Discriminative Embedding of Latent Variable Models for Structured Data

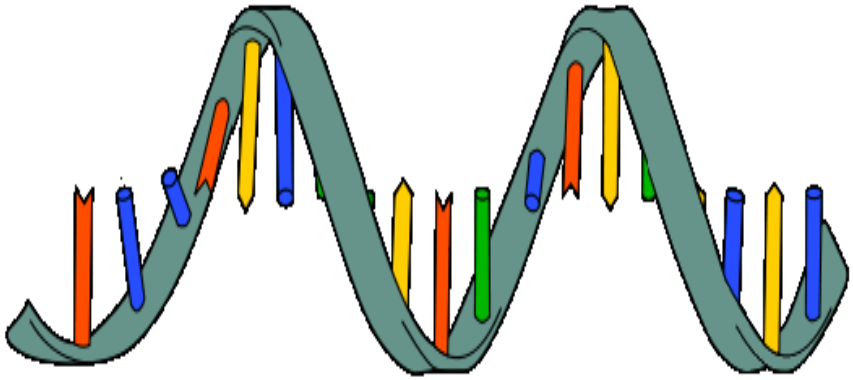
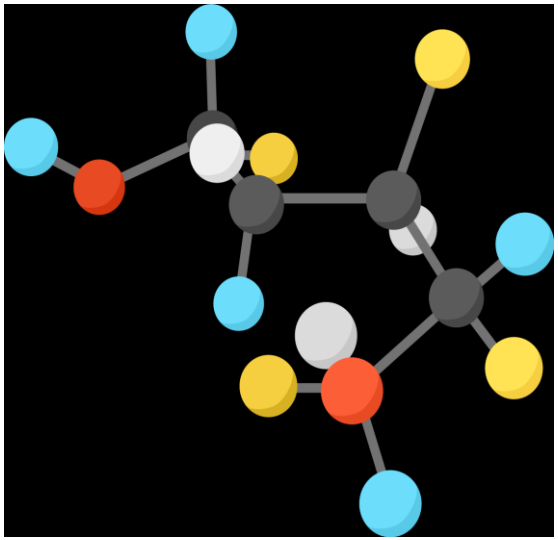
Le Song

Joint work with Hanjun Dai, Bo Dai

College of Computing

Georgia Institute of Technology

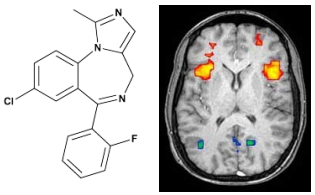
What are structured data?

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

Kernel methods

A modular framework to handle all kinds of complex data

Theoretical property well understood



ACAAGAT
GCCATTG
TCCCCCG

Kernel function

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

$m \times m$

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]

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

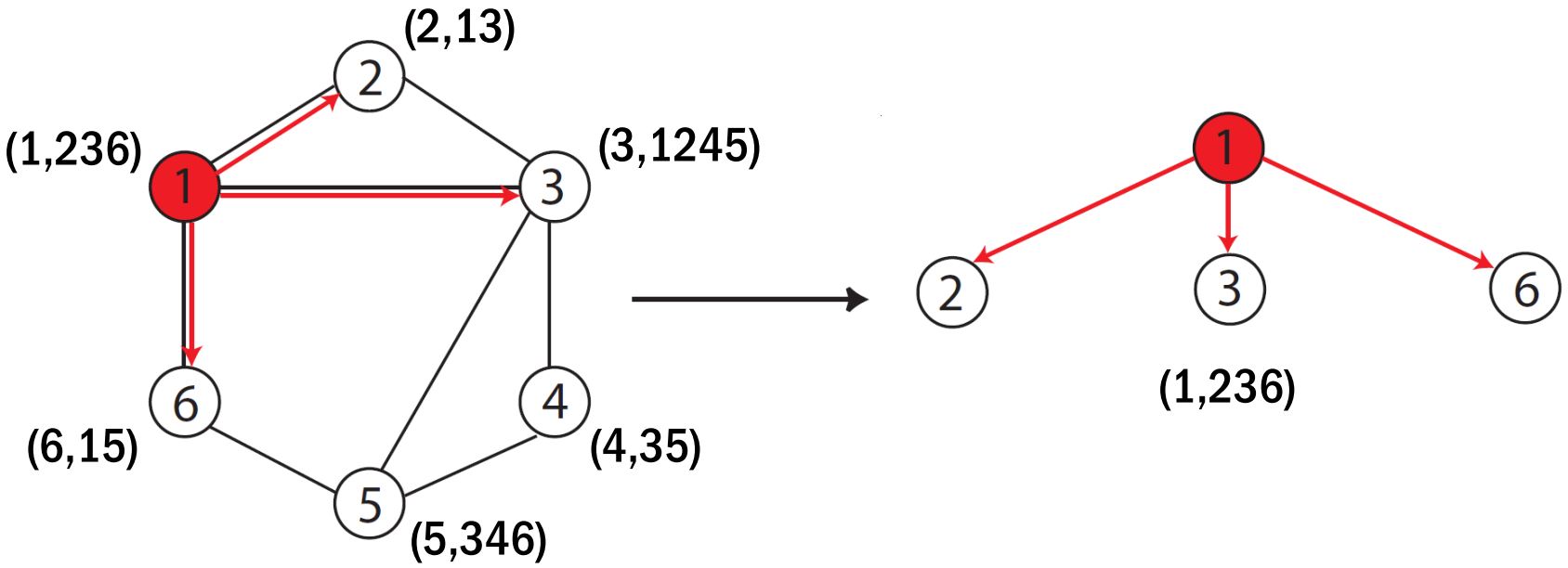
χ': C U U A C U U

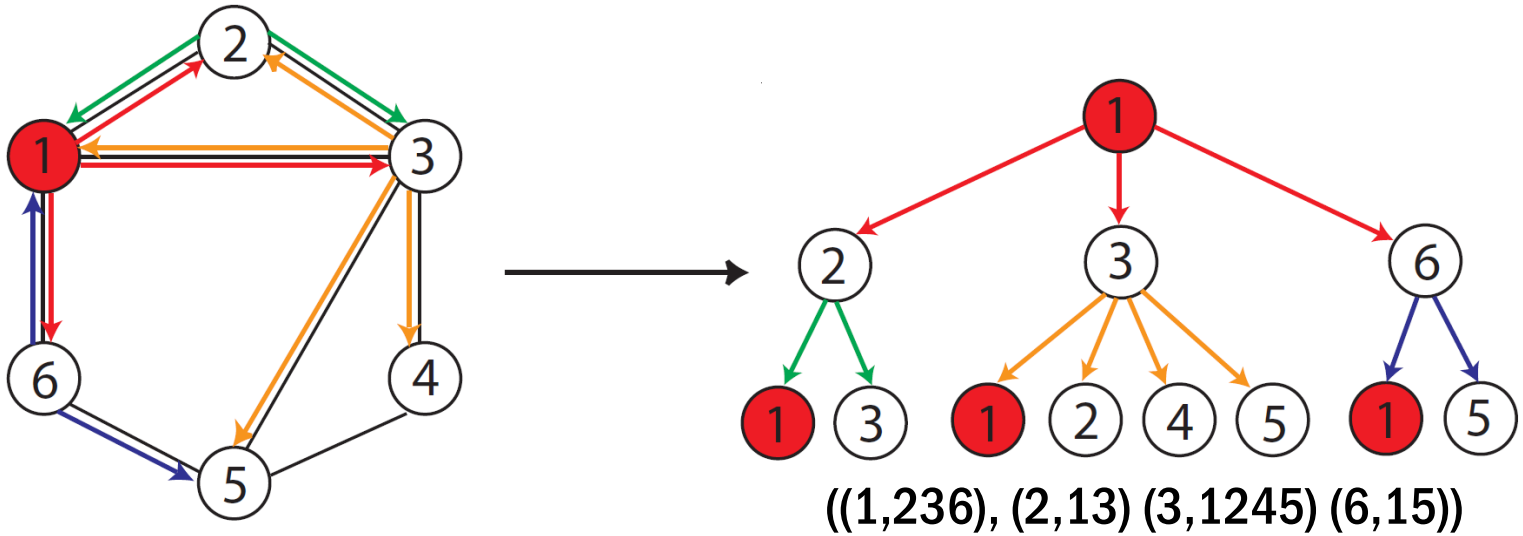
$$\phi(\chi) = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{bmatrix} \begin{matrix} \text{CUU} \\ \text{UUC} \\ \vdots \\ \text{CAG} \\ \text{AGU} \end{matrix} \qquad \phi(\chi') = \begin{bmatrix} 2 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix} \begin{matrix} \text{CUU} \\ \text{UUC} \\ \vdots \\ \text{CAG}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

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

[Leslie et al. 04]

