X})\right) \cdot \frac{\partial E(\boldsymbol{X},\boldsymbol{h})}{\partial\theta}\end{aligned}$$

$$E(\boldsymbol{X},\boldsymbol{H}) = -\boldsymbol{X}^T\boldsymbol{b} - \boldsymbol{c}^T\boldsymbol{H} - \boldsymbol{X}^T\boldsymbol{W}\boldsymbol{H}$$

- MCMC with $P(\boldsymbol{H}|\boldsymbol{X})$ to update $\boldsymbol{H}$ and $P(\boldsymbol{X}|\boldsymbol{H})$ to update $\boldsymbol{X}$ in turn throughout the system.

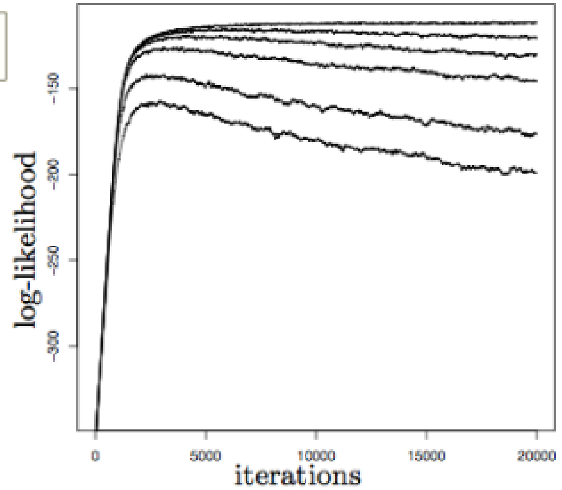
Training the Restricted Boltzmann Machine

- An advantage of restricted Boltzmann machine architecture – particularly easy Gibbs sampling



- k-step contrastive divergence
  - ★ **Input:** Graph G over v, h, training samples $S = \{v^{(1)}, \dots, v^{(n)}\}$
  - ★ **Output:** gradient $\{\Delta w_{ij}\}_{i\in[N], j\in[M]}, \{\Delta a_i\}_{i\in[N]}, \{\Delta b_j\}_{j\in[M]}$

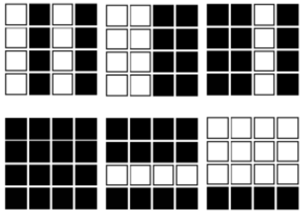
initialize $\Delta w_{ij}, \Delta a_i, \Delta b_j = 0$
Repeat
for all $v^{(\ell)} \in S$
$v(0) \leftarrow v^{(\ell)}$
for $t = 0, \dots, k-1$ **do**
  for $i = 1, \dots, N$ **do** sample $h(t)_i \sim \mu(h_i|v(t))$
  for $j = 1, \dots, M$ **do** sample $v(t+1)_j \sim \mu(v_j|h(t))$
for $i = 1, \dots, N, j = 1, \dots, M$ **do**
  $\Delta w_{ij} \leftarrow \Delta w_{ij} + \mathbb{E}_{\mu(h_i|v(0))}[h_i v(0)_j] - \mathbb{E}_{\mu(h_i|v(k))}[h_i v_j]$
  $\Delta a_i \leftarrow \Delta a_i + v(0)_j - v(k)_j$
  $\Delta b_j \leftarrow \Delta b_j + \mathbb{E}_{\mu(h_i|v(0))}[h_i] - \mathbb{E}_{\mu(h_i|v(k))}[h_i]$



Increasing k: convergence to maximum-likelihood solution

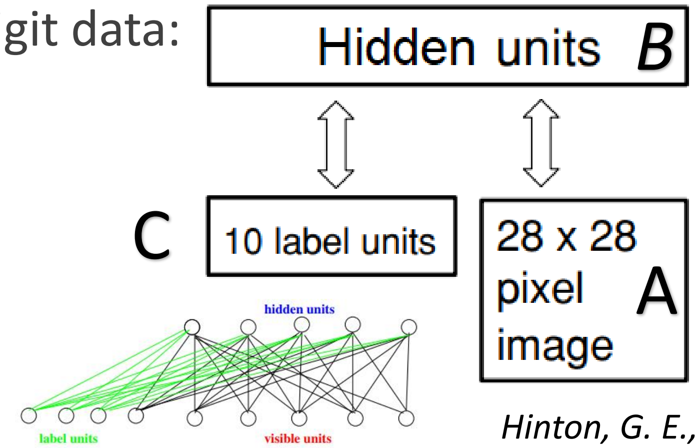
contrastive divergence with 16 hidden neurons and $k = 1,2,5,10,20,100$ on bars and stripes:

