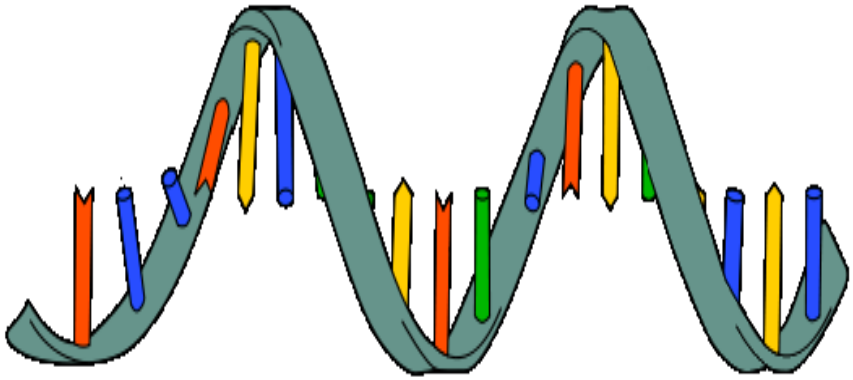
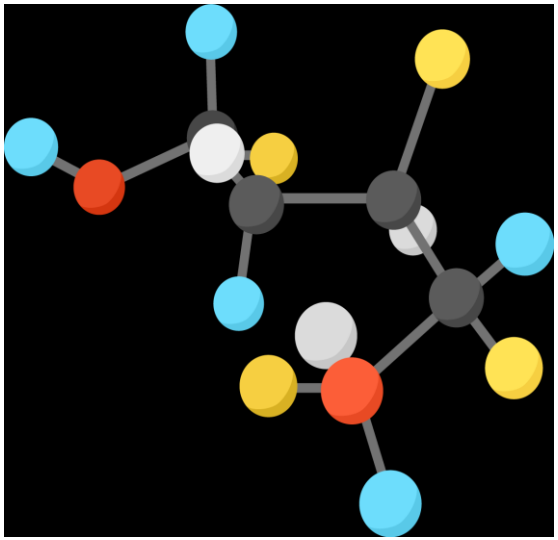
Discriminative Embedding of Molecular Structures for Property Prediction

Le Song

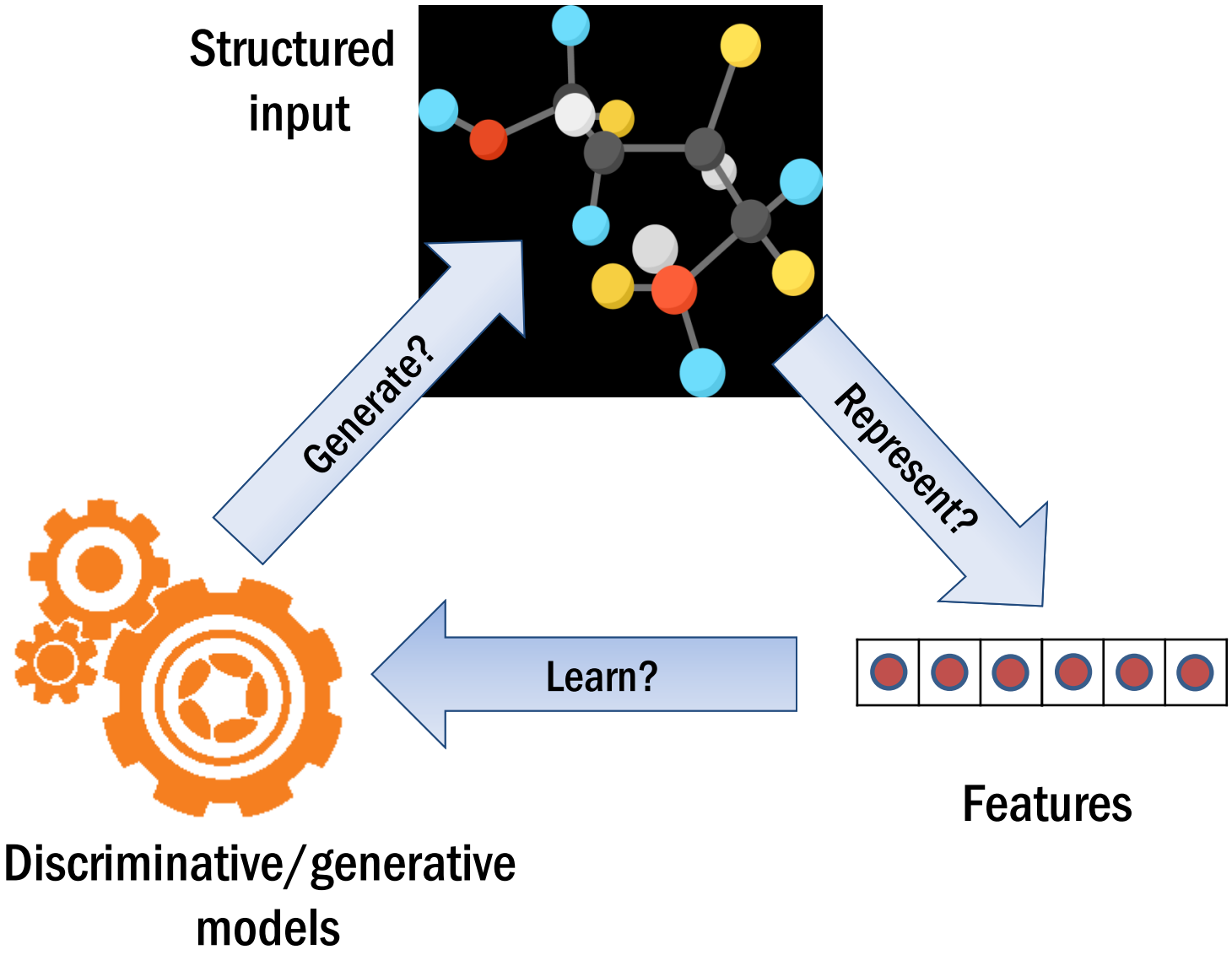
College of Computing

Georgia Institute of Technology

What are structured input data?

	String / Sequence	Graph
Example	 RNA Sequence	
Task	Binding /not binding?	Effective / ineffective?

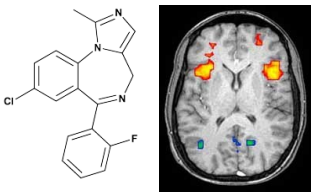
Three key questions



Kernel methods

A modular framework to handle all kinds of complex data

Theoretical property well understood



ACAAGAT
GCCATTG
TCCCCCG

Kernel
$k(x_i, x_j)$

Kernel matrix
$K_{ij} = k(x_i, x_j)$

$m \times m$

Algorithm

$$f(x) = \sum_{i=1}^{m} \alpha_i k(x_i, x)$$

Polynomial kernel : $(x^\top x' + c)^d$

Gaussian kernel: $\exp\left(-\frac{\|x-x'\|^2}{2\sigma^2}\right)$

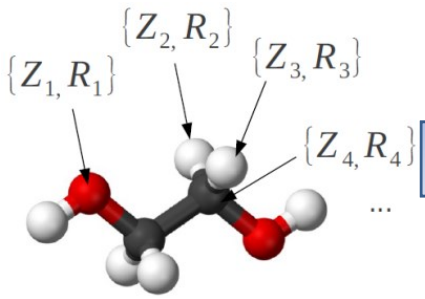
Time-series kernel
Sequence kernel
Graph kernel
Tree kernel
Pyramid kernel ...

Support vector machines
Fisher discriminant analysis
Logistic regression
Ridge regression
Novelty detection
Principal component analysis
Canonical correlation analysis
Clustering
Two-sample test
Independence test ...

[Scholkopf & Smola 02]
[Shaw-Taylor & Cristianini 04]

Coloumb matrix

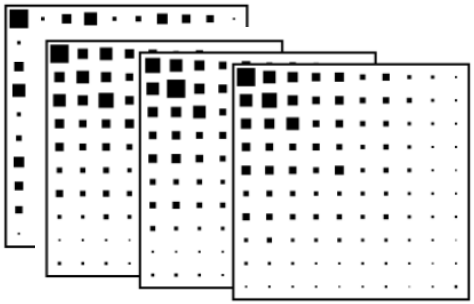
Molecular structure



represent

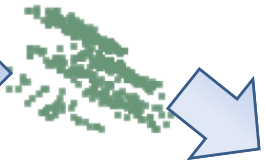
C =

Permutation invariant

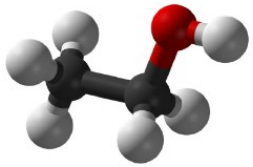


equivalent

Structured distribution

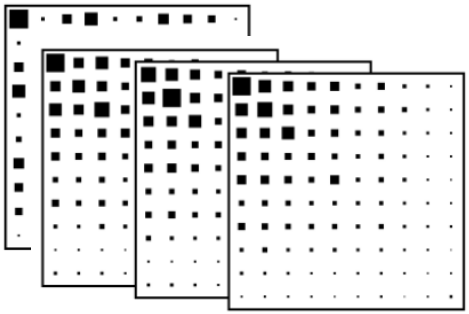


KRR

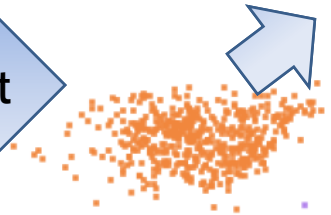


represent

C =



equivalent



Z: Charges

R: coordinates

Coulomb matrix:

$$C_{ii} = Z_i^{2.4}$$

$$C_{ij} = Z_i Z_j / \left\| R_i - R_j \right\|$$

Bag of structure (BOS) kernel

Two structured data points χ and χ', a dictionary of substructures S

$$k(\chi, \chi') = \sum_{s \in S} \#[s \in \chi] \cdot \#[s \in \chi']$$

Count occurrence

Eg. RNA sequences (alphabet: A, G, C, U)

S = { CUU, UUC, …, CAG, AGU }

χ: C U U C A G U

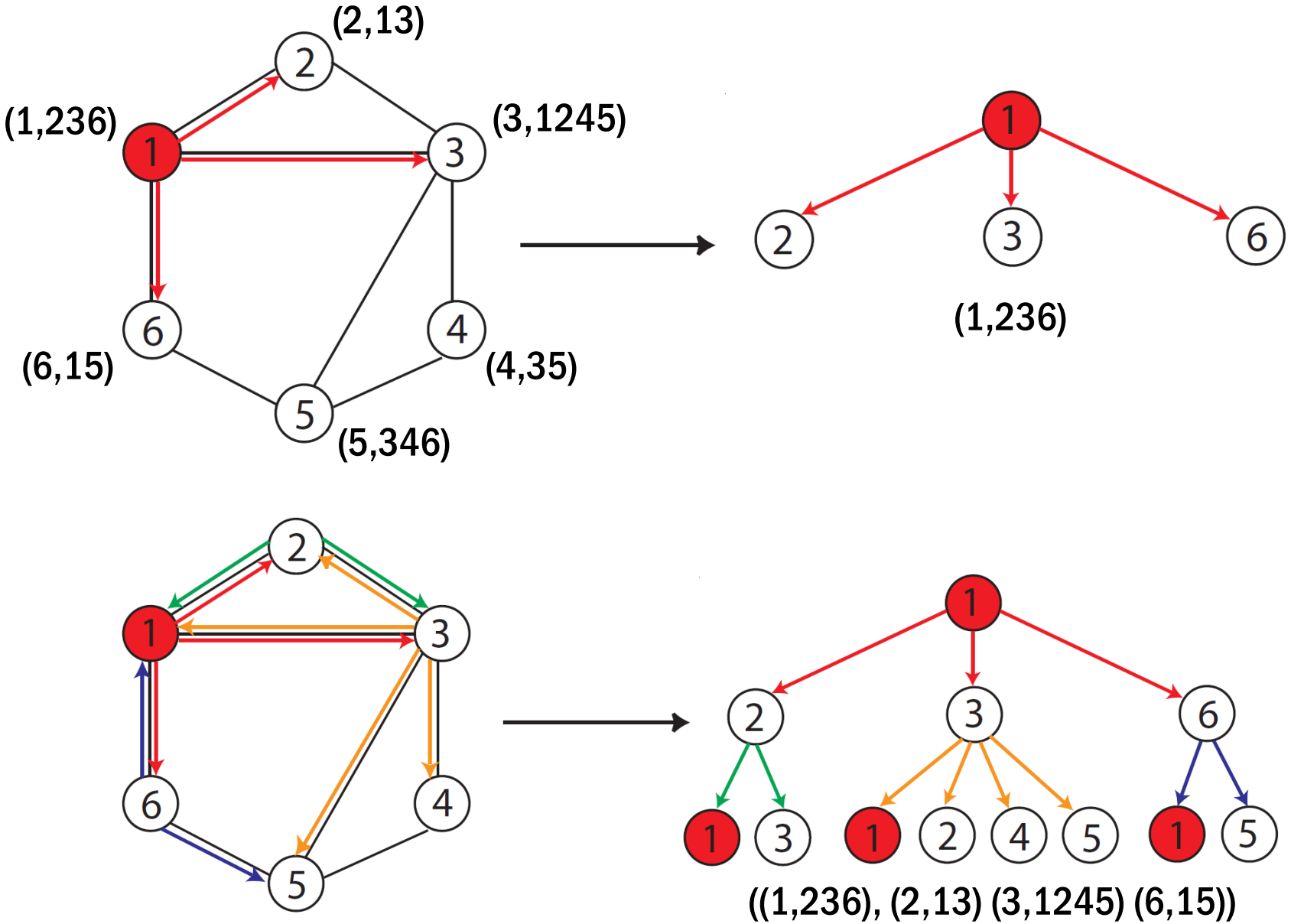
χ': C U U A C U U

$$\phi(\chi) = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{bmatrix} \begin{matrix} \text{CUU} \\ \text{UUC} \\ \vdots \\ \text{CAG} \\ \text{AGU} \end{matrix} \qquad \phi(\chi') = \begin{bmatrix} 2 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix} \begin{matrix} \text{CUU} \\ \text{UUC} \\ \vdots \\ \text{CAG} \\ \text{AGU} \end{matrix}$$