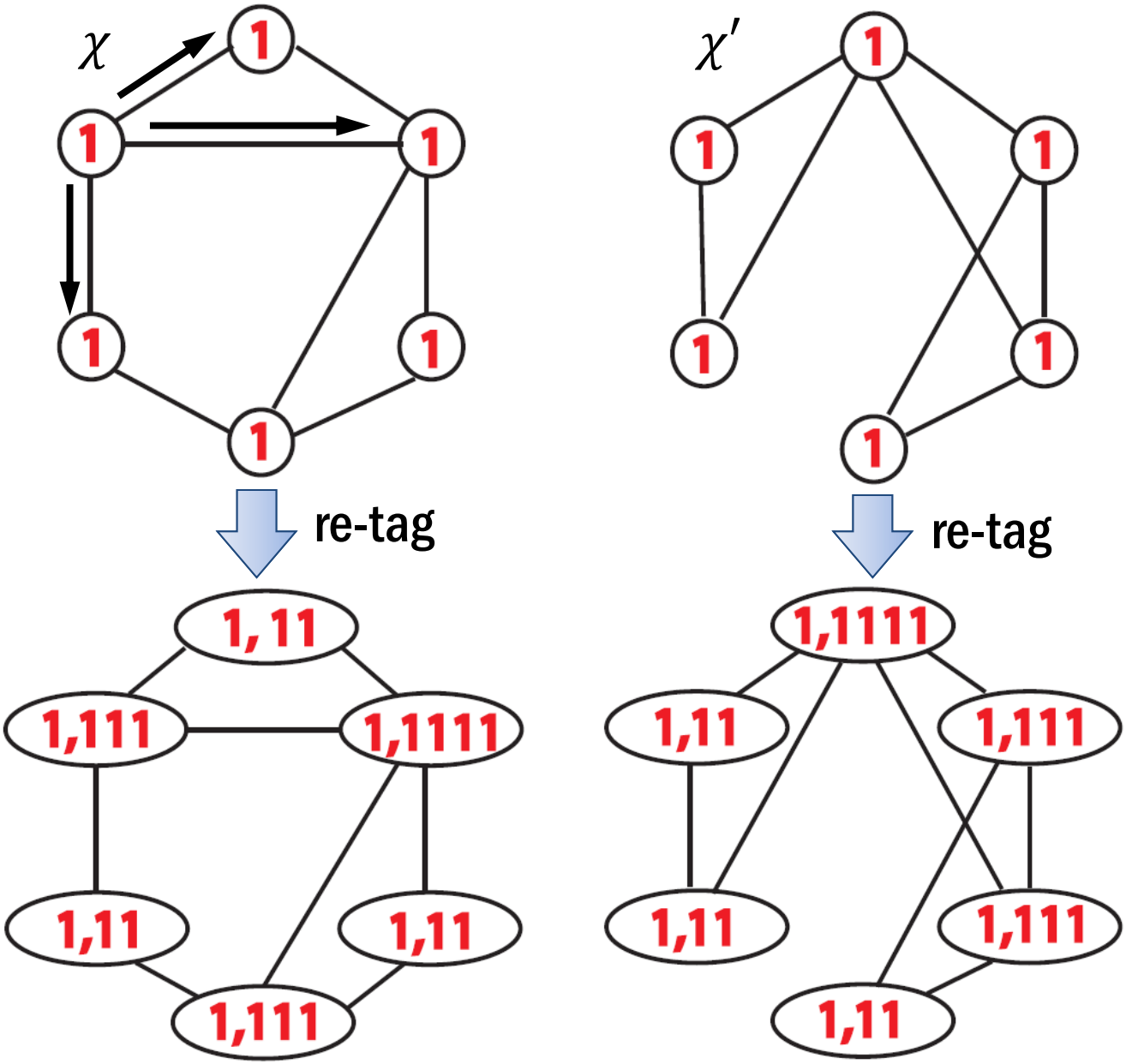
Bag of subtrees for graph



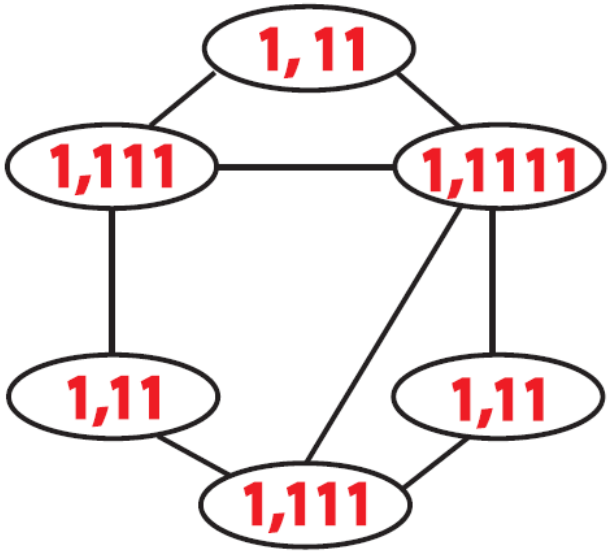


[Shervashidze et al. 11]

