ISCSLP 2016 Tutorial

# Undirected Graphical Models: Theory and Applications to Speech and Language Processing

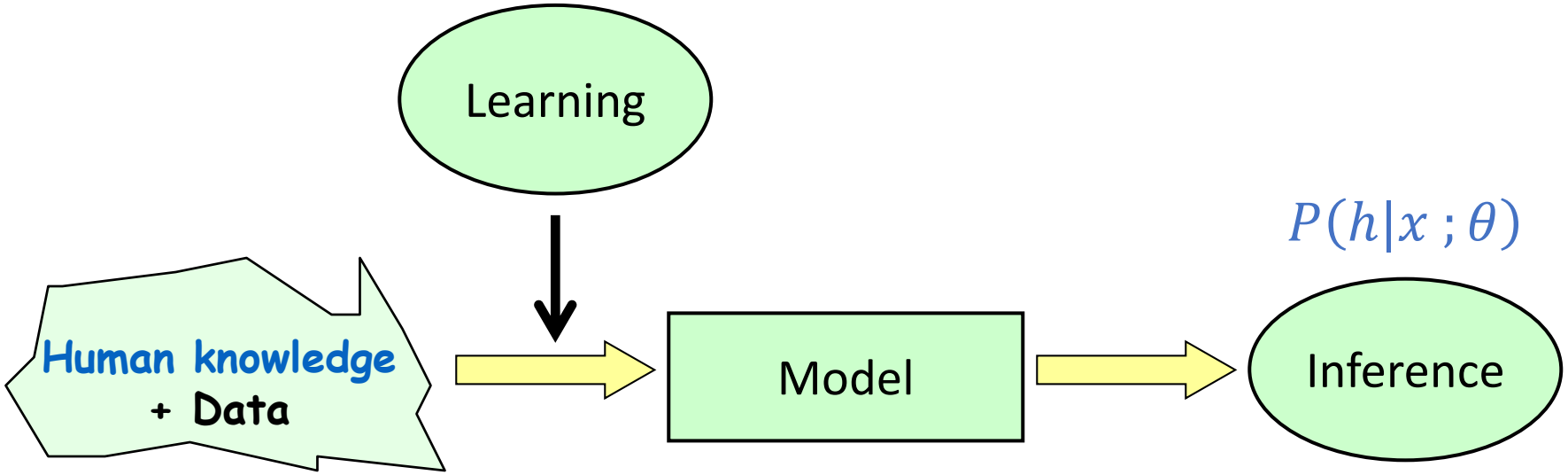
**Zhijian Ou**

Speech Processing and Machine Intelligence (SPMI) Lab

Tsinghua University

2016-10-17

# Introduction




$P(x, h\,;\,\theta)$ : Generative model, $e.g. HMMs$

$P(h|x\,;\theta)$ : Discriminative model, $e.g. CRFs, NNs$

# Introduction

- Probabilistic graphical models
  - A general framework for describing and applying statistical models
  - Statistical modeling, inference and learning
- Directed graphical models (DGMs)
  - aka Bayesian networks (BNs)
  - e.g. HMMs, Topic models (LDA)
- Undirected graphical models (UGMs)
  - aka Markov random fields (MRFs), random fields (RFs), Markov networks (MNs)
  - e.g. CRFs, RBMs, DBNs

This tutorial will

- Introduce the general and basic **concepts** of undirected graphical models,
- Demonstrate how to **apply** the theory to solve various problems through a number of case studies.

# Contents

# Graph theory basics

- A graph is a pair $g = (V, E)$
  - $V = \{ x_1, \ldots, x_N \}$ is a finite set of vertices, also called nodes, of $g$
  - $E$ is a subset of the set $V \times V = \{(x_i, x_j): i \neq j\}$, called edges of $g$

  - Undirected edge: both $(x_i, x_j)$ and $(x_j, x_i)$ belong to $E$

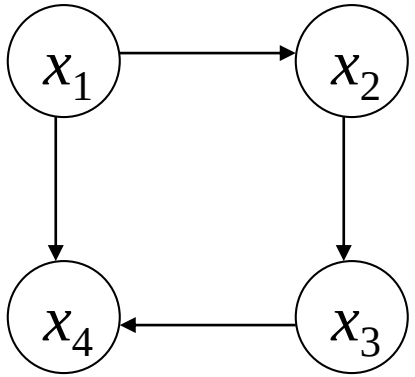
    $x_i \sim x_j$
  - Directed edge (arc): $(x_i, x_j) \in E$ and $(x_j, x_i) \notin E$

    $x_i \rightarrow x_j$

    we say that $x_i$ is a parent of $x_j$, $x_j$ is a child of $x_i$

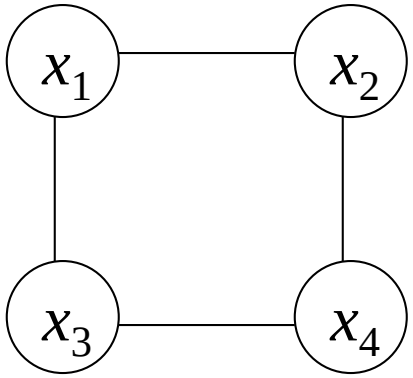
# Graph theory basics

- Directed graph: All edges in the graph are directed




$$V = \{ x_1, x_2 , x_3, x_4 \}$$
$$E = \{\{x_1, x_2\},$$
$$\{x_1, x_4\},$$
$$\{x_2, x_3\},$$
$$\{x_3, x_4\}\}$$