$$k(\chi, \chi') = \langle \phi(\chi), \phi(\chi') \rangle$$

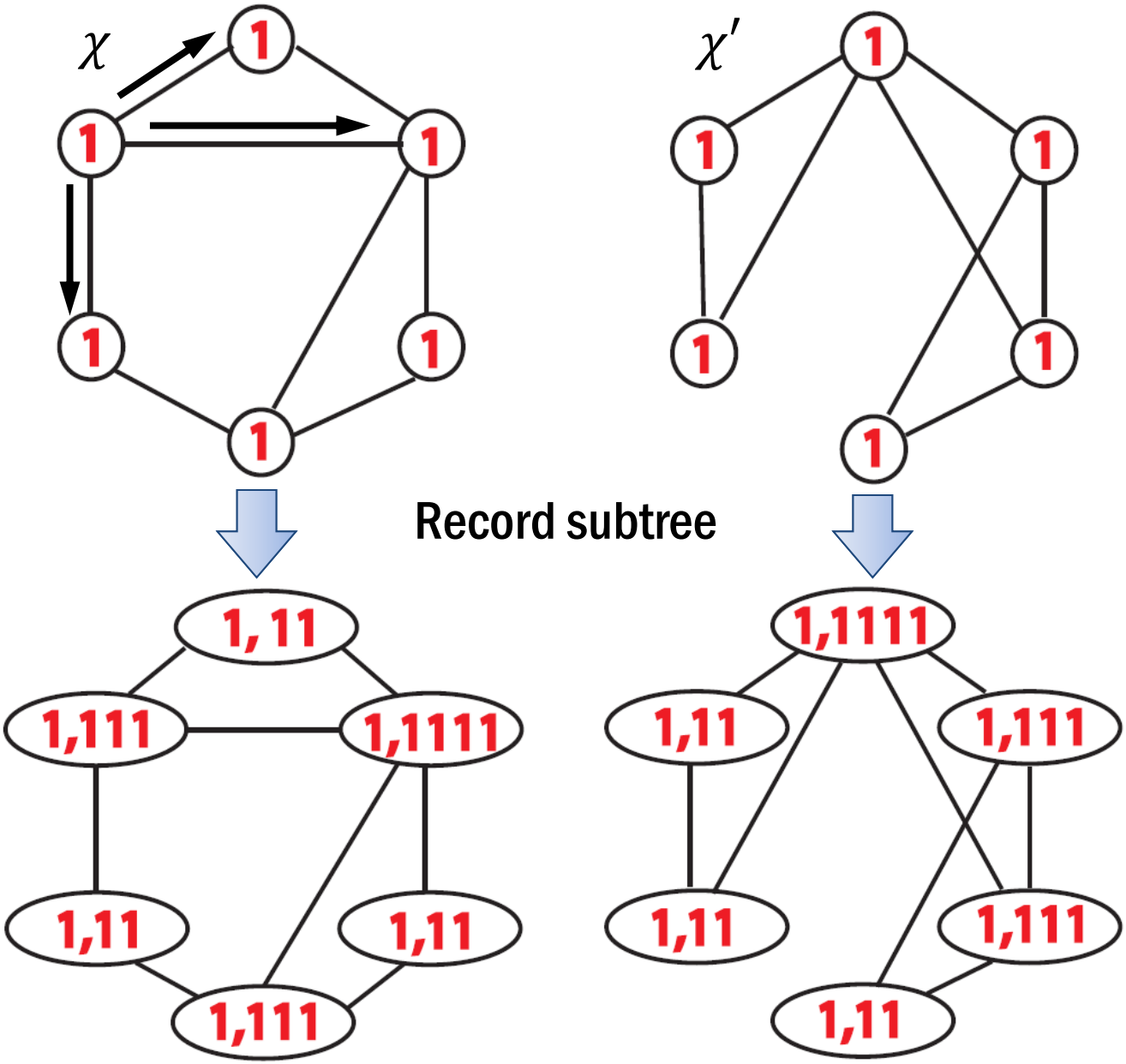
[Leslie et al. 04]

Bag of subtrees for graph

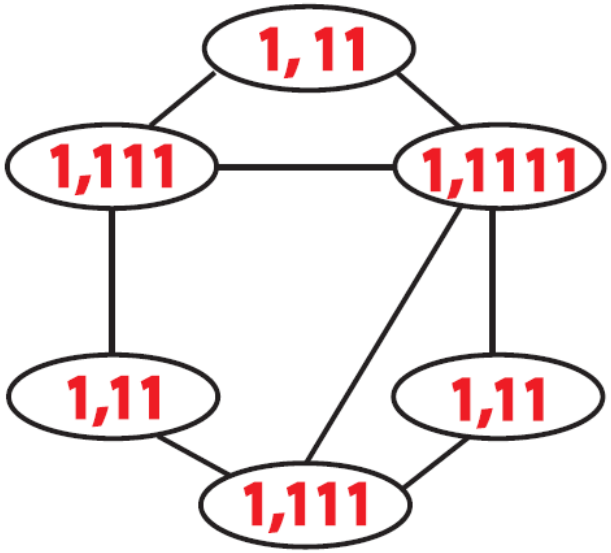


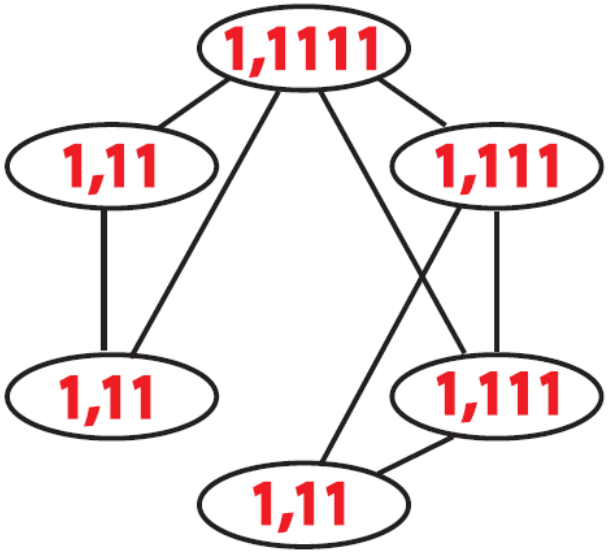
[Shervashidze et al. 11]