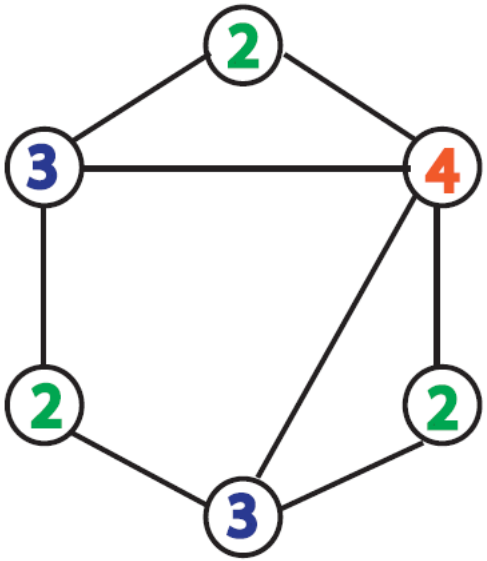
1. Re-tag nodes

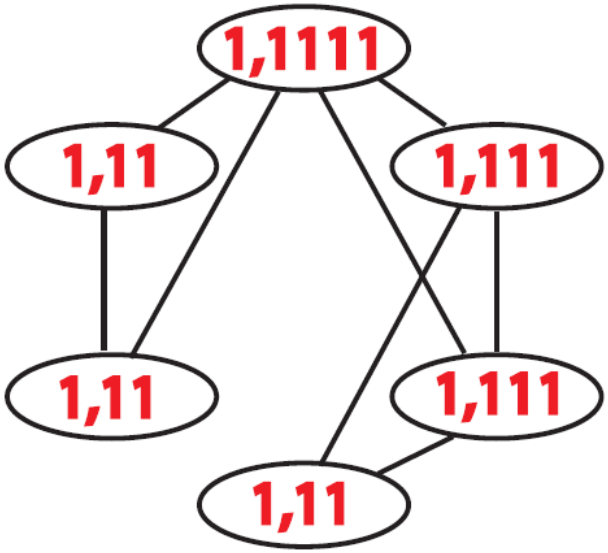


2. Simplify

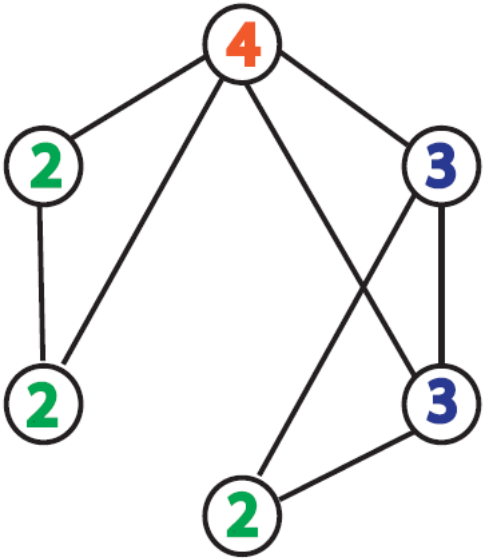


simplify





simplify

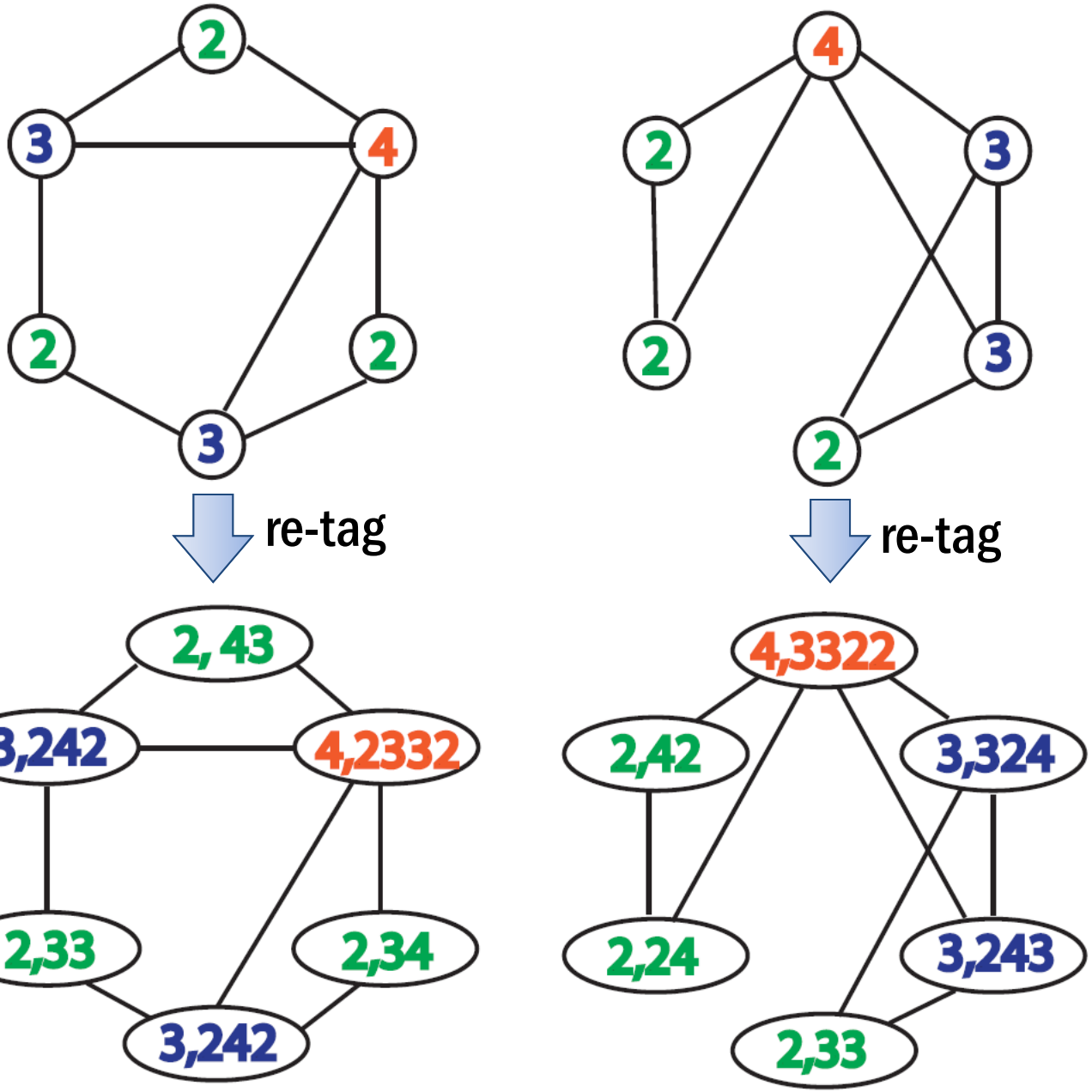


1, 11 → 2

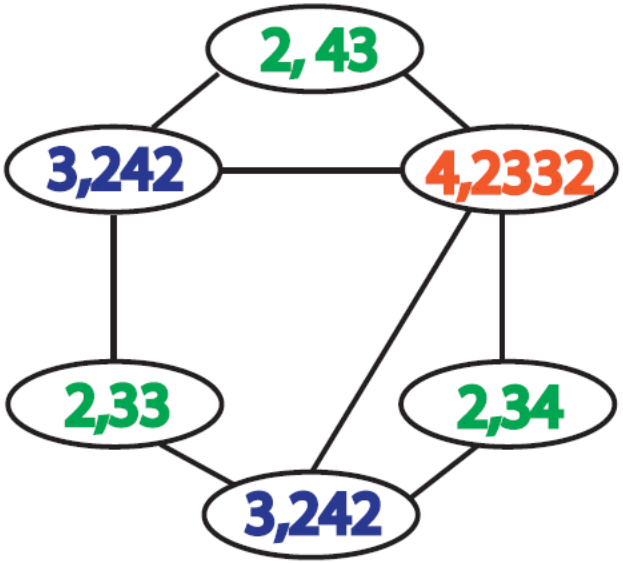
1,111 → 3

1,1111 → 4

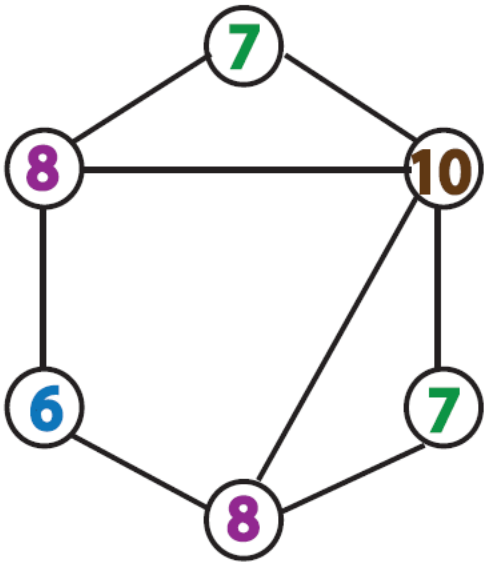
3. Re-tag again

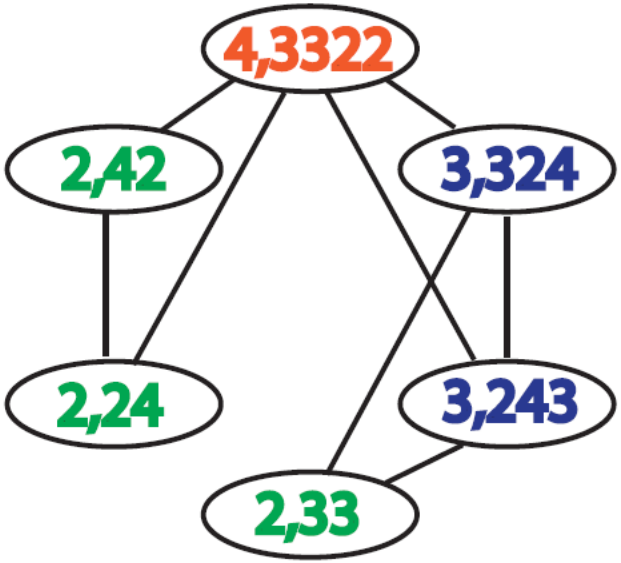


4. Simplify again

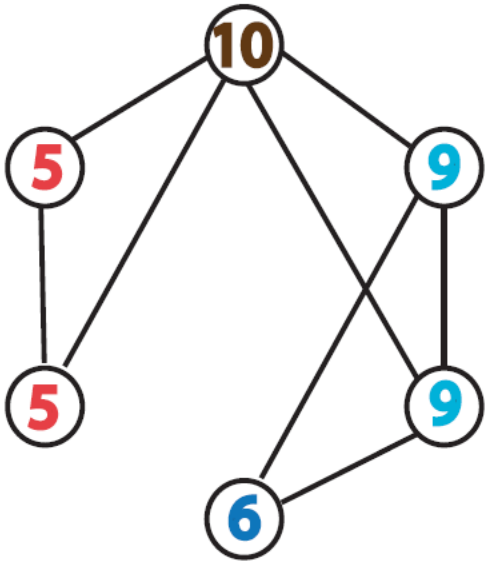


simplify





simplify



2,24 → 5

2,33 → 6

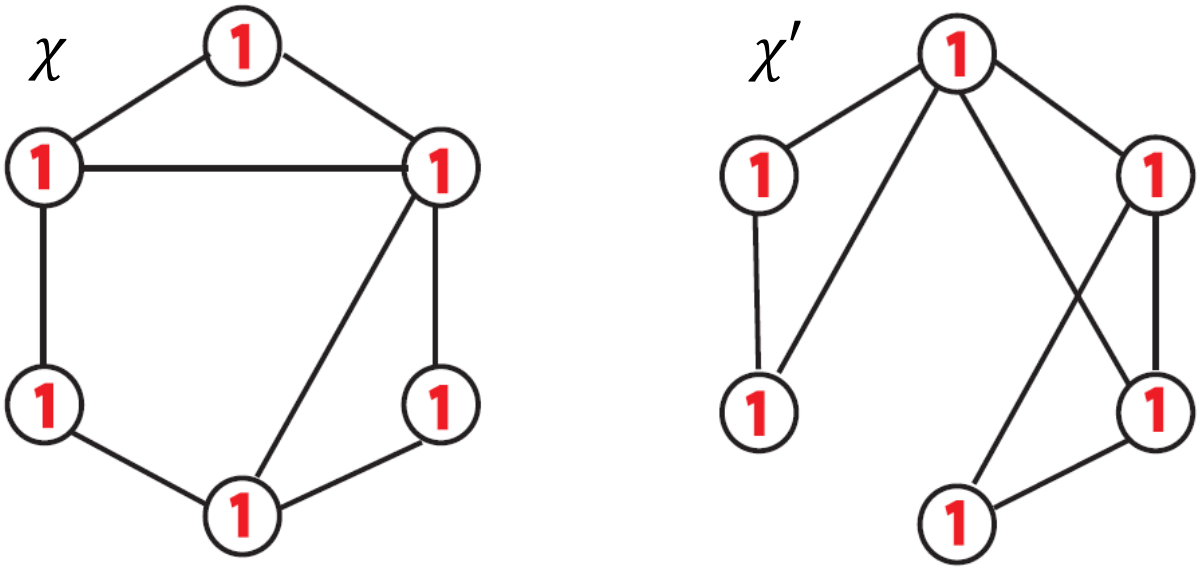
2,34 → 7

3,224 → 8

3,234 → 9

4,2233 → 10

Final features



1 2 3 4 5 6 7 8 9 10 … … … …

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

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

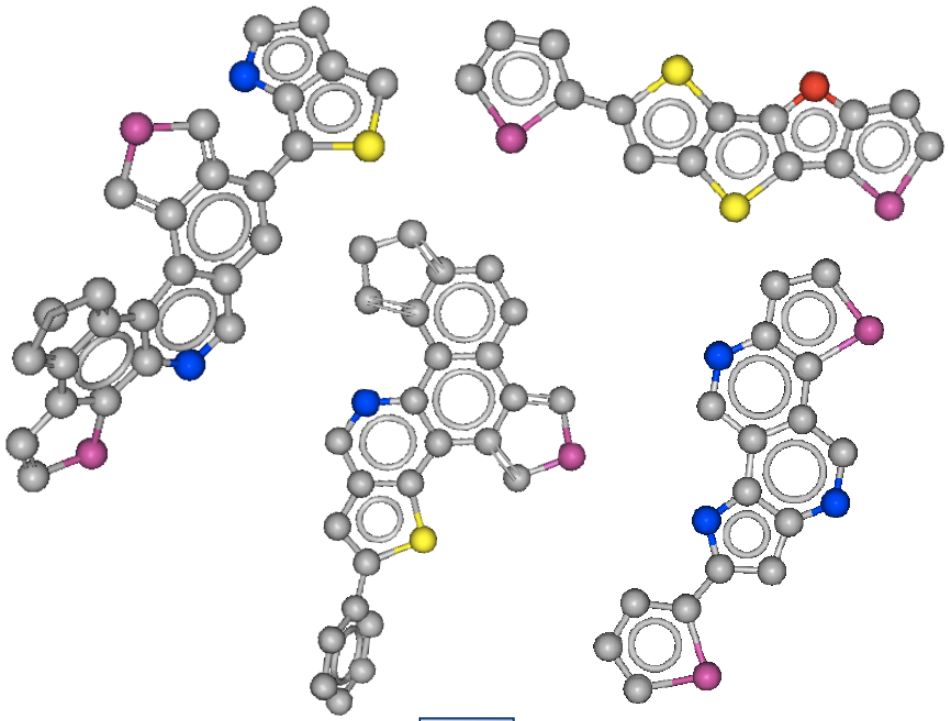
Level 0 feature

Level 1 feature

Level 2 feature

Higher level features

Explosive feature space