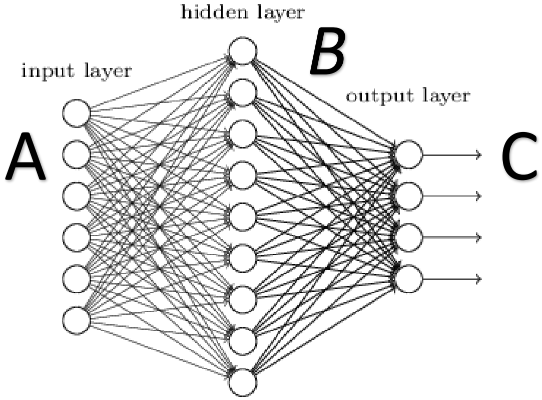
Application of the trained RBM as generative model for synthetic data also via sampling



Generating artificial hand-written digits with RBM

- Application of the RBM with additional digit class labels trained on 60,000 MNIST hand-written digit data: compared with feed-forward ANN:



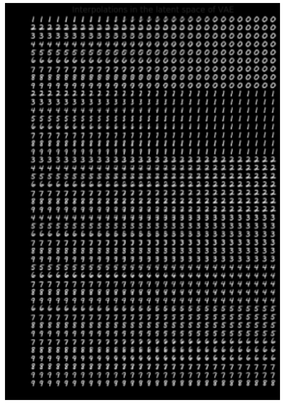


Hinton, G. E., Osindero, S. and Teh, Y., Neural Computation ***18****, 1527-1554 (2006).*

- Image recognition: input the visible units on the right, sampling the hidden units and the visible units on the left ($A \rightarrow B \,\&\, C$).

- Generating handwritten digits: input the visible layer on the left, sampling the hidden units with the visible units on the right ($C \rightarrow A \,\&\, B$).

- Interpolating different class labels:

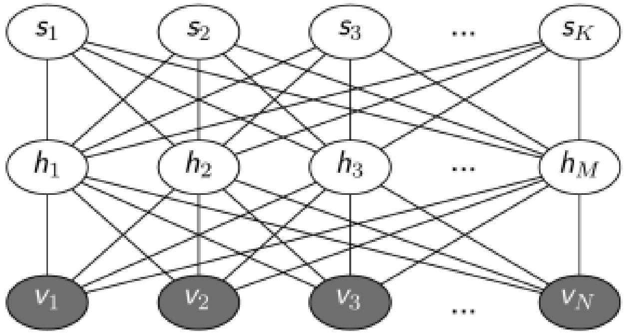


Deep Boltzmann Machine

- DBM consists of more than one layers of hidden neurons:

$$\mu(v, h, s) \quad = \quad \frac{1}{Z} \exp\left\{a^T v + b^T h + c^T s + v^T W^1 h + h^T W^2 s\right\}$$

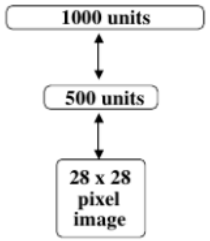
capable of learning more complex representations



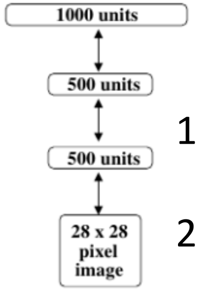
- Example: training on MINST data for hand-written digits:

Training Samples

2-layer BM



3-layer BM



1. Assume higher layers do not exist when training lower ones;
2. Use approximations, including variational methods; etc.

- However, training is also considerably more expensive: more sampling and averaging involved

Maximize: $$\mathcal{L}(W^1, W^2) \;=\; \log \sum \exp\left\{(v^{(\ell)})^T W^1 h + h^T W^2 s\right\} - \log Z$$

$$\Rightarrow \quad \frac{\partial \log \mu(v^{(\ell)})}{\partial W^1_{ij}} = \mathbb{E}_{\mu(h|v^{(\ell)})}[v_i^{(\ell)} h_j] - \mathbb{E}_{\mu(v,h)}[v_i h_j] \qquad \frac{\partial \log \mu(v^{(\ell)})}{\partial W^2_{ij}} = \mathbb{E}_{\mu(h,s|v^{(\ell)})}[h_i s_j] - \mathbb{E}_{\mu(h,s)}[h_i s_j]$$