1. Record subtree



2. Re-number (or hash)



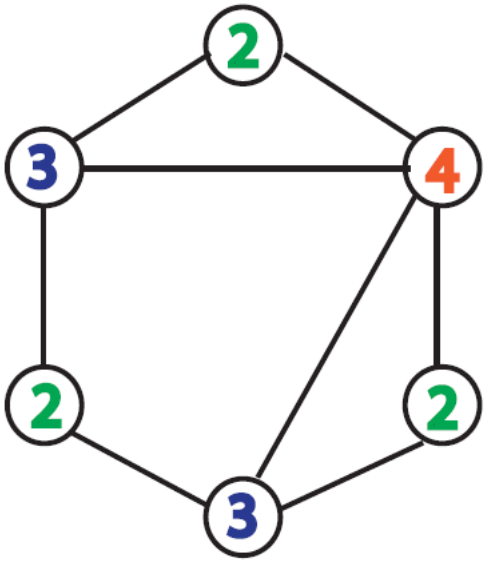


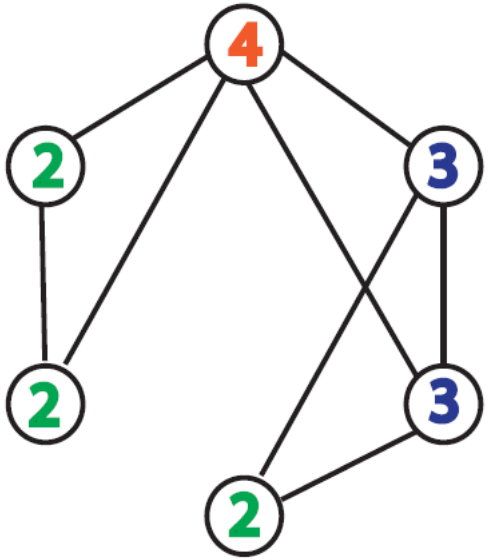
Re-number nodes

1, 11 → 2

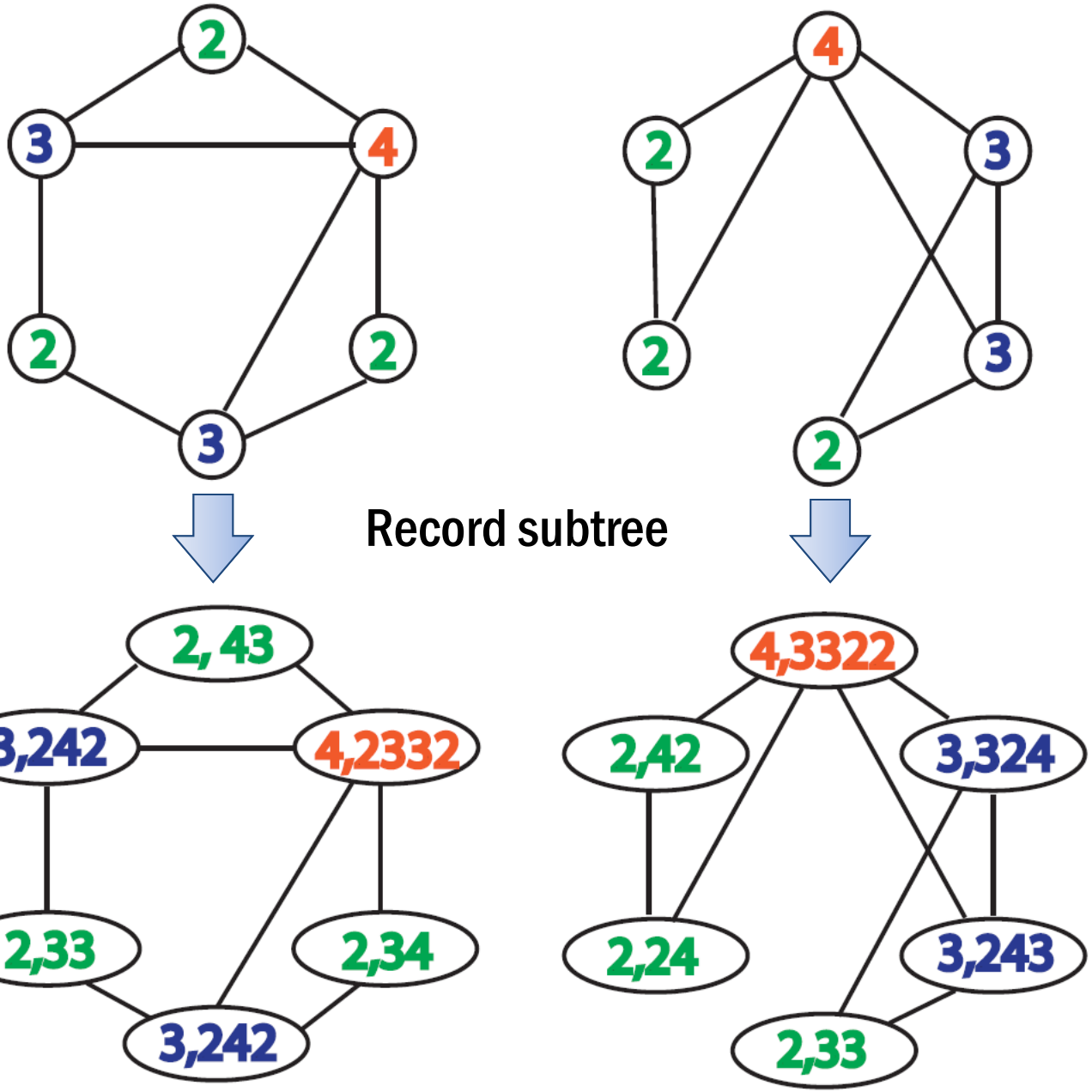
1,111 → 3

1,1111 → 4

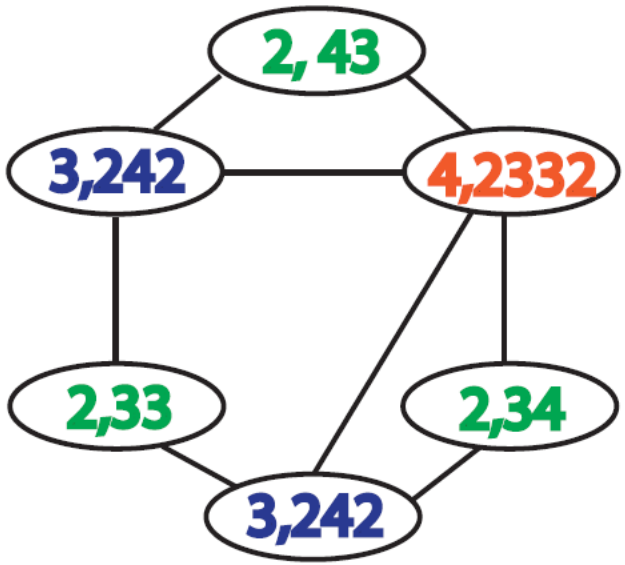


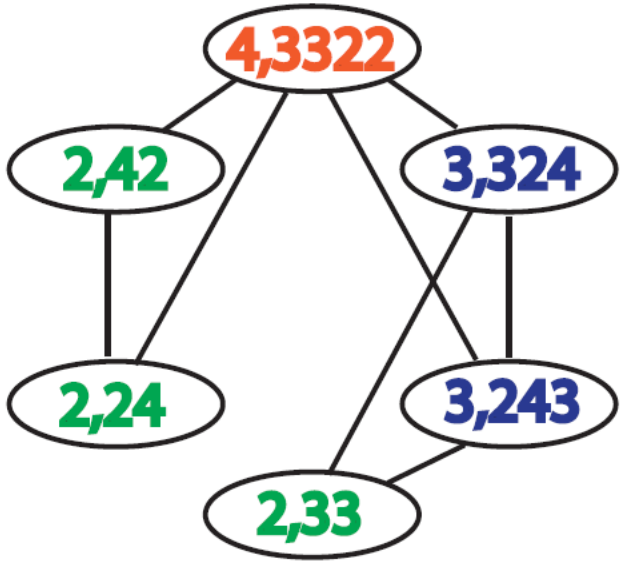


3. Record subtree again

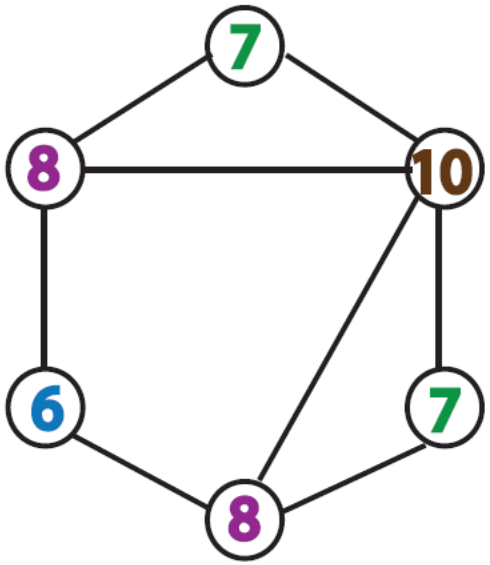


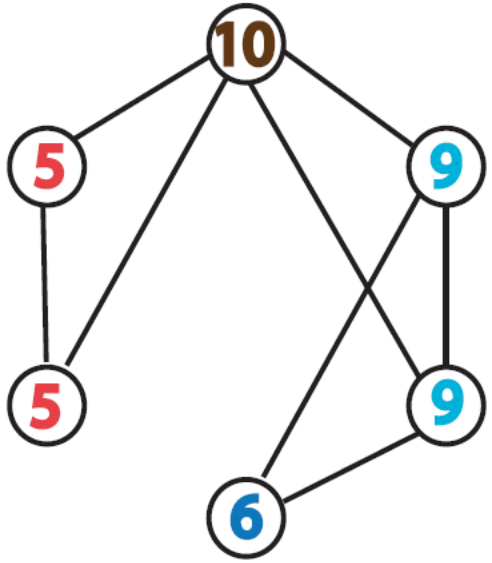
4. Re-number (or hash) again





Re-number nodes





2,24 → 5

2,33 → 6

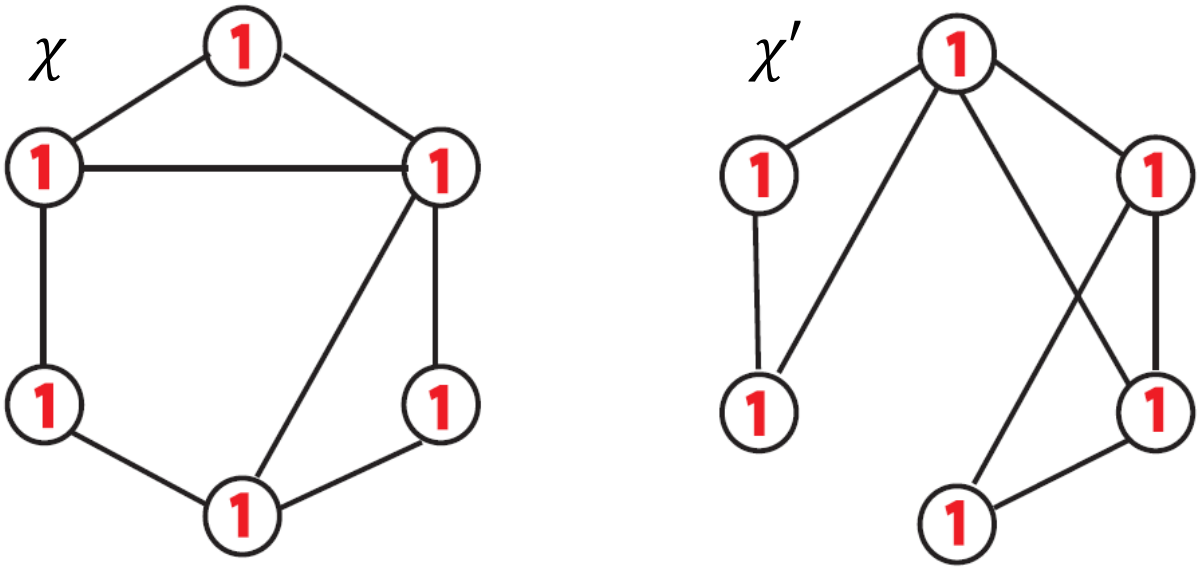
2,34 → 7

3,224 → 8

3,234 → 9

4,2233 → 10

Final features



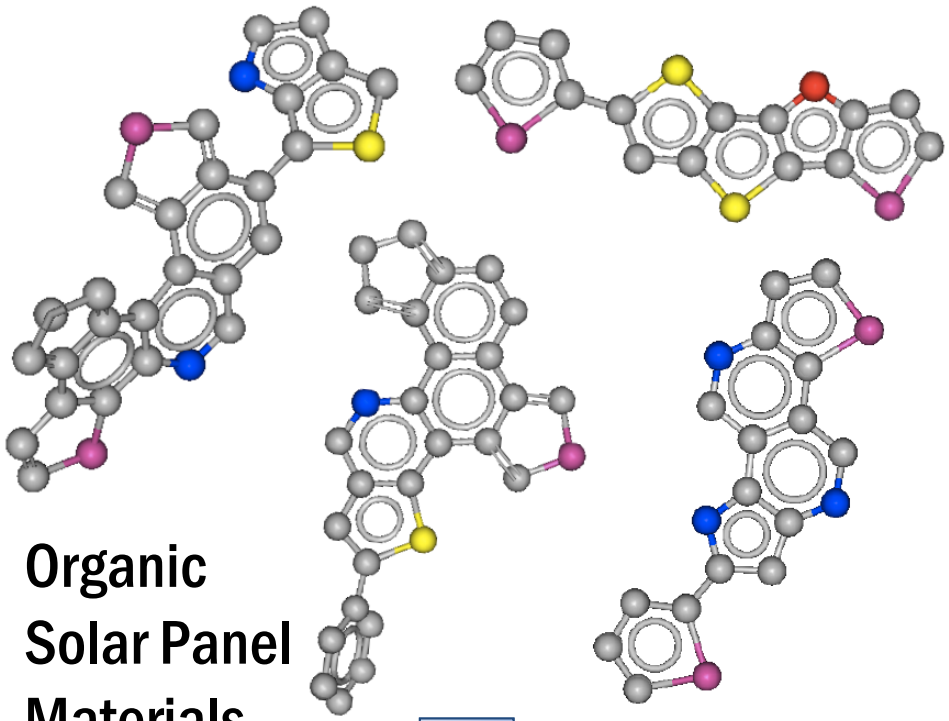
1 2 3 4 5 6 7 8 9 10 … … … …

$$\phi(\chi) = (6, 3, 2, 1, 0, 1, 2, 2, 0, 1, \ldots\ldots\ldots\ldots)^{\top}$$

$$\phi(\chi') = (6, 3, 2, 1, 2, 1, 0, 0, 2, 1, \ldots\ldots\ldots\ldots)^{\top}$$

Level 0 feature — Level 1 feature — Level 2 feature — Higher level features

Big dataset, explosive feature space



Organic
Solar Panel
Materials

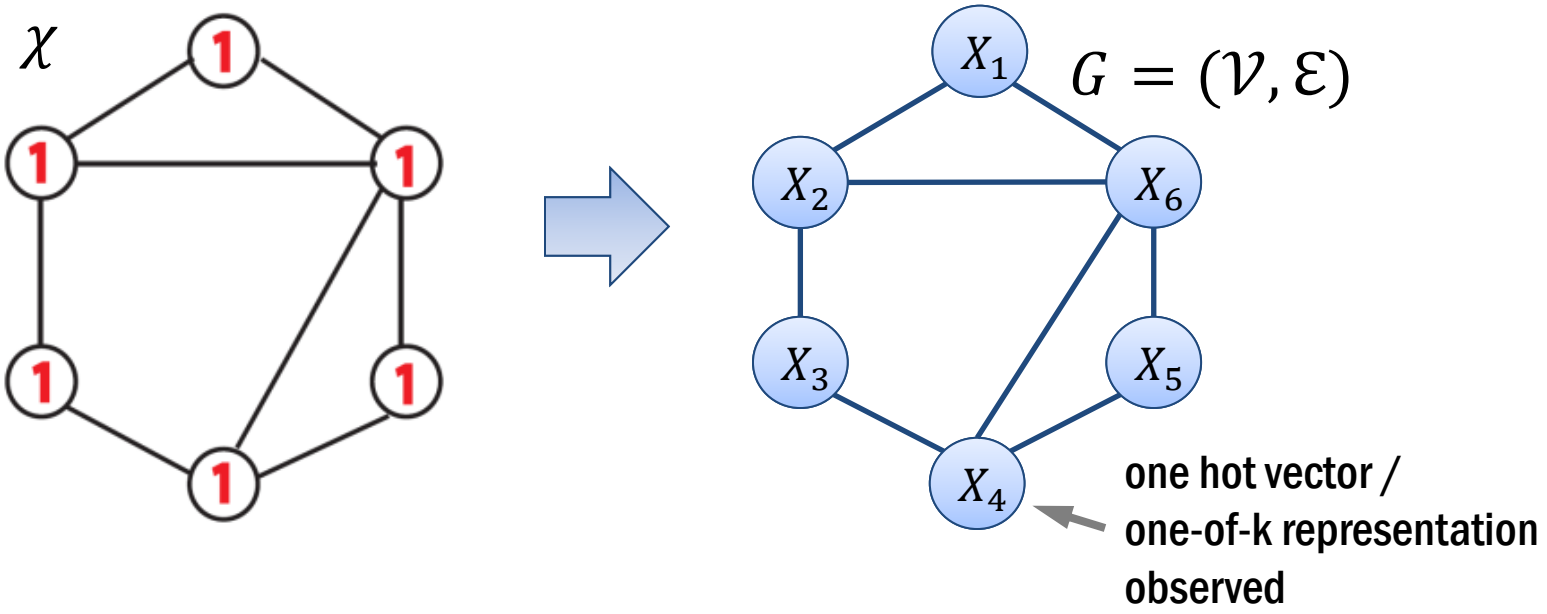
predict

Power Conversion Efficiency (PCE)
(0 -12 %)