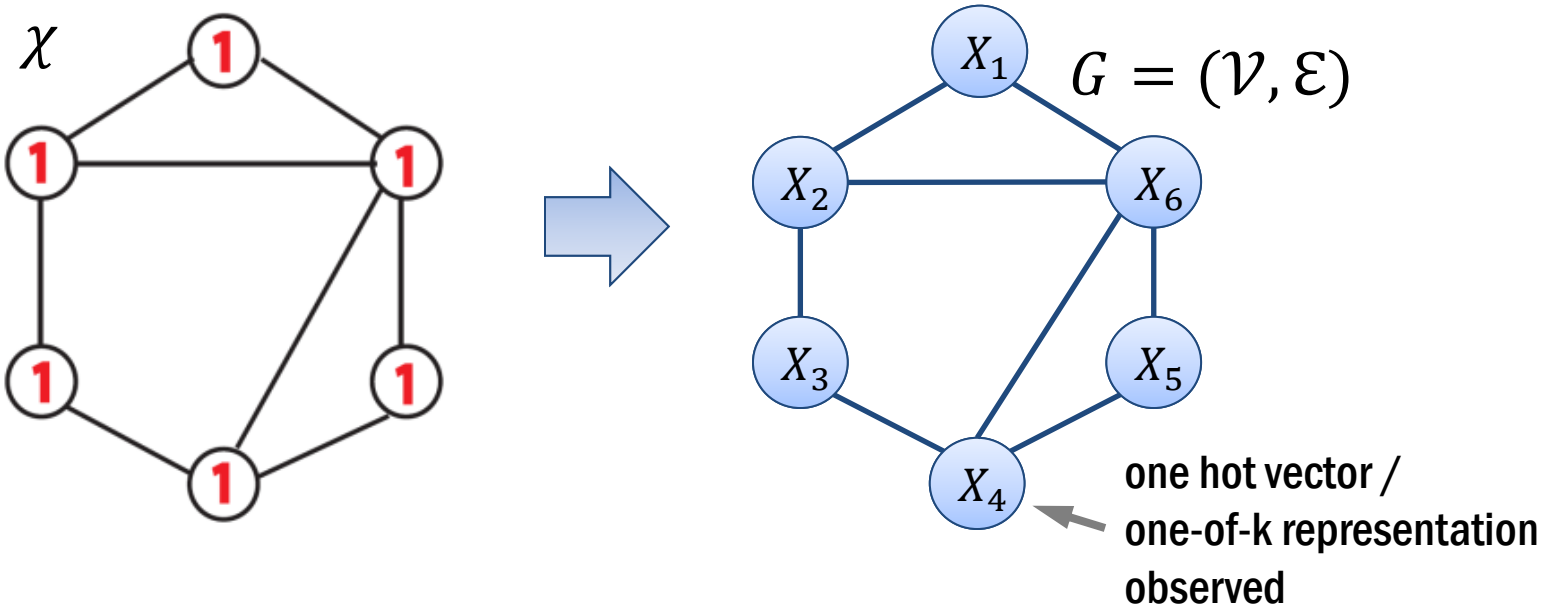
predict

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

Dataset	Harvard clean energy project
Size	2.3 million
Type	Molecule
Tag #	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)



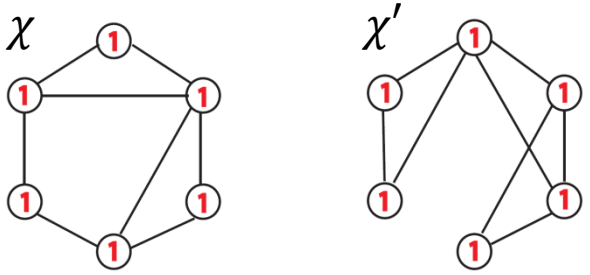
Define conditional independence structure using structure of the data

Model each data point as a Markov random field

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

[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
Parameter sharing?	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 $k(\chi, \chi')$	$U_\chi^\top I^{-1} U_{\chi'}$ $U_\chi = \nabla \log p(\chi \mid \theta^*)$ $I = \mathbb{E}\left[U_\chi U_\chi^\top\right]$	$\int p(\chi \mid \theta_\chi)^{1/2} p(\chi' \mid \theta_{\chi'})^{1/2}$

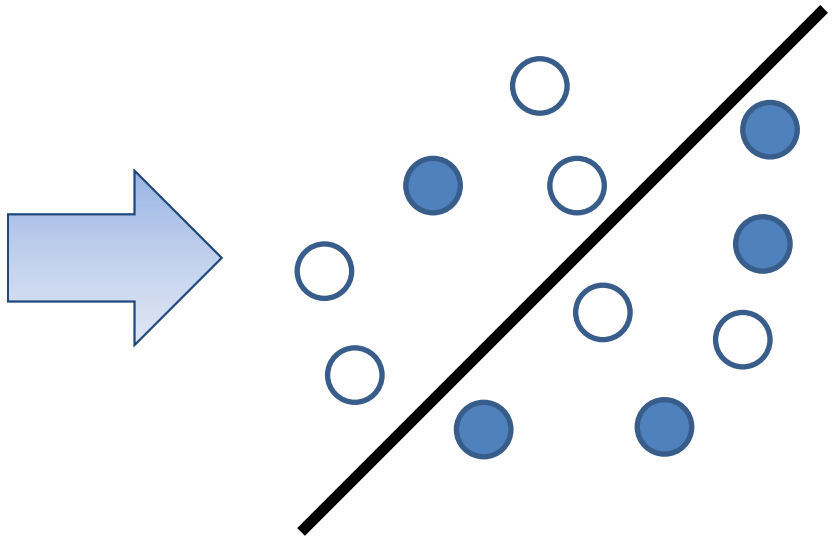
[Jaakkola et al. 99] [Jebara et al. 04]