- Undirected graph: All edges in the graph are undirected




$$V = \{ x_1, x_2 , x_3, x_4 \}$$
$$E = \{\{x_1, x_2\}, \{x_2, x_1\},$$
$$\{x_1, x_3\}, \{x_3, x_1\},$$
$$\{x_2, x_4\}, \{x_4, x_2\},$$
$$\{x_3, x_4\}, \{x_4, x_3\}\}$$

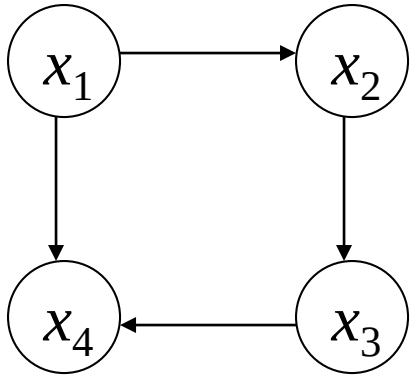
# Semantics of DGMs

- A graphical model is a family of probability distributions defined in terms of a directed or undirected graph.
- Semantics: how the family of distributions is defined.

Consider a directed acyclic graph (DAG) : $g = (V, E)$

$x_V$ : a collection of random variables indexed by the nodes

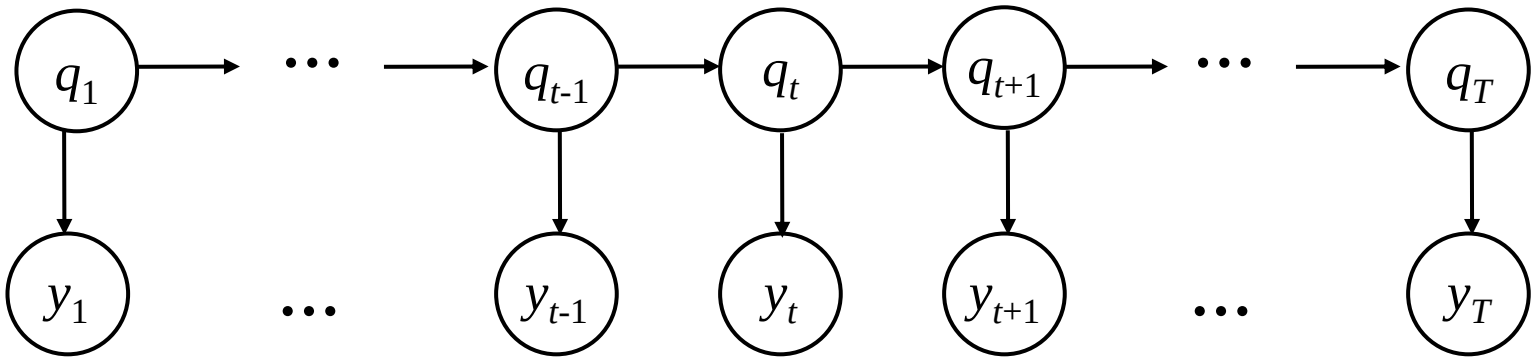
$pa(v)$ : the parent nodes of $v$




$$p(x_V) \triangleq \prod_{v \in V} p\left(x_v \mid x_{pa(v)}\right)$$

$$p(x_1, x_2, x_3, x_4) \triangleq p(x_1)\, p(x_2 \mid x_1)\, p(x_3 \mid x_2)\, p(x_4 \mid x_1, x_3)$$

# DGM Example: HMM viewed as DGM




The joint probability distribution of a hidden Markov model (HMM) :

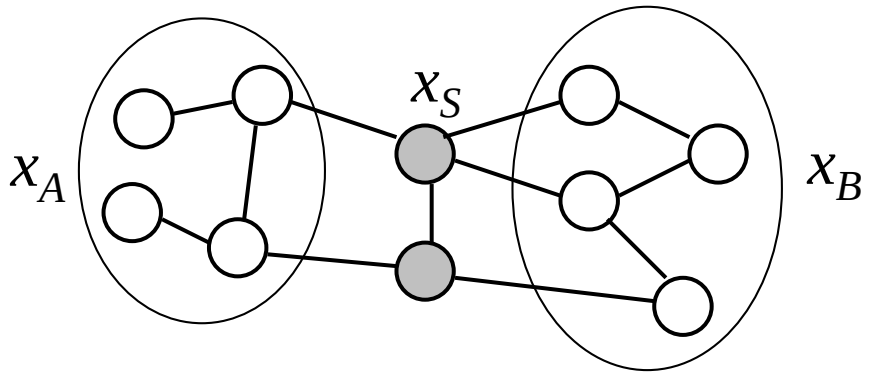
$$p(q_{1:T}, y_{1:T}) = p(q_1) \cdot \prod_{t=1}^{T-1} p(q_{t+1} \mid q_t) \cdot \prod_{t=1}^{T} p(y_t \mid q_t)$$