Dataset	Harvard clean energy project
Size	2.3 million
Type	Molecule
Alphabet #	6
Avg node #	28
Avg edge #	33

feature	dimension	MAE
Level 3	1.6 million	0.143
Level 6	1.3 billion	0.096

Kernels based on graphical models (GM)



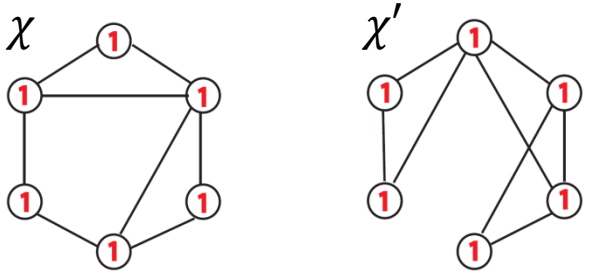
Define conditional independence structure
using structure of the data

Model each data point as a Markov random field

$$p(\chi|\theta) \propto \prod_{i \in \mathcal{V}} \Psi_1(X_i|\theta) \prod_{(i,j) \in \mathcal{E}} \Psi_2(X_i, X_j|\theta)$$

[Jaakkola et al. 99] [Jebara et al. 04]

Fisher kernel vs. probability product kernel



Each structured data point

$\Leftrightarrow$

a graphical model

	Fisher Kernel	Probability product kernel
Share parameter?	Yes	No
How to learn?	$\theta^* = \arg\max_\theta \sum_{i=1}^{m} \log p(\chi_i \mid \theta)$ For all data points together	$\theta_\chi = \arg\max_\theta \log p(\chi \mid \theta)$ For each χ
Kernel form $k(\chi, \chi')$	$U_\chi^\top I^{-1} U_{\chi'}$ Fisher vector: $U_\chi = \nabla_\theta \log p(\chi \mid \theta^*)$ Fisher information matrix: $I = \mathbb{E}\left[U_\chi U_\chi^\top\right]$	$\int p(\chi \mid \theta_\chi)^{1/2} p(\chi' \mid \theta_{\chi'})^{1/2}$

Need to perform many graphical model inferences

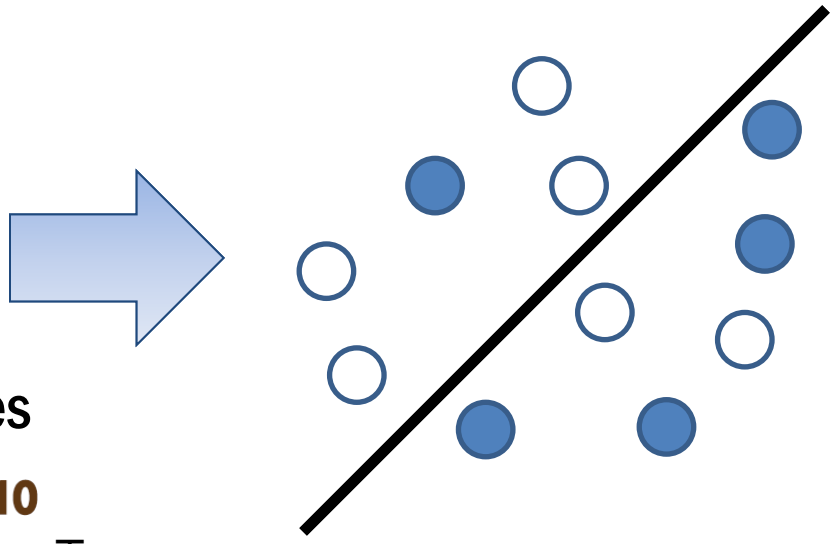
[Jaakkola et al. 99] [Jebara et al. 04]

Limitation of existing two-stage approach

Stage 1	Stage 2

Construct big kernel matrix

$$K_{ij} = k(x_i, x_j)$$



Or compute high-d explicit features

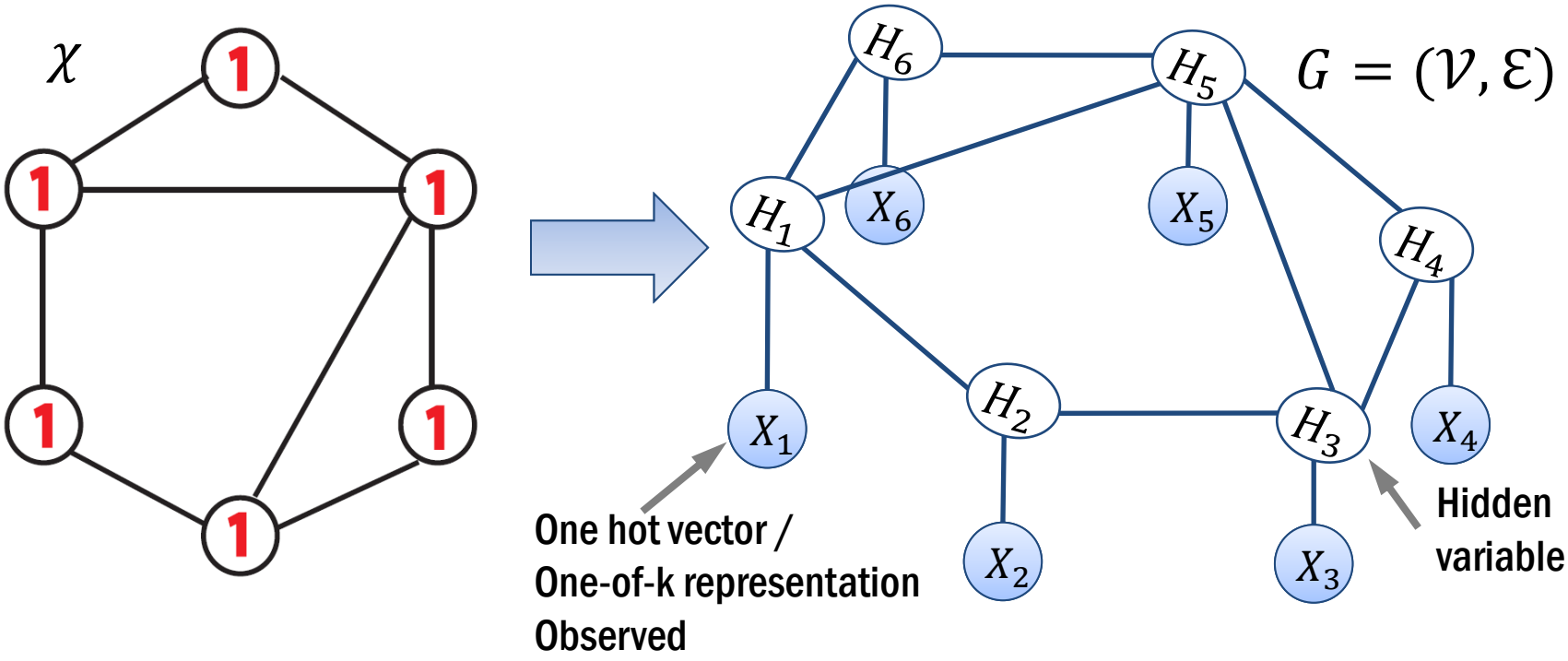
1 2 3 4 5 6 7 8 9 10

$$\phi(\chi) = (6, 3, 2, 1, 0, 1, 2, 2, 0, 1)^{\top}$$

Or learn many graphical models

Not scalable for large datasets	Feature construction not aware of supervised tasks

Represent data point as latent variable model

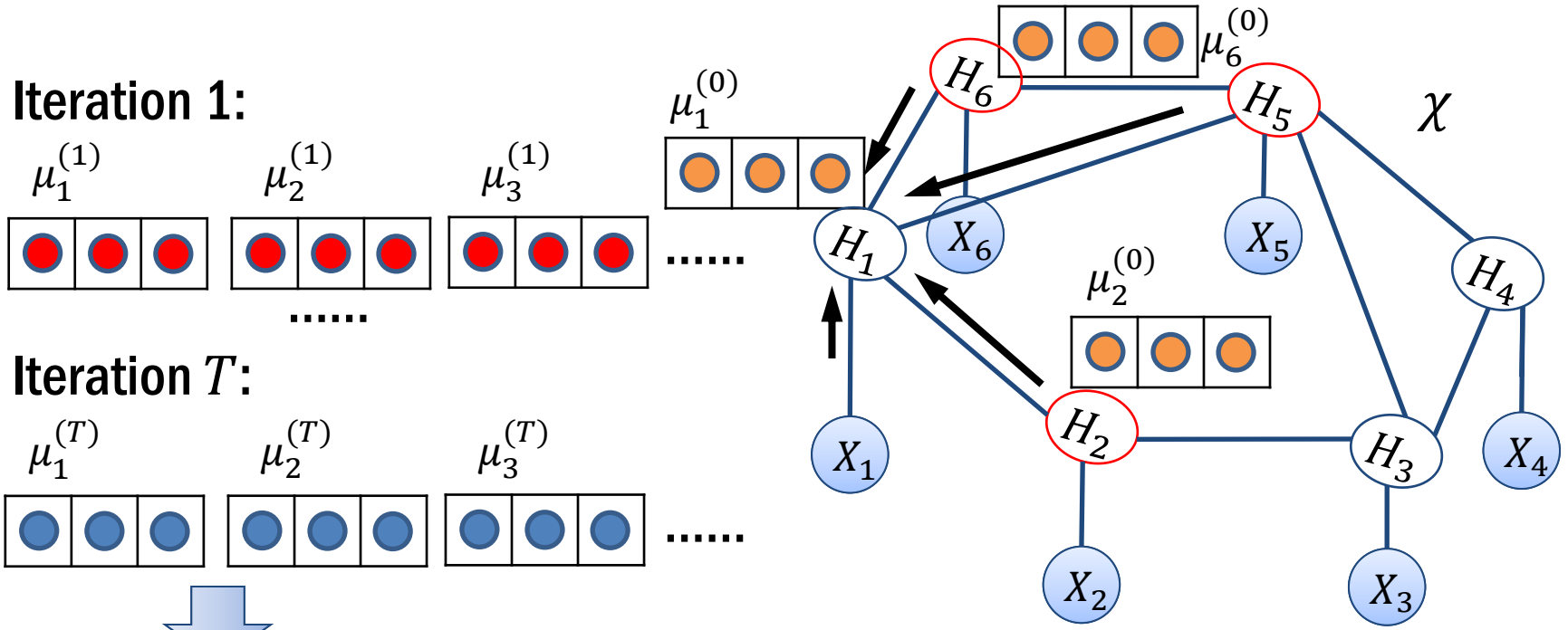


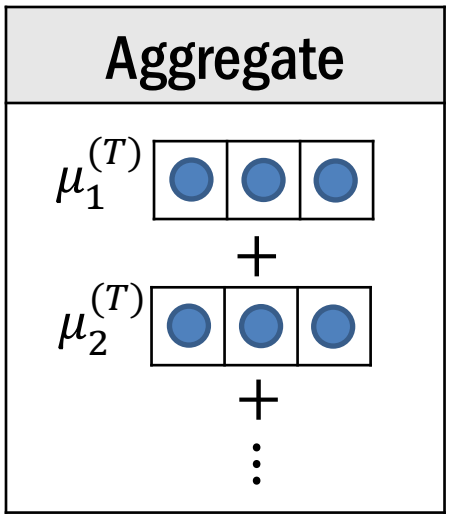
Model each data point as a Markov random field with latent variables

$$p(\{H_i\}, \{X_i\}|\theta) \propto \prod_{i \in \mathcal{V}} \Psi_1(H_i, X_i|\theta) \prod_{(i,j) \in \mathcal{E}} \Psi_2(H_i, H_j|\theta)$$

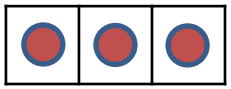
[Dai et al. 16]