Problem with existing two-stage approach

Stage 1	Stage 2

Construct kernel matrix

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



Or compute explicit features

1 2 3 4 5 6 7 8 9 10

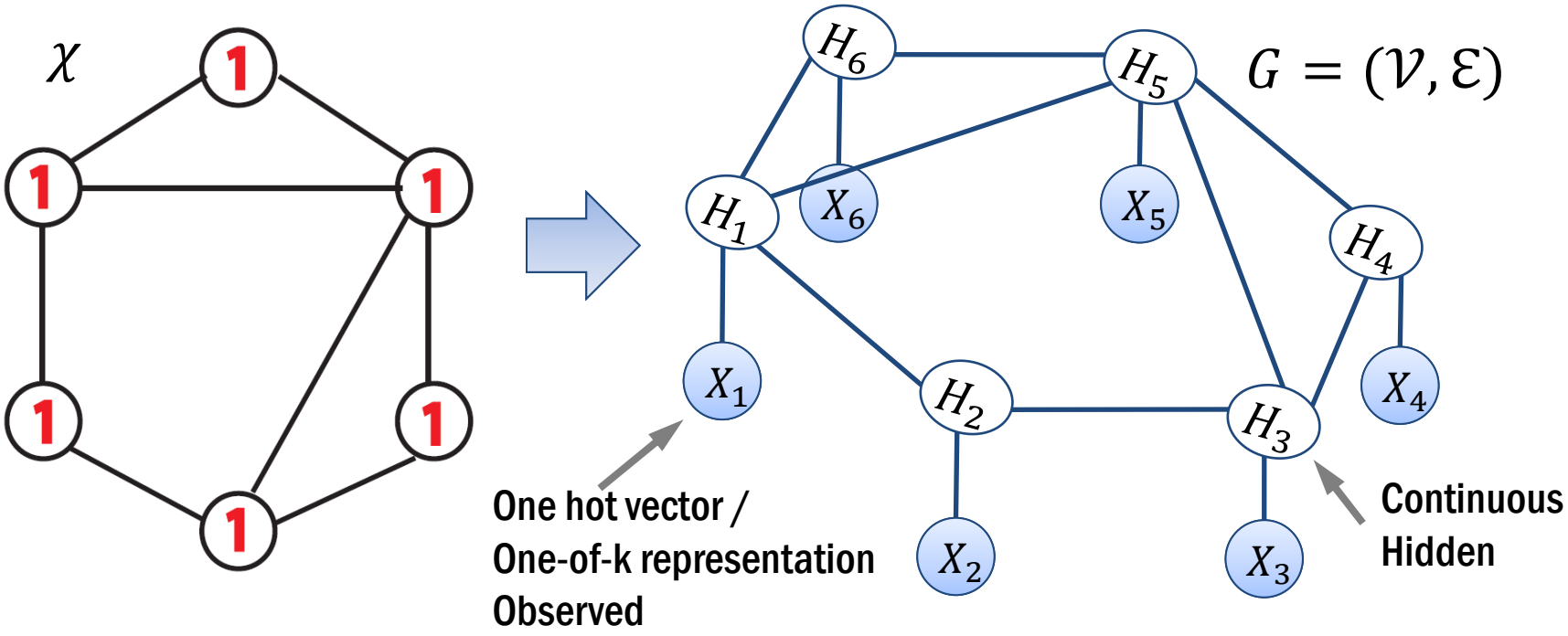
$$\phi(\chi) = (6, 3, 2, 1, 0, 1, 2, 2, 0, 1)^{\mathsf{T}}$$

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

Comparison

Methods	Feature learning?	Joint learning of features and classifier?
BOS kernel		
GM kernel		
Embedding (our)		

Represent data point as latent variable model

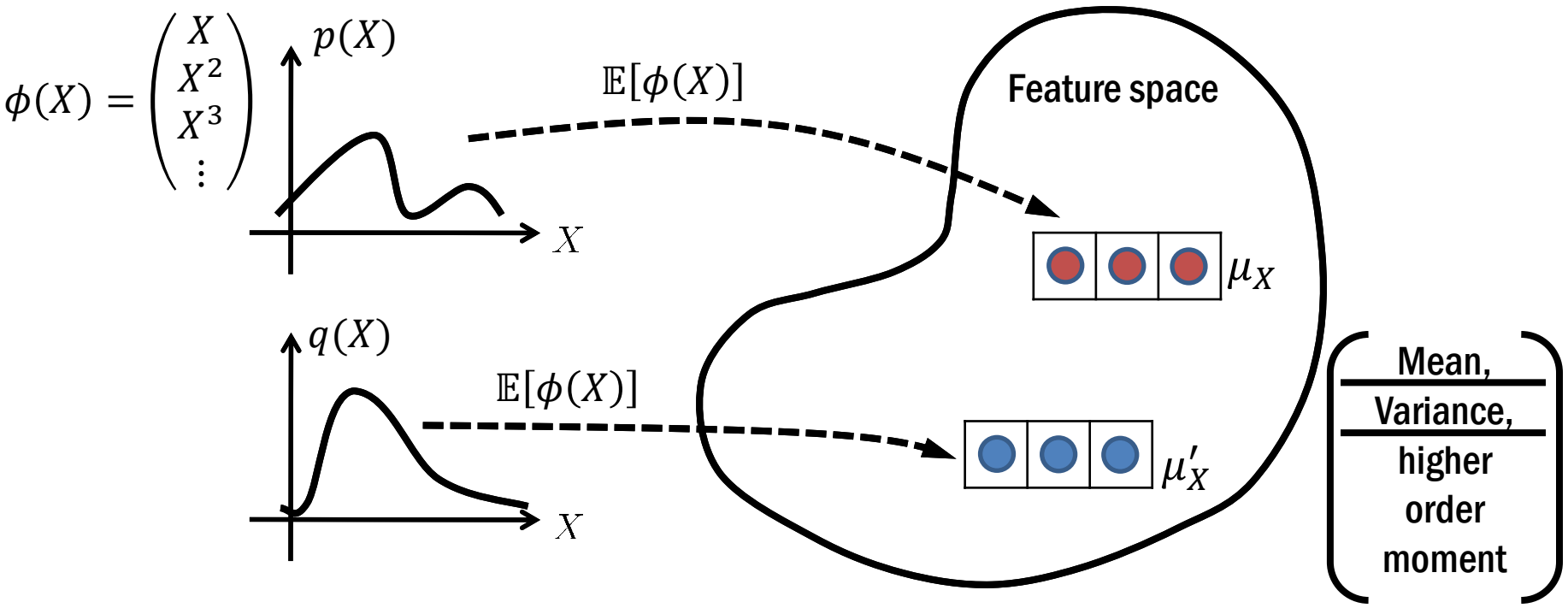


Model each data point as a Markov random field with latent variables

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

[Dai et al. 16]

Hilbert (feature) space embedding of distribution



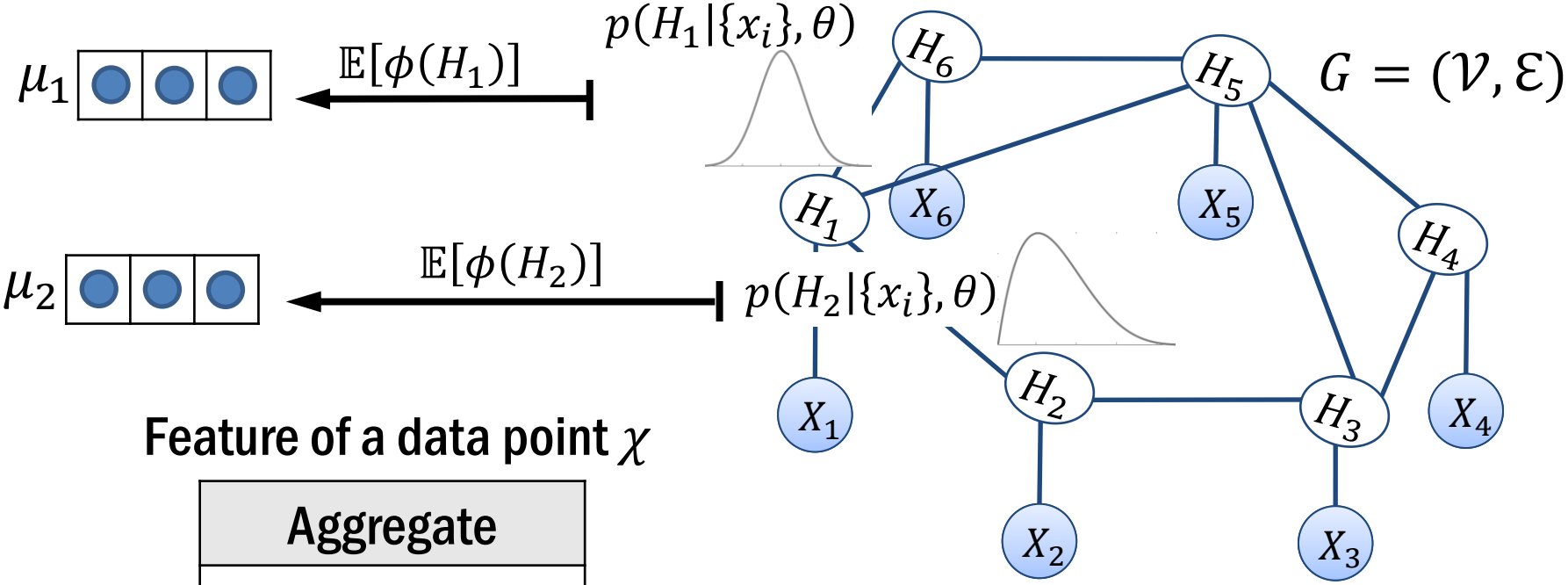
Rich representation: $p(x) = q(x)$ **iff.** $\|\mu_X - \mu'_X\|^2$