Training the Restricted Boltzmann Machine revisited

- Train RBM to fit a *given distribution*, e.g. to minimize the KL divergence

- Example: classical fields coupled to quadratic fermions: the Falicov-Kimball model on 2D lattice (classical MC but with potentially nontrivial probability distribution)

$$\hat{H}_{\mathrm{FK}} = \sum_{i,j} \hat{c}_i^\dagger \mathcal{K}_{ij} \hat{c}_j + U \sum_{i=1}^{N} \left(\hat{n}_i - \frac{1}{2}\right)\left(x_i - \frac{1}{2}\right)$$

$$x_i \in \{0,1\} \qquad \mathcal{K}_{ij} = -t \qquad \beta = 1/T$$

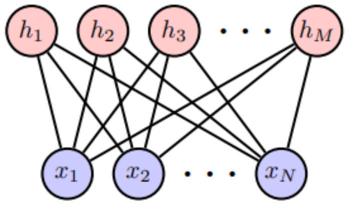
- Integrate out the fermions:

$$p_{\mathrm{FK}}(\mathbf{x}) = e^{-F_{\mathrm{FK}}(\mathbf{x})}/Z_{\mathrm{FK}}$$

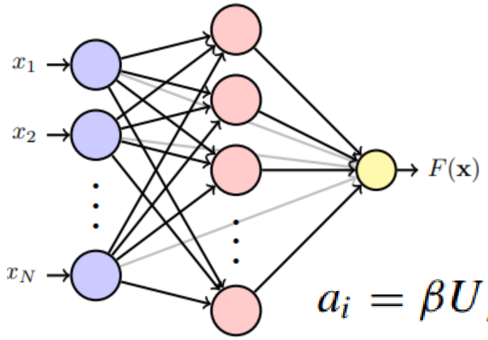
$$-F_{\mathrm{FK}}(\mathbf{x}) = \frac{\beta U}{2} \sum_{i=1}^{N} x_i + \ln\det(1 + e^{-\beta\mathcal{H}}) \qquad \mathcal{H}_{ij} = \mathcal{K}_{ij} + \delta_{ij} U (x_i - 1/2)$$

- To be compared and fit with RBM:

$$-F(\mathbf{x}) = \sum_{i=1}^{N} a_i x_i + \sum_{j=1}^{M} \ln(1 + e^{b_j + \sum_{i=1}^{N} x_i W_{ij}})$$



Train as a feed forward neural network via supervised machine learning for weights and biases a_i, b_j and W_{ij} :



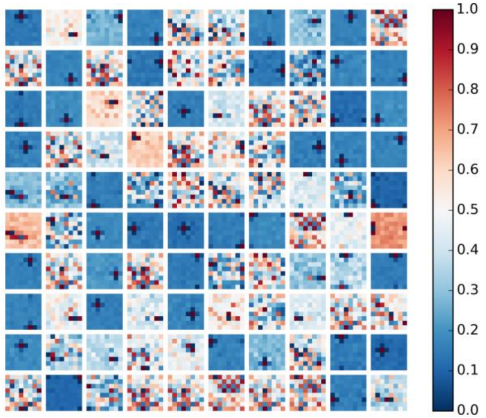
$$a_i = \beta U/2$$

*Li Huang and Lei Wang, Phys. Rev. B **95**, 035105 (2017).*

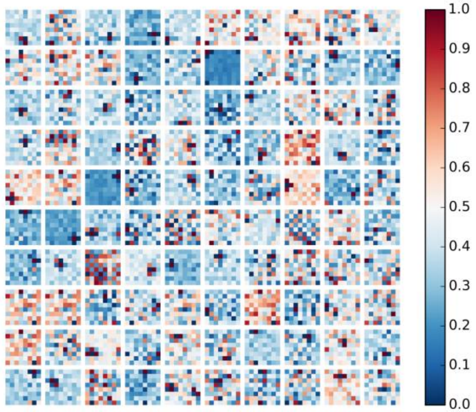
Restricted Boltzmann Machine for Monte Carlo updates

- The trained RBM successfully captures the probability distribution:

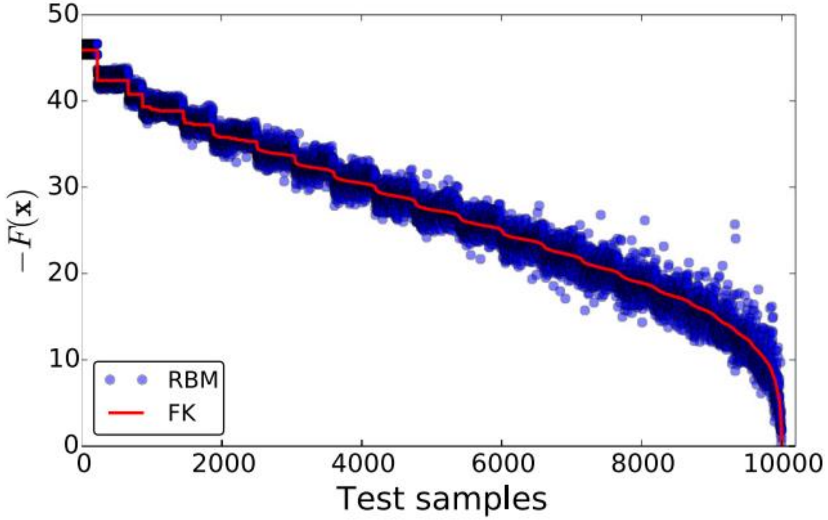
- Weights W_{ij} pick up the characteristic features of model:



Staggered DW pattern at $T/t = 0.15$



More visible pattern with enlarged correlation length at $T/t = 0.13$



100 hidden neurons, $\frac{U}{t} = 4, \frac{T}{t} = 0.15$ near critical point (difficult region with large fluctuations)

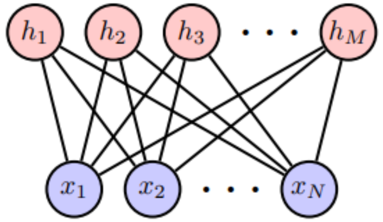
- Generate MC updates with RBM generative features:

Then, accept with probability to compensate imperfections: $$A(\mathbf{x} \to \mathbf{x}') = \min\left[1, \frac{p(\mathbf{x})}{p(\mathbf{x}')} \cdot \frac{p_{\mathrm{FK}}(\mathbf{x}')}{p_{\mathrm{FK}}(\mathbf{x})}\right]$$

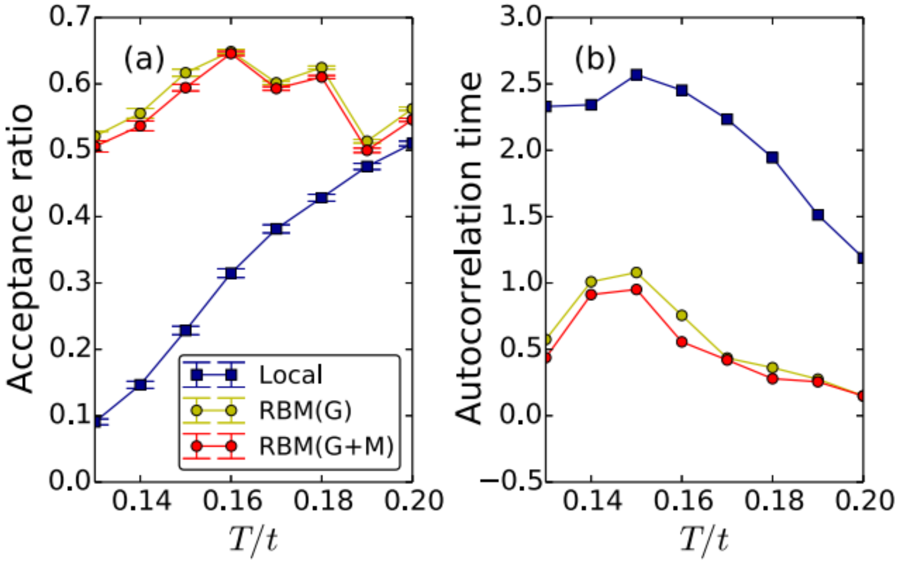
*Li Huang and Lei Wang, Phys. Rev. B **95**, 035105 (2017).*

Restricted Boltzmann Machine for Monte Carlo updates

- The hidden variable has a nonlocal effect on the physical (visible) variables.



- Drastically improved acceptance ratio and autocorrelation time:



*Li Huang and Lei Wang, Phys. Rev. B **95**, 035105 (2017).*