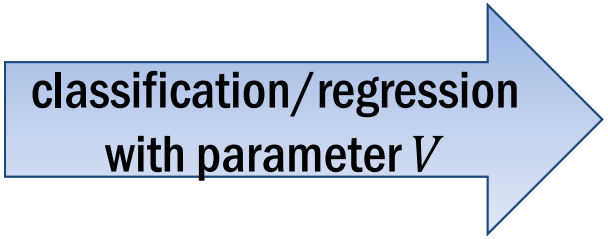
Recursive nonlinear feature extraction





$= \mu_a(W, \chi)$





Label y

Parameter learning

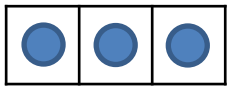
Estimate parameters W and V which minimize empirical loss

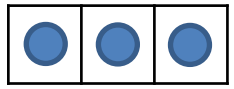
$$\min_{V,W} L(V,W) := \sum_{i=1}^{m} (y_i - V^\top \mu^a(W, \chi_i))^2$$

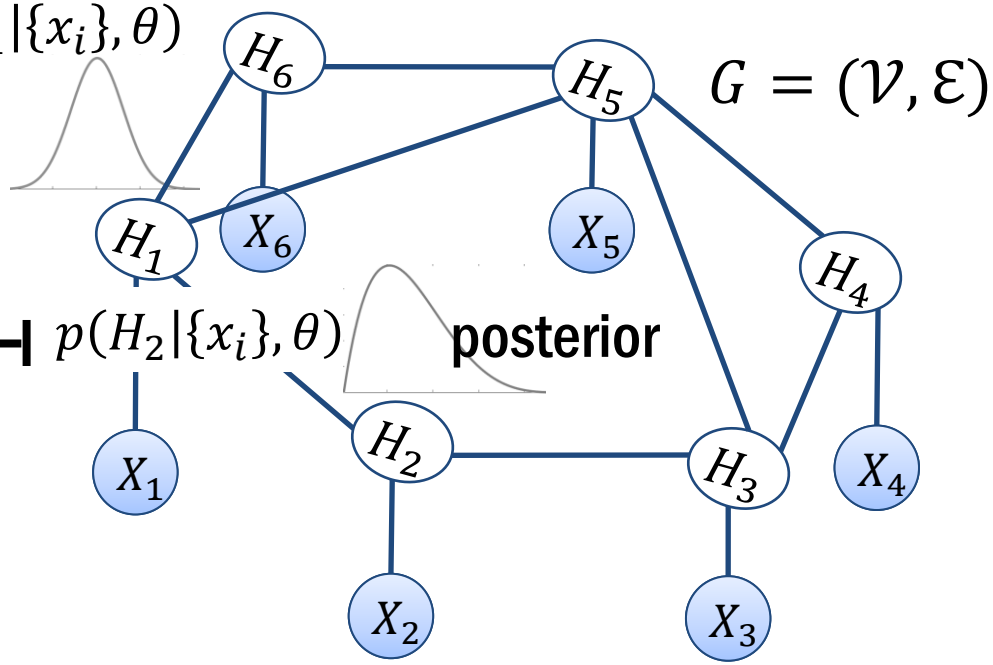
Computation	Operation	Similar to
Objective $L(V,W)$	A forward sequence of nonlinear mappings	Graphical model inference
Gradient $\frac{\partial L}{\partial W}$	Chain rule of derivatives in reverse order of the mappings	Back propagation in deep learning

Posterior embedding as features

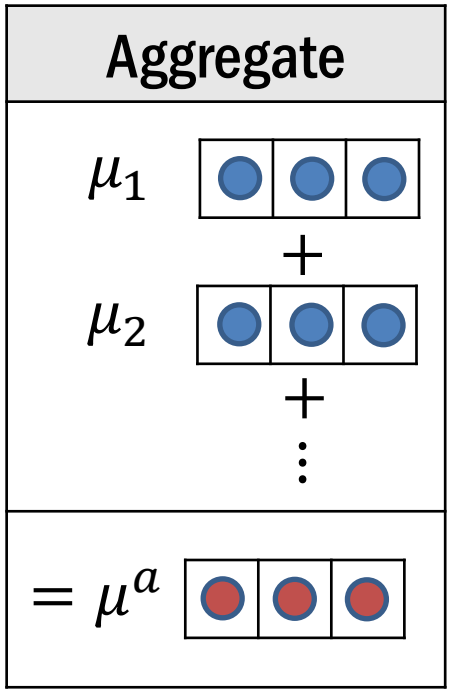
Embedding

μ_1  $\mathbb{E}[\phi(H_1)]$

μ_2  $\mathbb{E}[\phi(H_2)]$

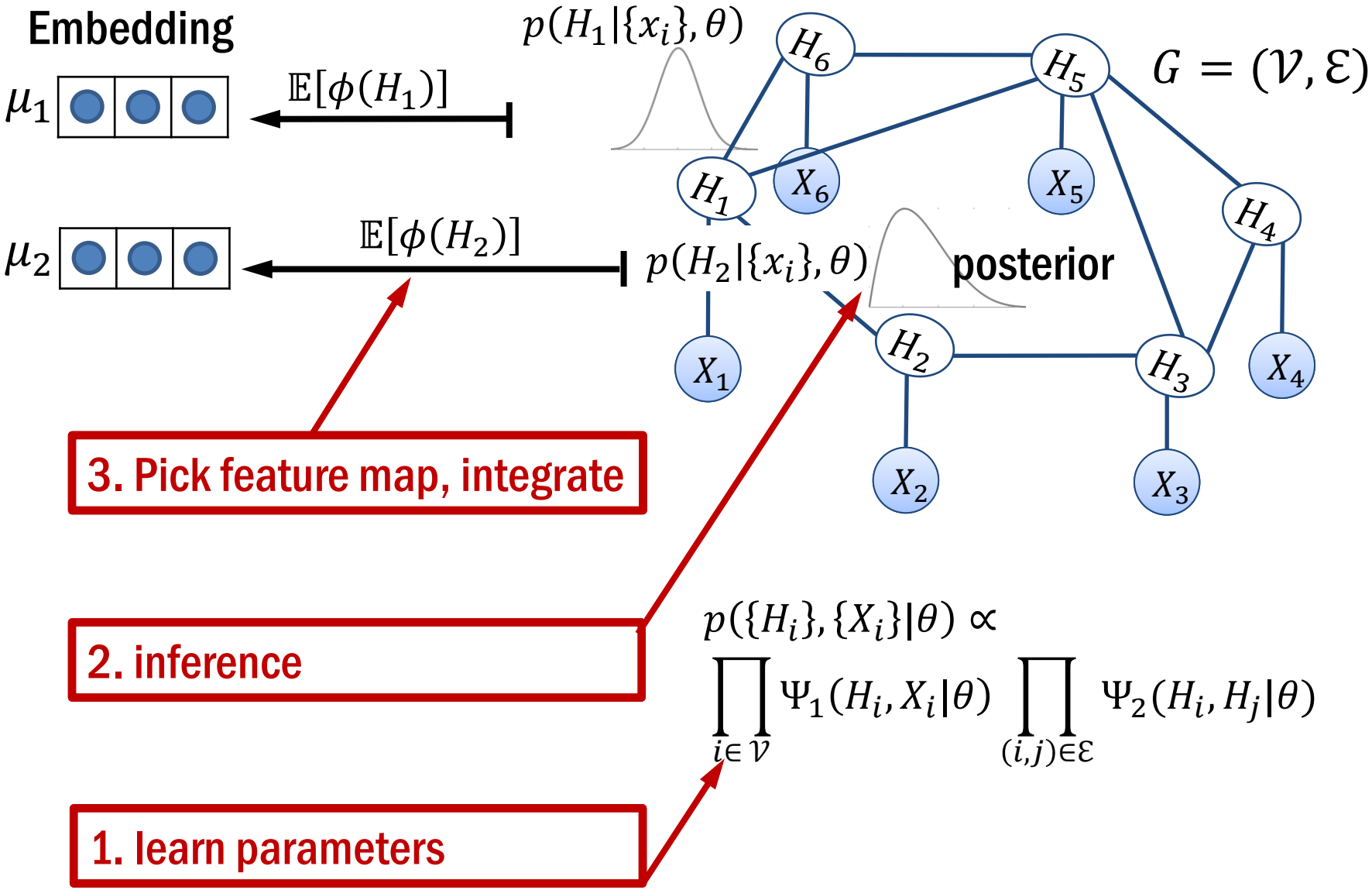


Feature of a data point χ



$$p(\{H_i\},\{X_i\}|\theta) \propto \prod_{i\in\mathcal{V}} \Psi_1(H_i, X_i|\theta) \prod_{(i,j)\in\mathcal{E}} \Psi_2(H_i, H_j|\theta)$$

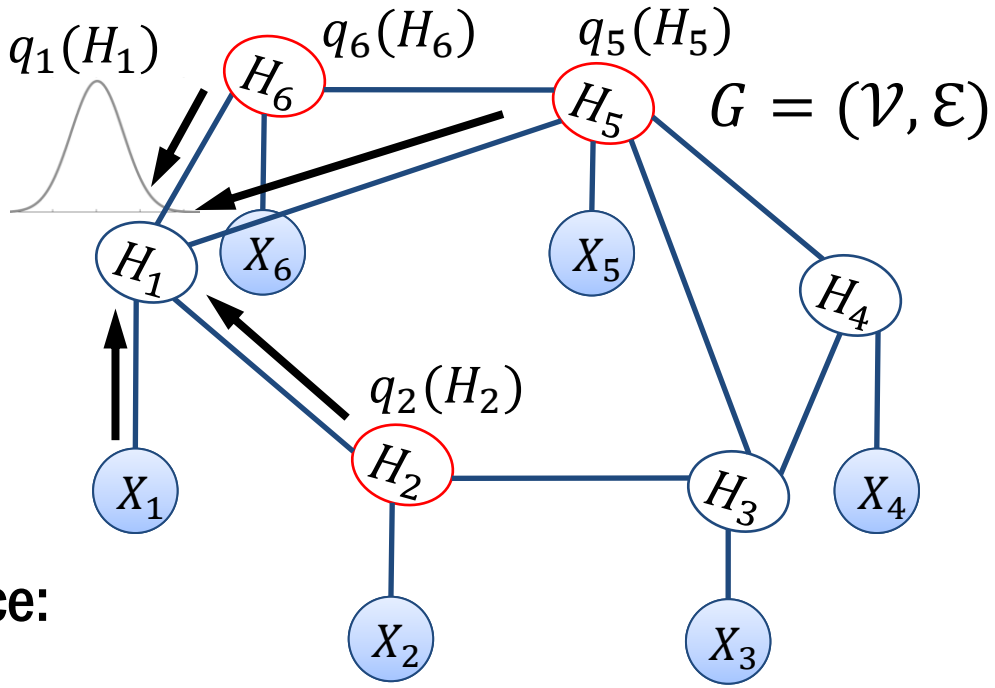
Posterior embedding as features



Operator view of mean field inference

Approximate joint posterior

$$p\left(H_1, \ldots, H_{|\mathcal{V}|} \middle| \{x_j\}, \theta\right) \approx \prod_{i \in \mathcal{V}} q_i(H_i)$$