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

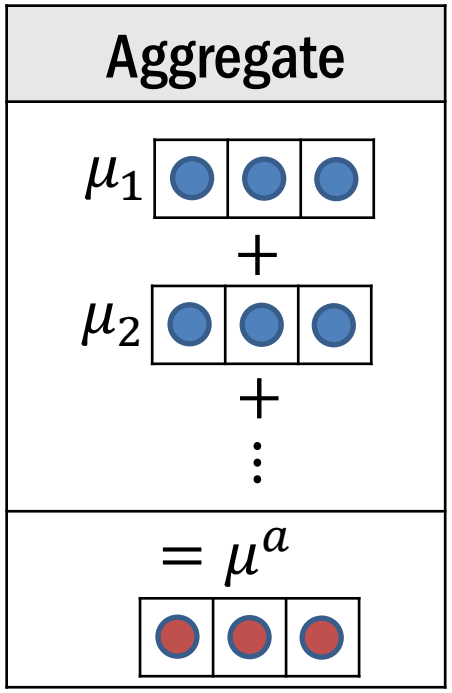
Operator $\mathcal{T}: \mathcal{P} \mapsto \mathcal{H}$
$\boldsymbol{\mathcal{T}} \circ p(x) = \widetilde{\boldsymbol{\mathcal{T}}} \circ \mu_X$

[Smola, Gretton, Song and Scholkopf. 2007]

Posterior embedding as features

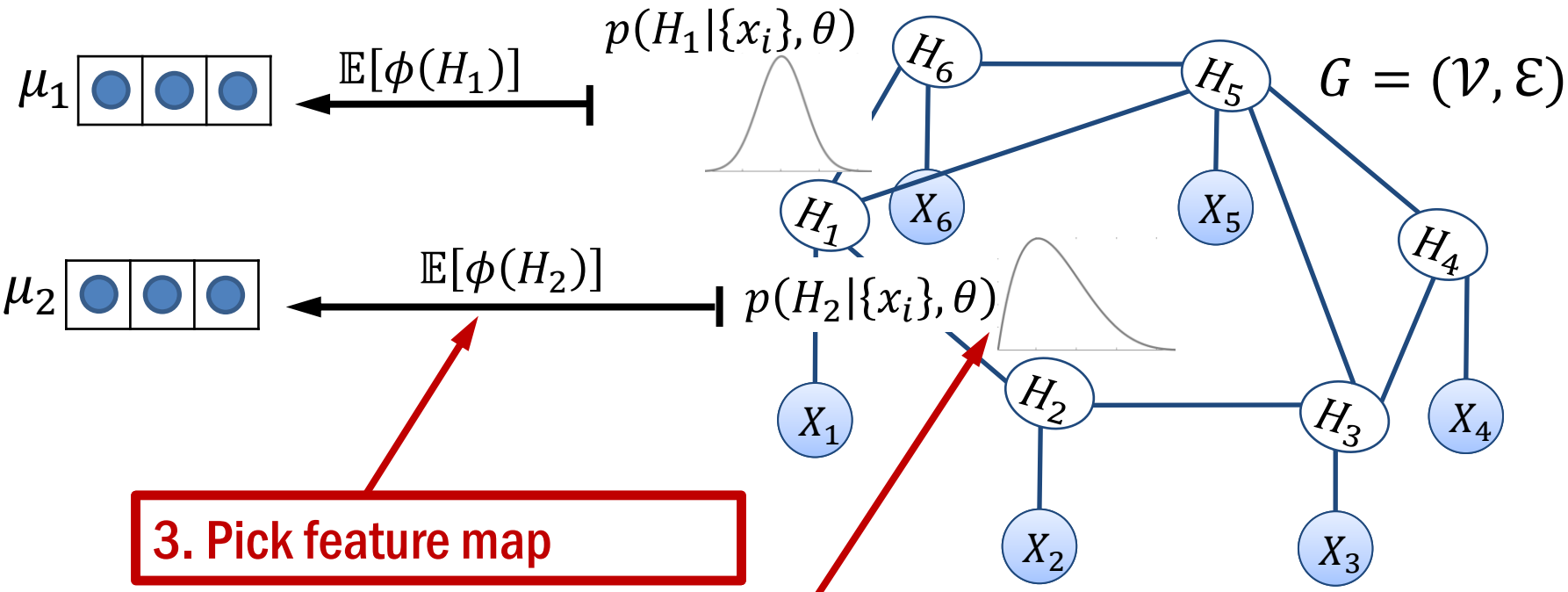


Feature of a data point χ



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

Posterior embedding as features



3. Pick feature map

2. inference

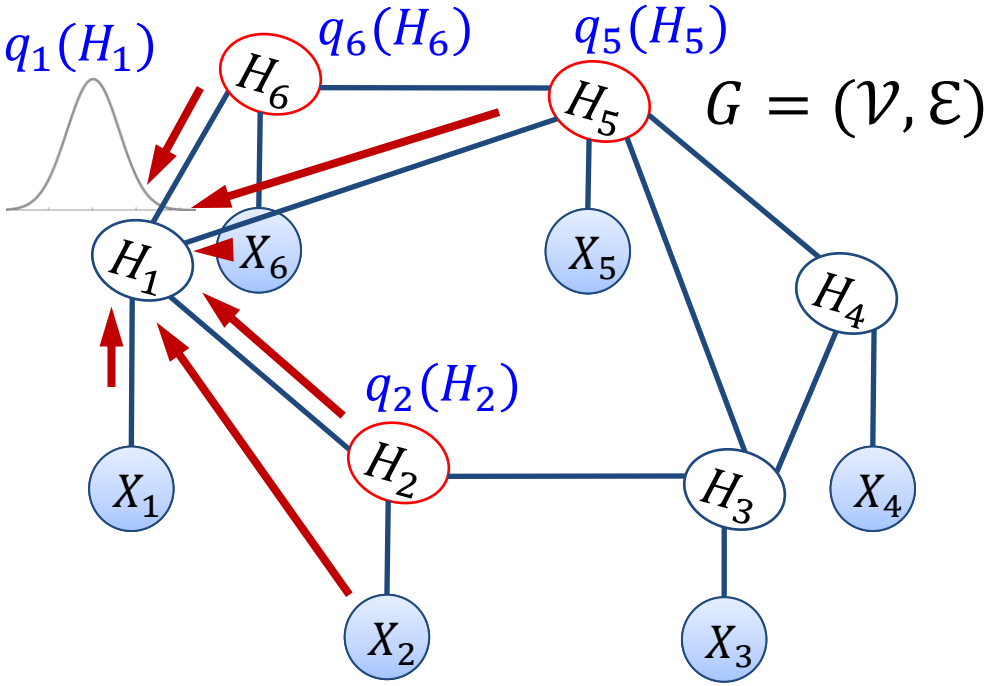
1. learn parameters

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

Mean field inference

Approximate joint posterior

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



Iterate till convergence:

For each i,

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

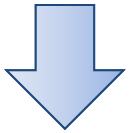
Operator view

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

Mean field update using embeddings

Operator view

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



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

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

Embedding μ_X is a sufficient statistic of $p(X)$

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

Parameterize mean field update

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

Assume $\mu_i \in \mathcal{R}^d$, parametrize as a single nonlinear mapping

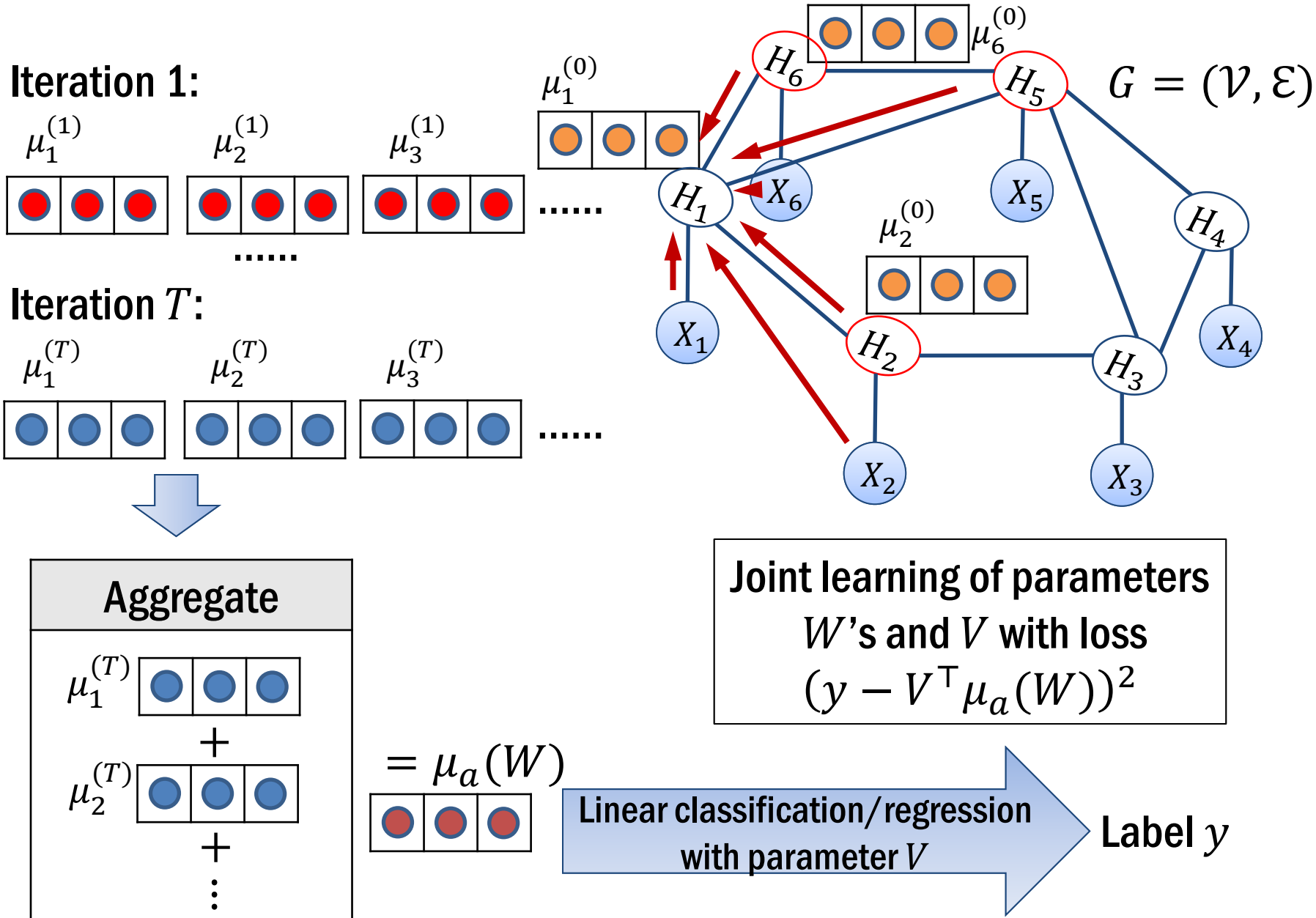
$$\mu_i \leftarrow \sigma\left(W_1 x_i + W_2 \sum_{j \in \mathcal{N}(i)} x_j + W_3 \sum_{j \in \mathcal{N}(i)} \mu_j\right)$$

$Max\{0,\cdot\}$

Variables	Details
x_i	E.g., m dim. one hot vector of atomic number
W_1	$d \times m$ matrix
W_2	$d \times m$ matrix
W_3	$d \times d$ matrix

Learn with supervision

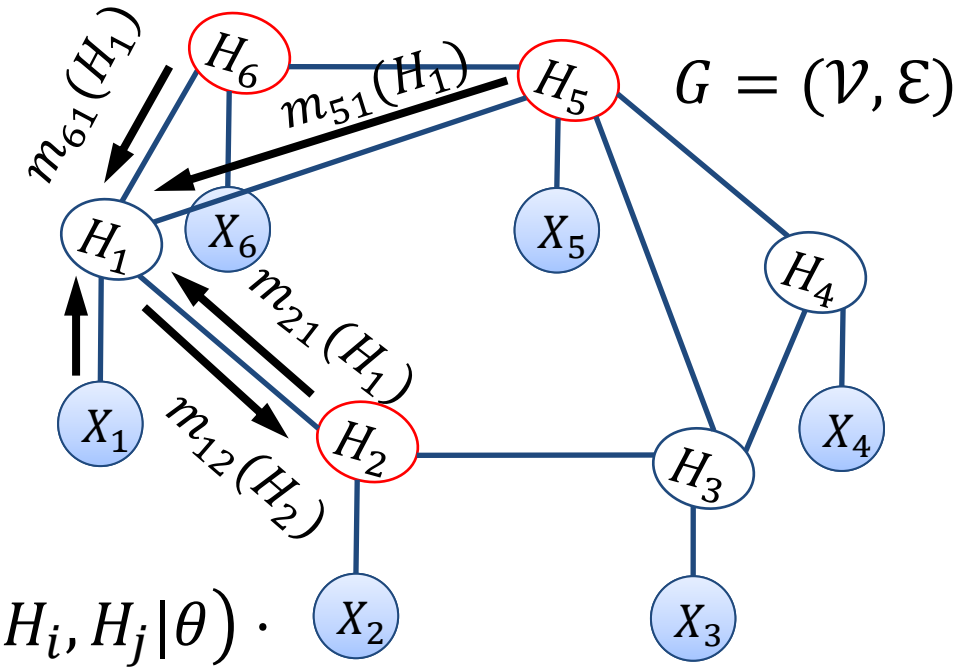
Mean field iterations as nonlinear mappings



Inference with belief propagation

Approximate $p(H_i|\{x_j\},\theta)$ 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:

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

Operator view

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

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

Belief propagation using embeddings

Operator view

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

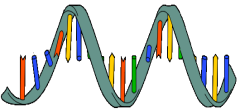
$$m_{ij}(H_j) \leftarrow \boldsymbol{\mathcal{T}}'(\theta) \circ \left(x_i, \{m_{\ell i}(H_i)\}_{\ell \in \mathcal{N}(i) \setminus \mathrm{j}} \right)$$

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

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

Embedding μ_X is a sufficient statistic of $p(X)$

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



Experiment I: sequence binary classification

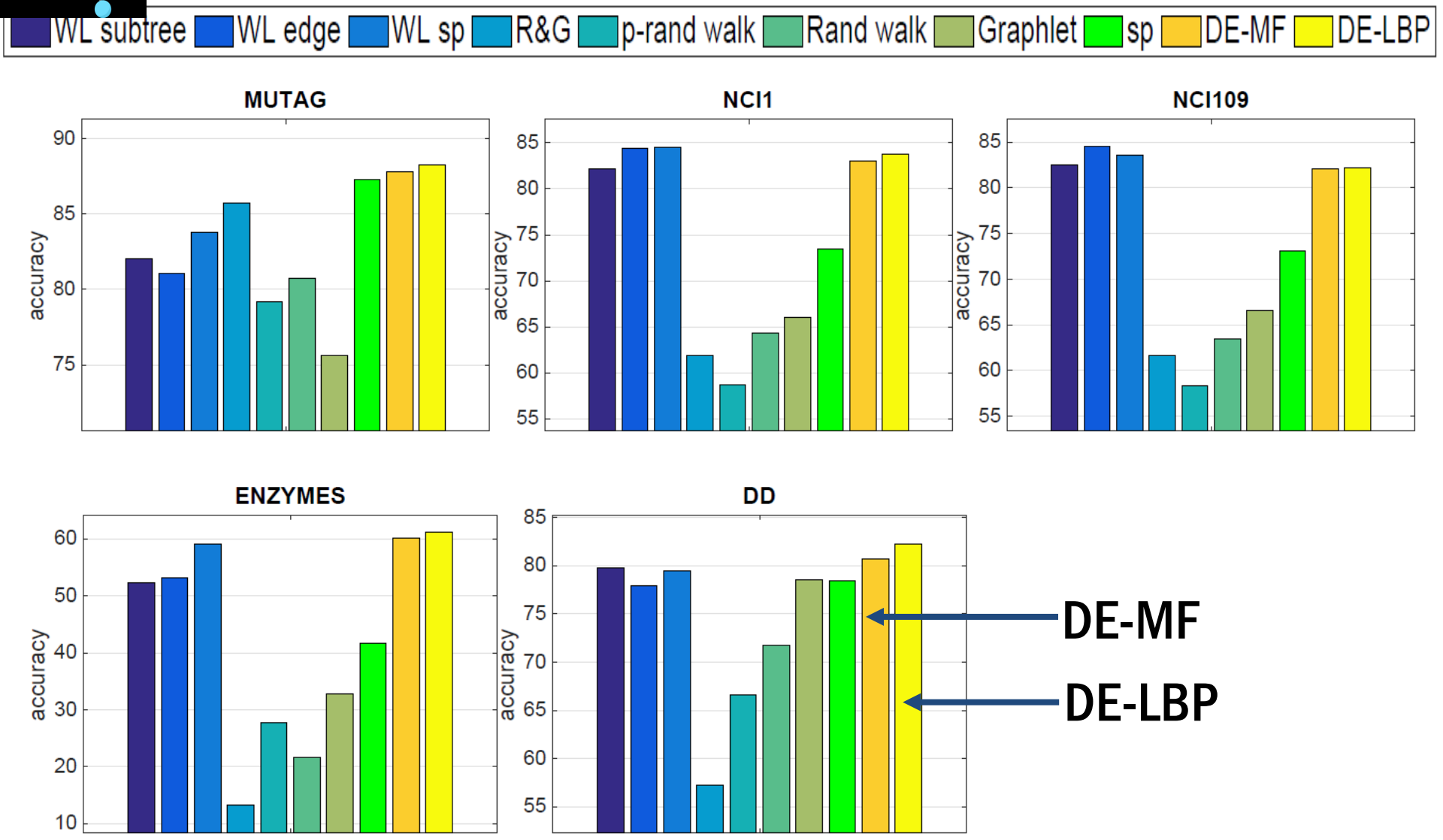
dataset	type	#sample	Alphabet size	Avg sequence length
CRISPR/Cas9	RNA	5310	4	30
SCOP	Protein	7329	26	100

	CRISPR/Cas9	SCOP
kmer-single	0.76±0.02	0.71±0.05
kmer-concat	0.76±0.02	0.85±0.05
mismatch	0.76±0.02	0.86±0.12
fisher	0.73±0.03	0.87±0.09
DE-Mean Field	0.77±0.02	0.91±0.07
DE-Loopy BP	0.77±0.02	0.92±0.06