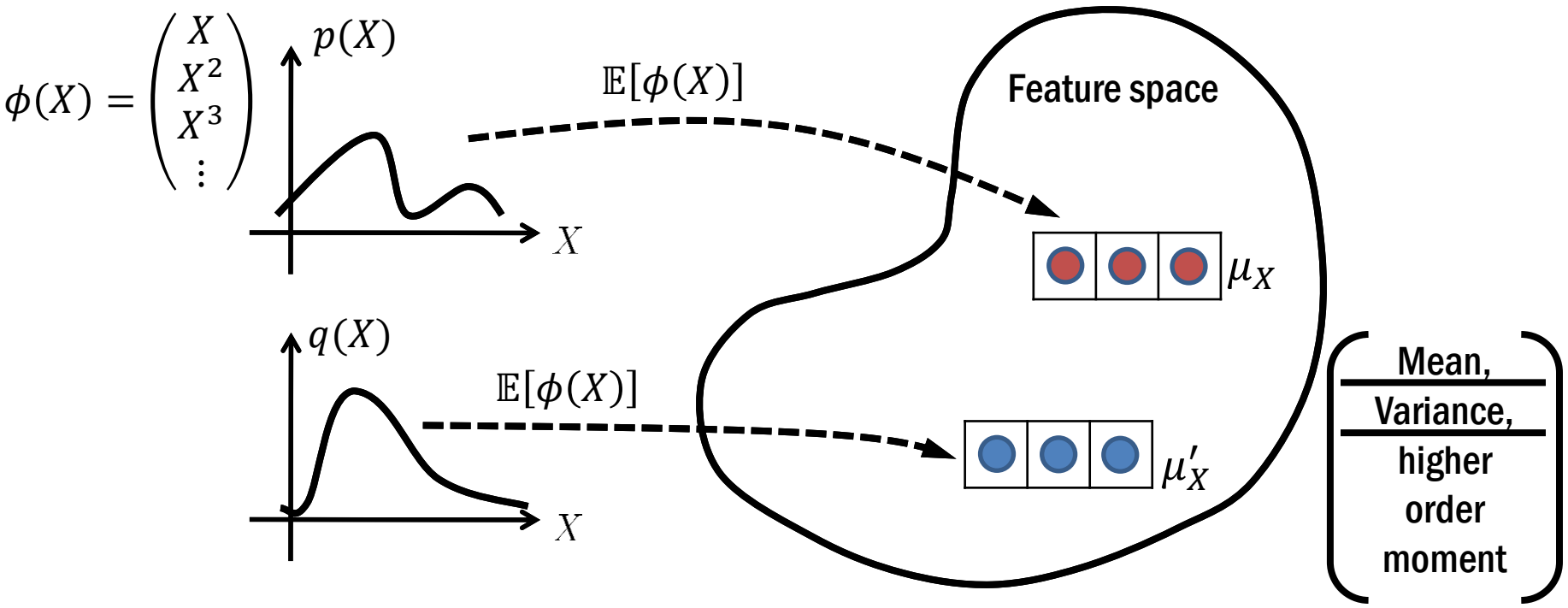
For each i, iterate till convergence:

$$q_i(H_i) \leftarrow \Psi_1(H_i, x_i|\theta) \cdot \prod_{j \in \mathcal{N}(i)} \exp\left(\int_{\mathcal{H}} q_j(H_j) \log\left(\Psi_2\left(H_i, H_j|\theta\right)\right) dH_j \right)$$

Operator view

$$q_i(H_i) \leftarrow \mathcal{T}(\theta) \circ \left(x_i, \{q_j(H_j)\}_{j \in \mathcal{N}(i)} \right)$$

Hilbert (feature) space embedding of distribution



Rich representation: $p(x) = q(x)$ **iff.** $\|\mu_X - \mu'_X\|^2$

One-to-one mapping: μ_X **is a sufficient statistic of** $p(X)$

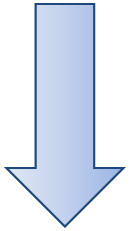
Operator $\mathcal{T}: \mathcal{P} \mapsto \mathcal{H}$
$\boldsymbol{\mathcal{T}} \circ p(x) = \widetilde{\boldsymbol{\mathcal{T}}} \circ \mu_X$

[Smola, Gretton, Song and Scholkopf. 2007]
[Song et al. 10a,b, 11a,b]

Mean field update using embedding

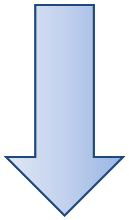
Operator view

$$q_i(H_i) \leftarrow \boldsymbol{\mathcal{T}}(\theta) \circ \left(x_i, \{q_j(H_j)\}_{j \in \mathcal{N}(i)} \right)$$



Represent as embedding

$$q_i(H_i) \Rightarrow \mu_i = \int_{\boldsymbol{\mathcal{H}}} \phi(H_i) q_i(H_i)\, dH_i$$



Embedding update

$$\mu_i \leftarrow \widetilde{\boldsymbol{\mathcal{T}}} \circ \left(x_i, \{\mu_j\}_{j \in \mathcal{N}(i)} \right)$$

Parameterize mean field update

$$\mu_i \leftarrow \widetilde{\mathcal{T}}(W) \circ \left(x_i, \{\mu_j\}_{j \in \mathcal{N}(i)}\right)$$

One can parameterize it as any flexible nonlinear model

Assume $\mu_i \in \mathcal{R}^d, x_i \in \mathcal{R}^n$, parametrize as one-layer neural network

$$\mu_i \leftarrow \sigma\left(W_1 x_i + W_2 \sum_{j \in \mathcal{N}(i)} \mu_j\right)$$

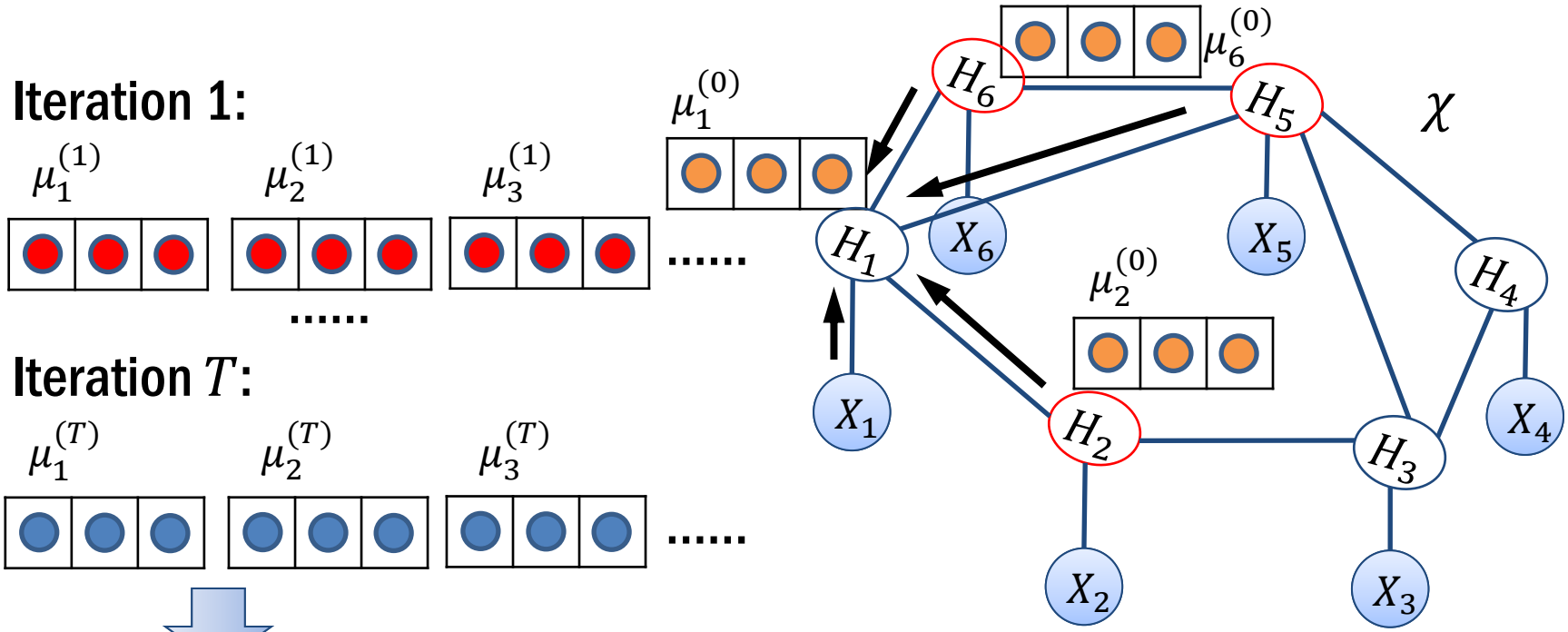
$Max\{0,\cdot\}$

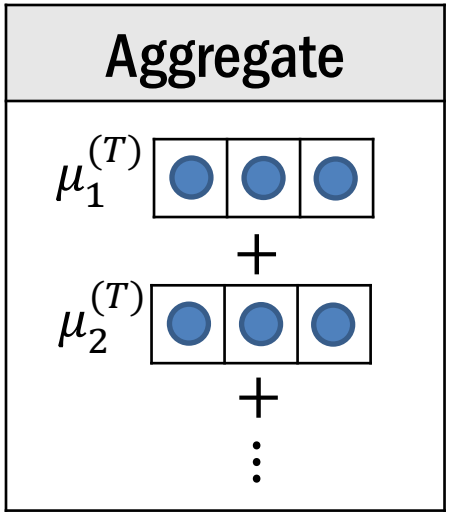
$d \times n$ matrix

$d \times d$ matrix

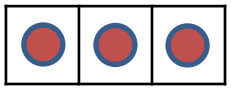
Learn with supervision

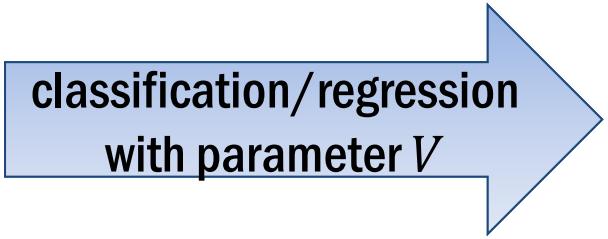
Mean field iterations as nonlinear mappings





$= \mu_a(W, \chi)$



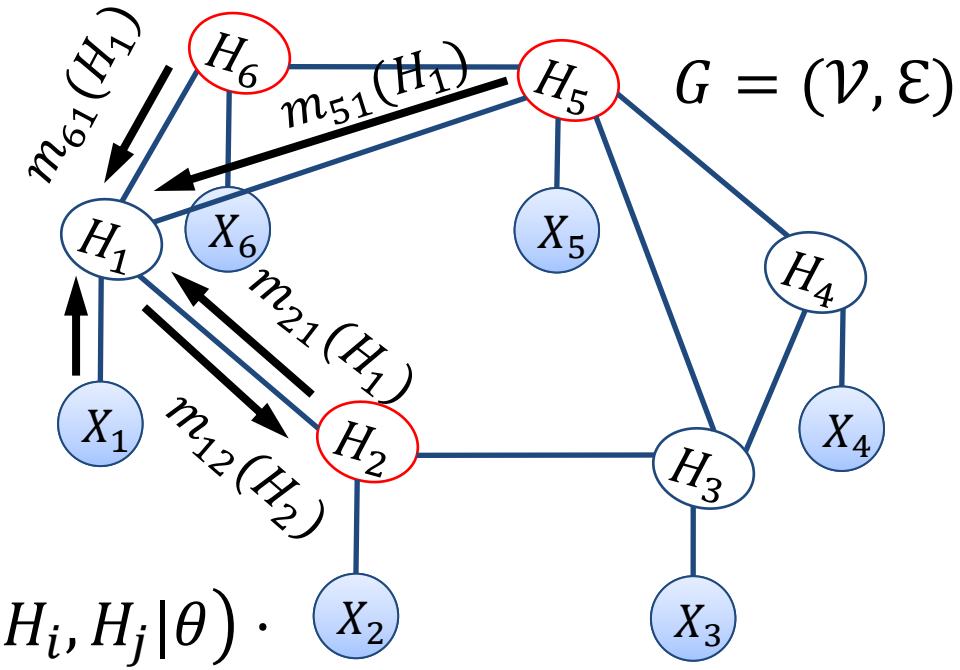


Label y

Benefit of the new view: belief propagation

Approximate $p\left(H_i \middle| \{x_j\}, \theta\right)$ as

$$q_i(H_i) = \Psi_1(H_i, x_i|\theta) \prod_{j \in \mathcal{N}(i)} m_{ji}(H_i)$$



with messages updated iteratively:

$$m_{ij}\left(H_j\right) \leftarrow \int_{\mathcal{H}} \Psi_1(H_i, x_i|\theta)\Psi_2\left(H_i, H_j|\theta\right) \cdot \prod_{\ell \in \mathcal{N}(i)\backslash \mathrm{j}} m_{\ell i}(H_i)\, dH_i$$

Operator view

$$q_i(H_i) \leftarrow \boldsymbol{\mathcal{T}}(\theta) \circ \left(x_i, \{m_{ji}(H_j)\}_{j \in \mathcal{N}(i)} \right)$$

$$m_{ij}\left(H_j\right) \leftarrow \boldsymbol{\mathcal{T}}'(\theta) \circ \left(x_i, \{m_{\ell i}(H_i)\}_{\ell \in \mathcal{N}(i)\backslash \mathrm{j}}\right)$$

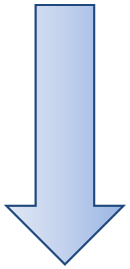
[Song et al. 11a,b]
[Song et al. 10a,b]

Belief propagation using embedding

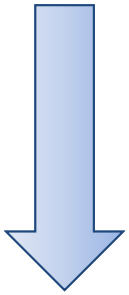
Operator view

$$q_i(H_i) \leftarrow \boldsymbol{\mathcal{T}}(\theta) \circ \left(x_i, \{m_{ji}(H_j)\}_{j \in \mathcal{N}(i)} \right)$$