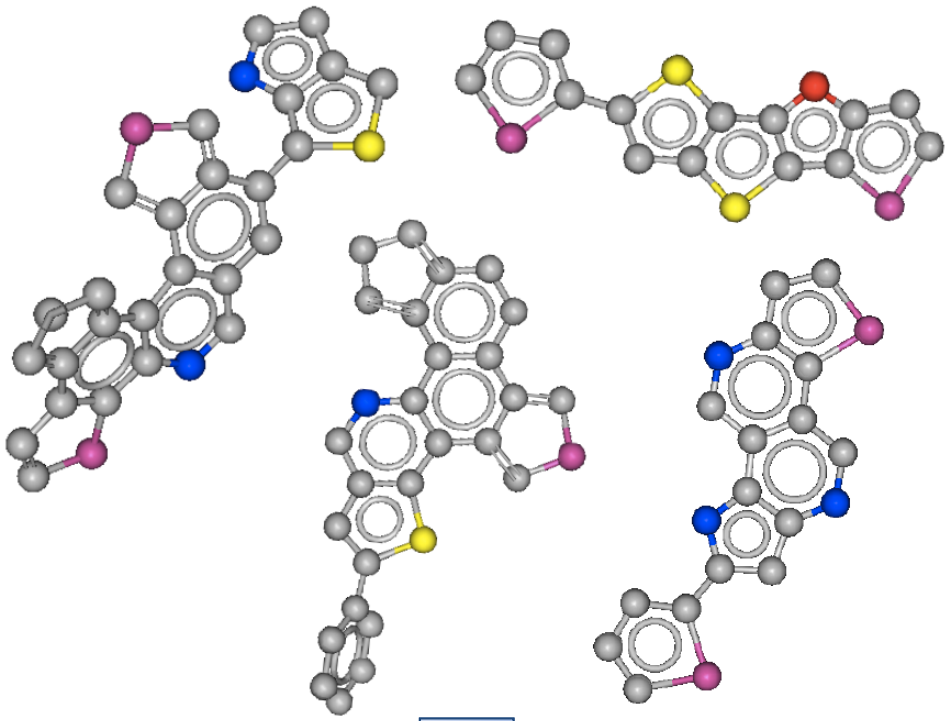
Experiment II: graph classification

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 II: graph classification



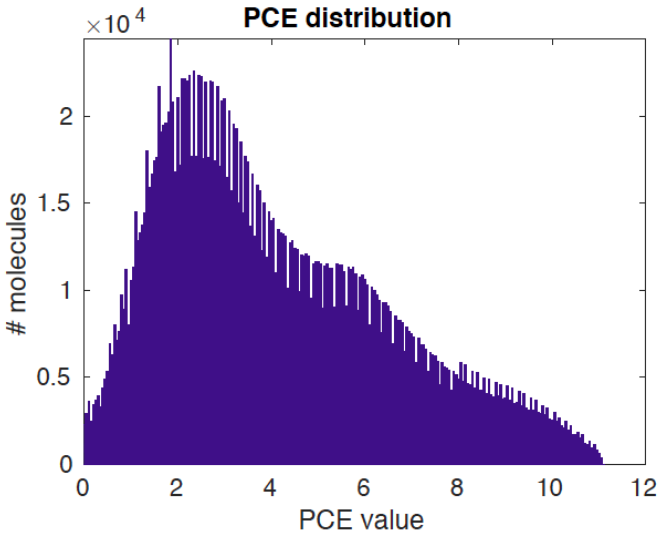
Predicting efficiency of solar panel materials

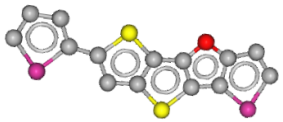


predict

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

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



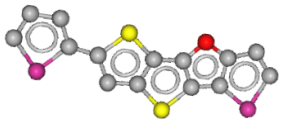


Regression result

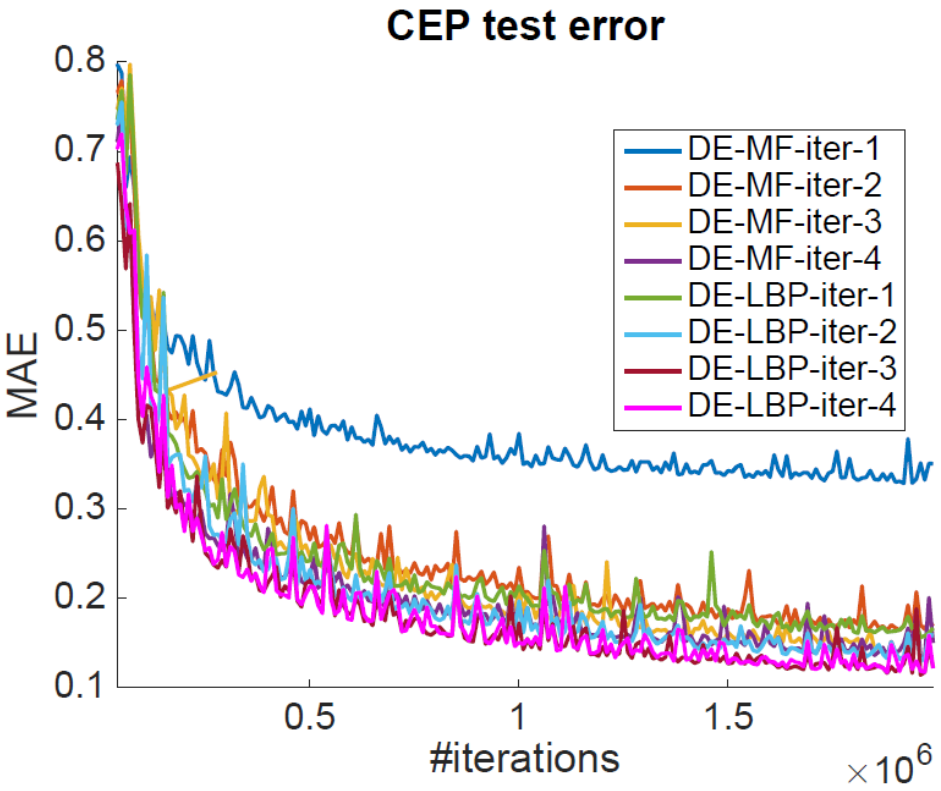
10% data for testing

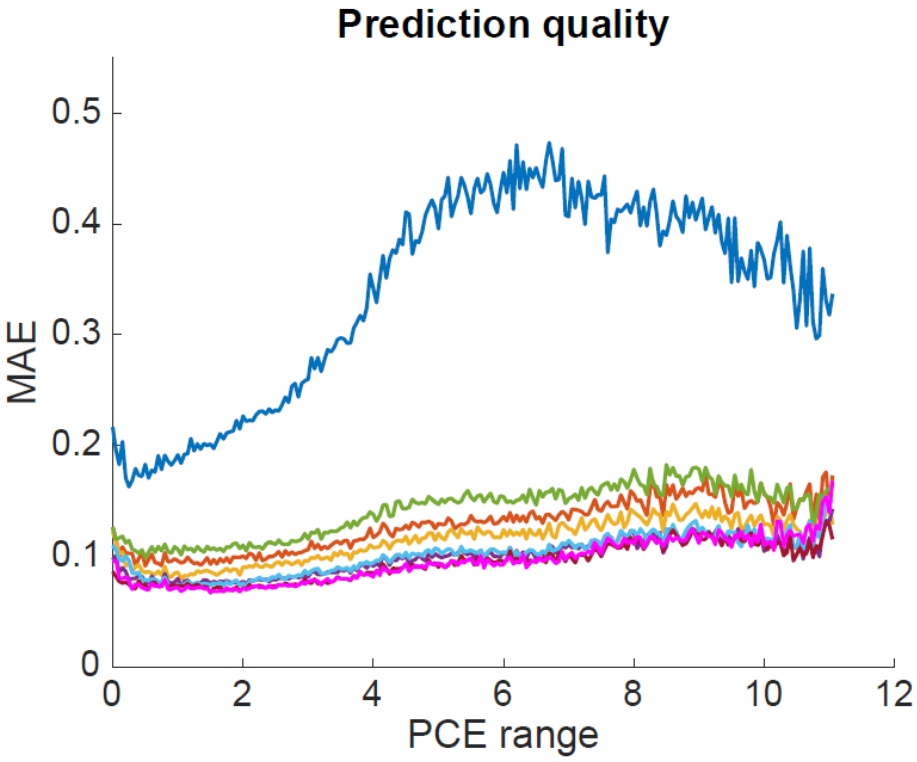
	Test MAE	Test RMSE	# parameters
Mean predictor	1.986	2.406	1
WL level-3	0.143	0.204	1.6 m
WL level-6	0.096	0.137	1378 m
DE-MF	0.091	0.125	0.1 m
DE-LBP	0.085	0.117	0.1 m

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



More on test errors





Test error
vs
Optimization iteration

Test error
vs
PCE value

Conclusion and future work

- Key components of the method
  - Extract features using graphical model inference
  - Graphical model inference $\Leftrightarrow$ a recursive sequence of nonlinear mappings
  - Learn this mapping end-to-end with downstream classification
- Pros: much smaller model, scalable to big data, state-of-the-art results
- Many other advanced graphical model inference unexplored
- Learning the behavior of algorithms on discrete structures
- Codes: https://github.com/Hanjun-Dai/graphnn