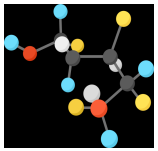
$$m_{ij}(H_j) \leftarrow \boldsymbol{\mathcal{T}}'(\theta) \circ \left(x_i, \{m_{\ell i}(H_i)\}_{\ell \in \mathcal{N}(i) \setminus \mathrm{j}} \right)$$



Embed $q_i(H_i) \Rightarrow \mu_i, \quad m_{ij}(H_j) \Rightarrow \nu_{ij}$

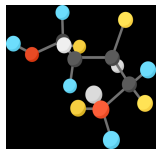


$$\mu_i = \widetilde{\boldsymbol{\mathcal{T}}} \circ \left(x_i, \{\nu_{ji}\}_{j \in \mathcal{N}(i)} \right), \nu_{ij} = \widetilde{\boldsymbol{\mathcal{T}}}' \circ \left(x_i, \{\nu_{\ell i}\}_{\ell \in \mathcal{N}(i) \setminus \mathrm{j}} \right)$$

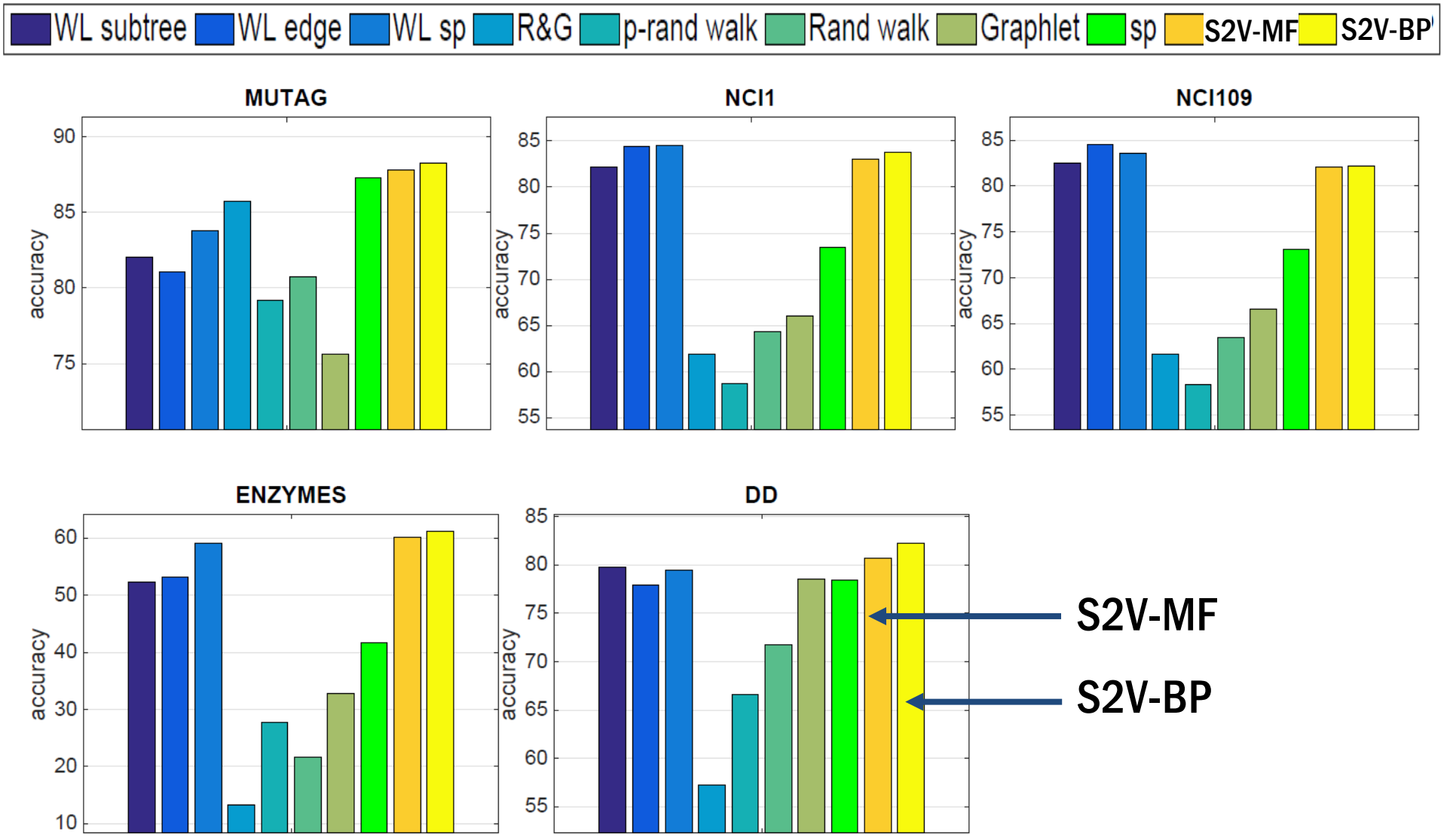


Experiment II: drug prediction

dataset	MUTAG	NCI1	NCI109	ENZYMES	D&D
type	Chem	Chem	Chem	Protein	Protein
#sample	188	4110	4127	600	1178
#class	2	2	2	6	2
Avg $\|V\|$	18	30	30	33	284
Avg $\|E\|$	20	32	32	62	715
Alphabet size	7	37	38	3	82
Real world problem	Mutagenic / non-mutagenic compounds for *Salmonella Typhimurium*	Active / inactive compounds in an anti-cancer screen	Active / inactive compounds in an anti-cancer screen	Enzymes / non-enzymes	Enzymes / non-enzymes

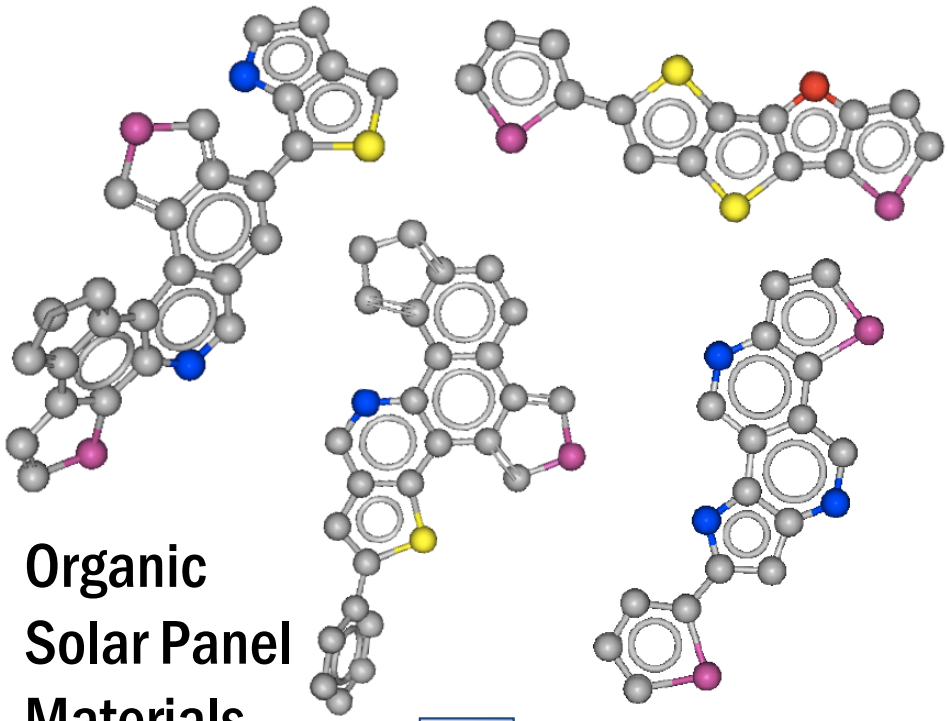


Experiment III: graph classification



[Dai et al. 16]

Predicting efficiency of solar panel materials



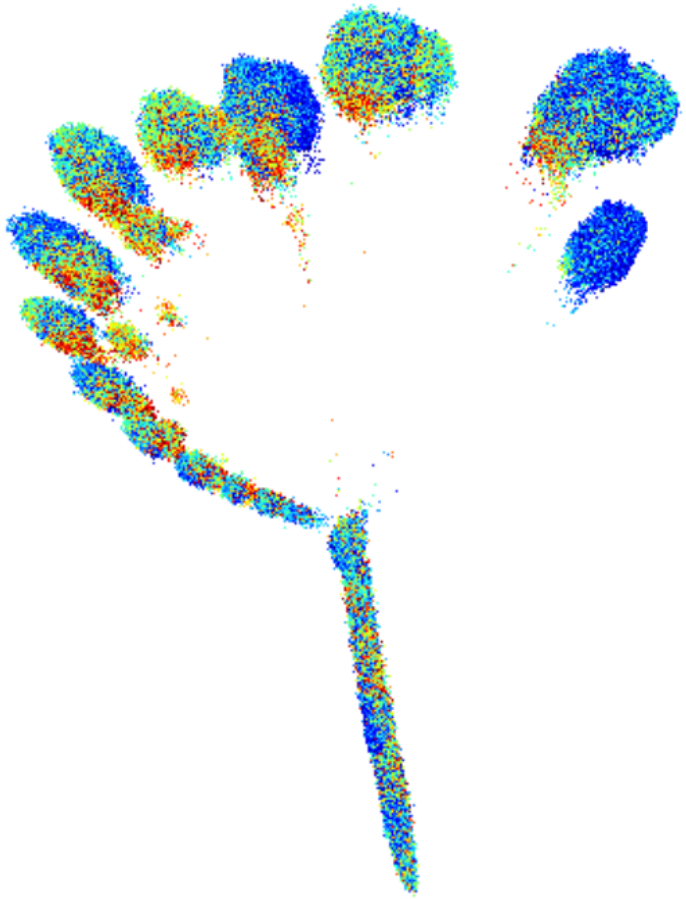
Organic
Solar Panel
Materials

predict

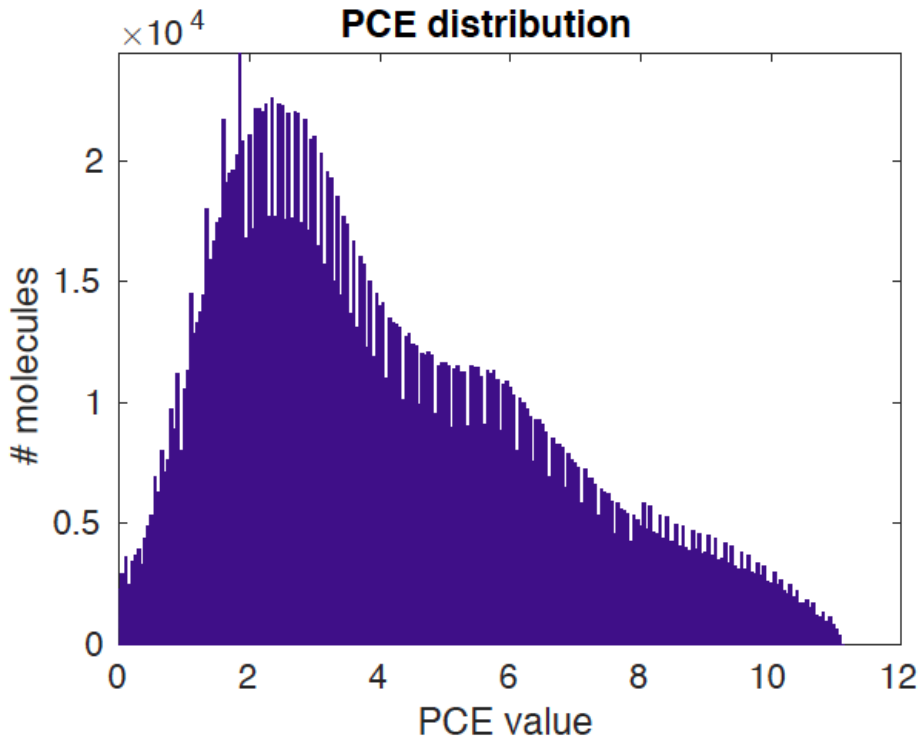
Power Conversion Efficiency (PCE)
(0 -12 %)

Dataset	Harvard clean energy project
Size	2.3 million
Type	Molecule
Alphabet #	6
Avg node #	28
Avg edge #	33

Characteristics of the data

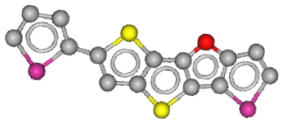


Clusters of data points



Unbalanced output value
Resample data during training
to make it balance

[Dai et al. NIPS 14]
[Xie et al. NIPS 15]



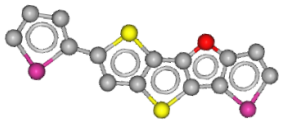
Small model but accurate prediction

10% data for testing

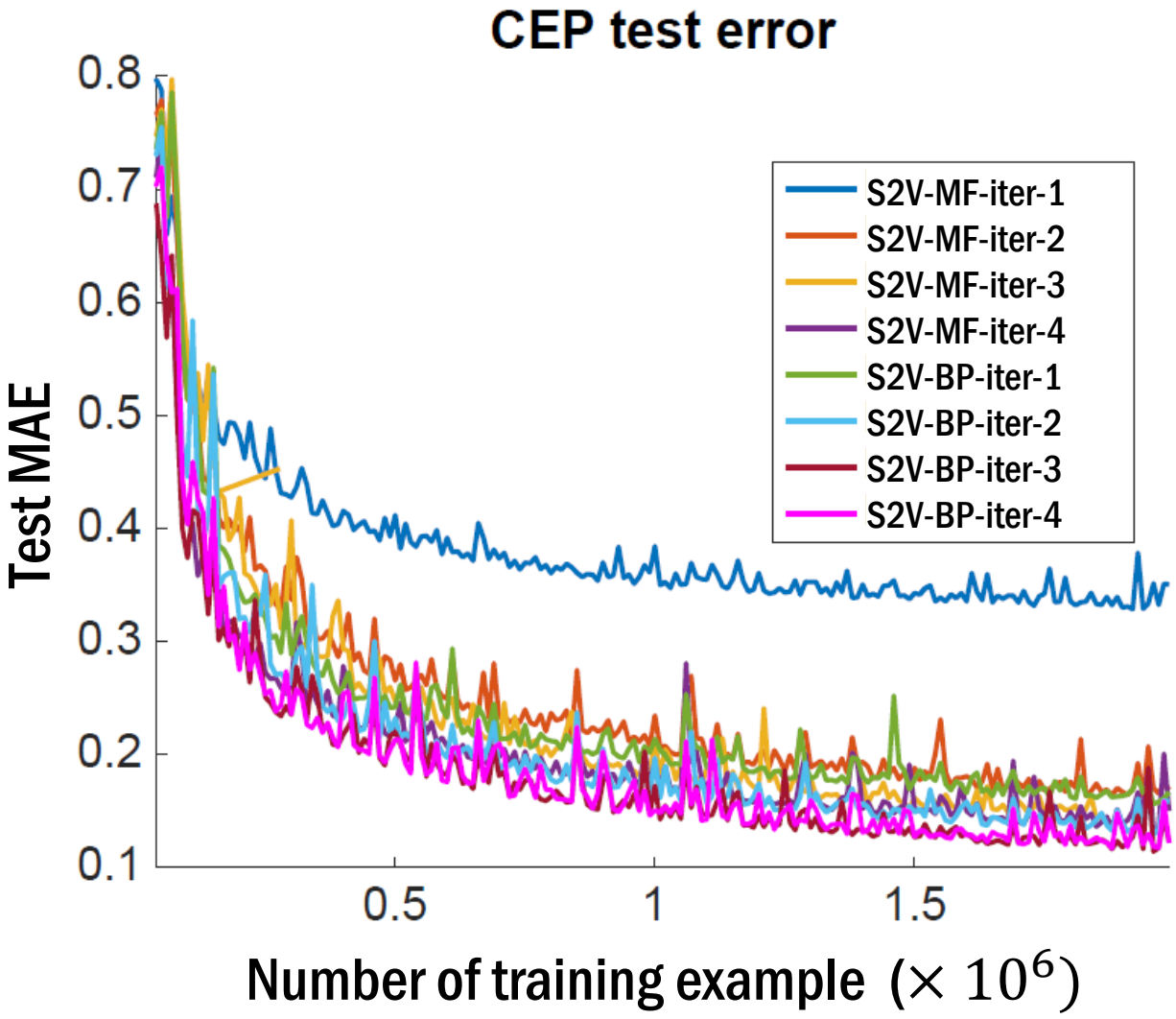
	Test MAE	Test RMSE	# parameters
Mean predictor	1.986	2.406	1
WL level-3	0.143	0.204	1.6 m
WL level-6	0.096	0.137	1378 m
S2V-MF	0.091	0.125	0.1 m
S2V-BP	0.085	0.117	0.1 m

We get ~4% relative error
with 10,000 times smaller model!

[Dai et al. 16]

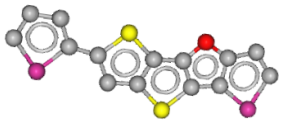


Effects of inference iterations on errors

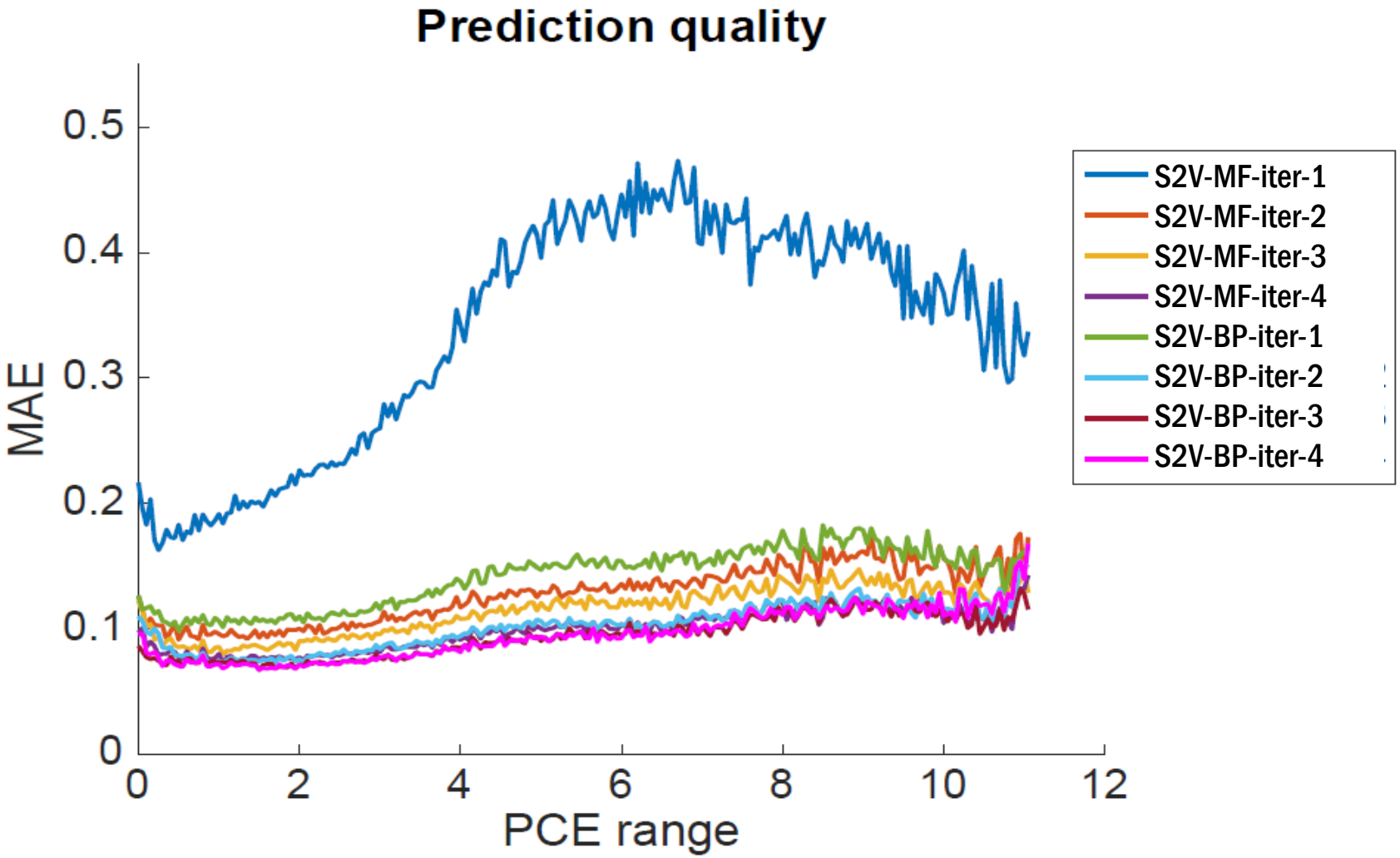


more inference interactions
⇒
better features

[Dai et al. 16]

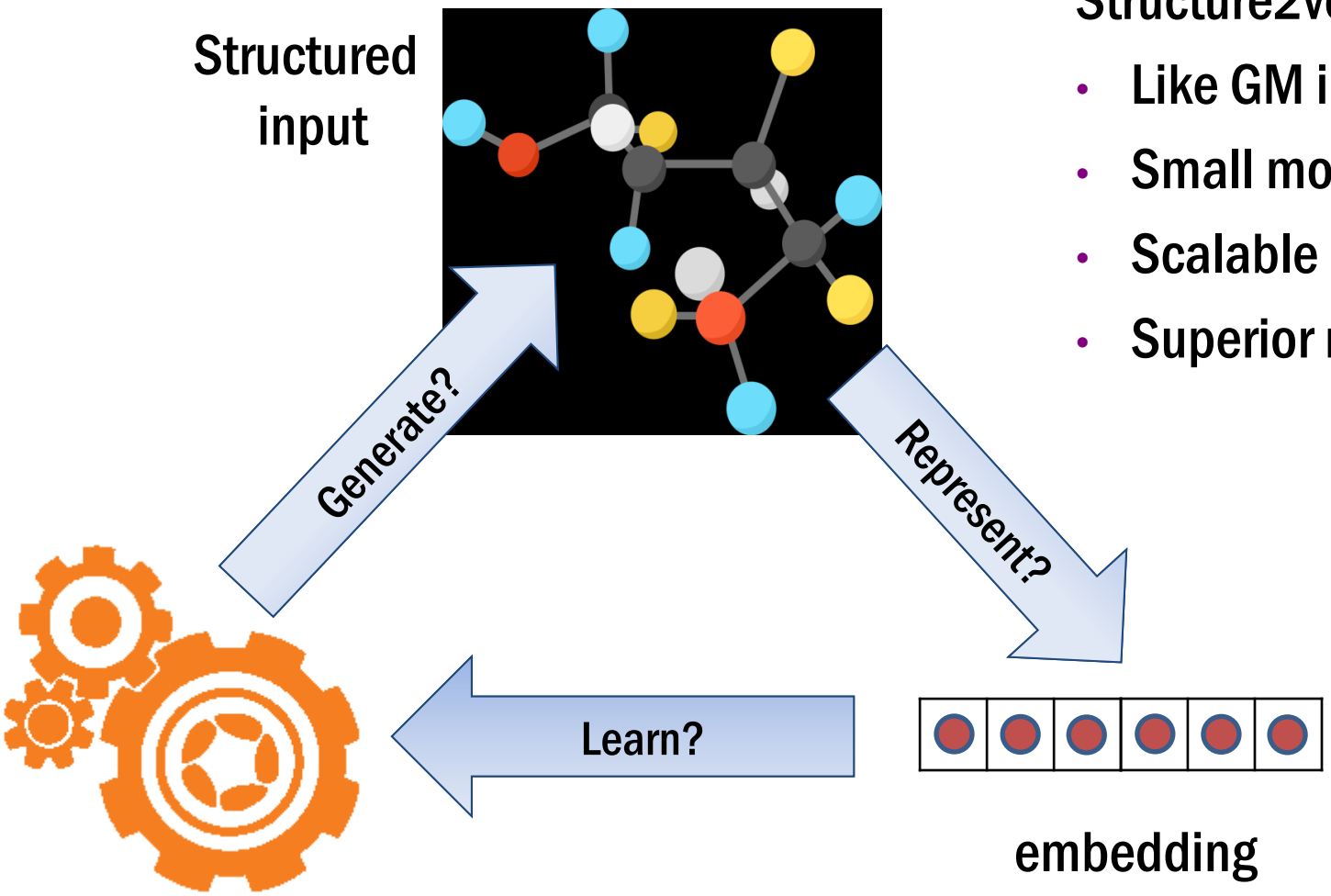


Errors across different output range



[Dai et al. 16]

Conclusion and future work



Structure2vec:

- Like GM inference
- Small model
- Scalable
- Superior results

Discriminative/generative models

Codes: https://github.com/Hanjun-Dai/graphnn