Parameterized by: $(\pi, A, B)$

# UGM Semantics - (G) property

- A probability distribution $p(x_V)$ is said to obey the global Markov property, relative to $g$ , if for any triple $(A, B, S)$ of disjoint subsets of $V$ such that $S$ *separates* $A$ *from* $B$,

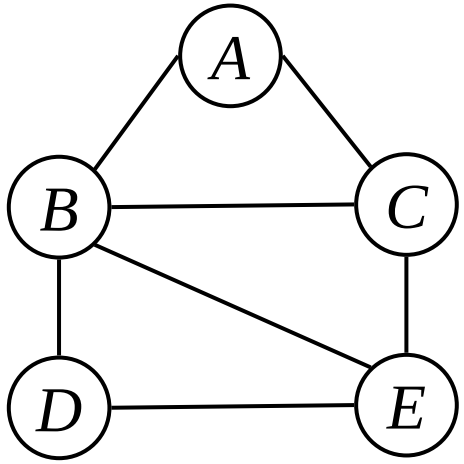
$$x_A \perp x_B \mid x_S$$




$S$ *separates* $A$ *from* $B$ : if all trails from $A$ to $B$ intersect $S$

# Graph theory basics

| | | |
|---|---|---|
| clique | $C$ | A subset of nodes $C$ is called a clique, if every pair of nodes in $C$ is joined. |
| maximal clique | | A clique which is maximal (with respect to $\subseteq$) |



# UGM Semantics - Factorization property (F)

- A probability distribution $p(x_V)$ is said to factorize according to $g$, if there exist non-negative functions (called potential functions) $\phi_C(x_C)$ for all cliques $C$ such that

$$p(x_V) = \frac{1}{Z}\prod_{C\in\mathcal{C}}\phi_C(x_C) \quad \text{or} \quad p(x_V) \propto \prod_{C\in\mathcal{C}}\phi_C(x_C)$$

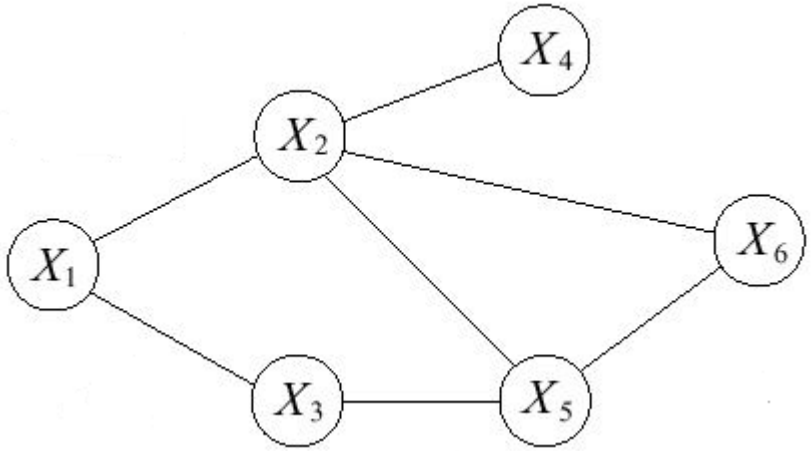
where $Z$ is the normalizing constant (partition function)

$$Z = \sum_{x_V}\prod_{C\in\mathcal{C}}\phi(x_C)$$

  - Potential functions $\phi_C(x_C)$ are not uniquely determined.
  - Without loss of generality, define potentials over maximum cliques.

**Hammersley-Clifford Theorem: If $p$ is strictly positive, (F)⇔(G).**

# UGM Example




$$p(x_1,x_2,x_3,x_4,x_5,x_6) \propto \phi(x_1)\phi(x_2)\phi(x_3)\phi(x_4)\phi(x_5)\phi(x_6)^{=1}$$
$$\times \phi(x_1,x_2)\phi(x_1,x_3)\phi(x_2,x_4)\phi(x_3,x_5)\phi(x_2,x_5)\phi(x_2,x_6)\phi(x_5,x_6)$$
$$\times \phi(x_2,x_5,x_6)$$

$$p(x_1,x_2,x_3,x_4,x_5,x_6) \propto \psi(x_1,x_2)\psi(x_1,x_3)\psi(x_2,x_4)\psi(x_3,x_5)\psi(x_2,x_5,x_6)$$

# UGMs and Energy-based models

- Let every clique potential be associated with a clique energy $E(x_C)$

$$E_C(x_C) = -log\phi_C(x_C)$$

- The resulting joint is known as the Gibbs (or Boltzman) distribution

$$p(x_V) \propto exp\left[-\sum_C E_C(x_C)\right]$$

High probability states correspond to low energy configurations.

# UGMs and log-linear models

- Let each clique potential be a log-linear function

$$log\phi_C(x_C) = \theta_C^T f_C(x_C)$$

where $f_C(x_C)$ is a feature vector derived from the values of the variables $x_C$, $\theta_C$ is the associated feature weight vector.

- The resulting joint has the form

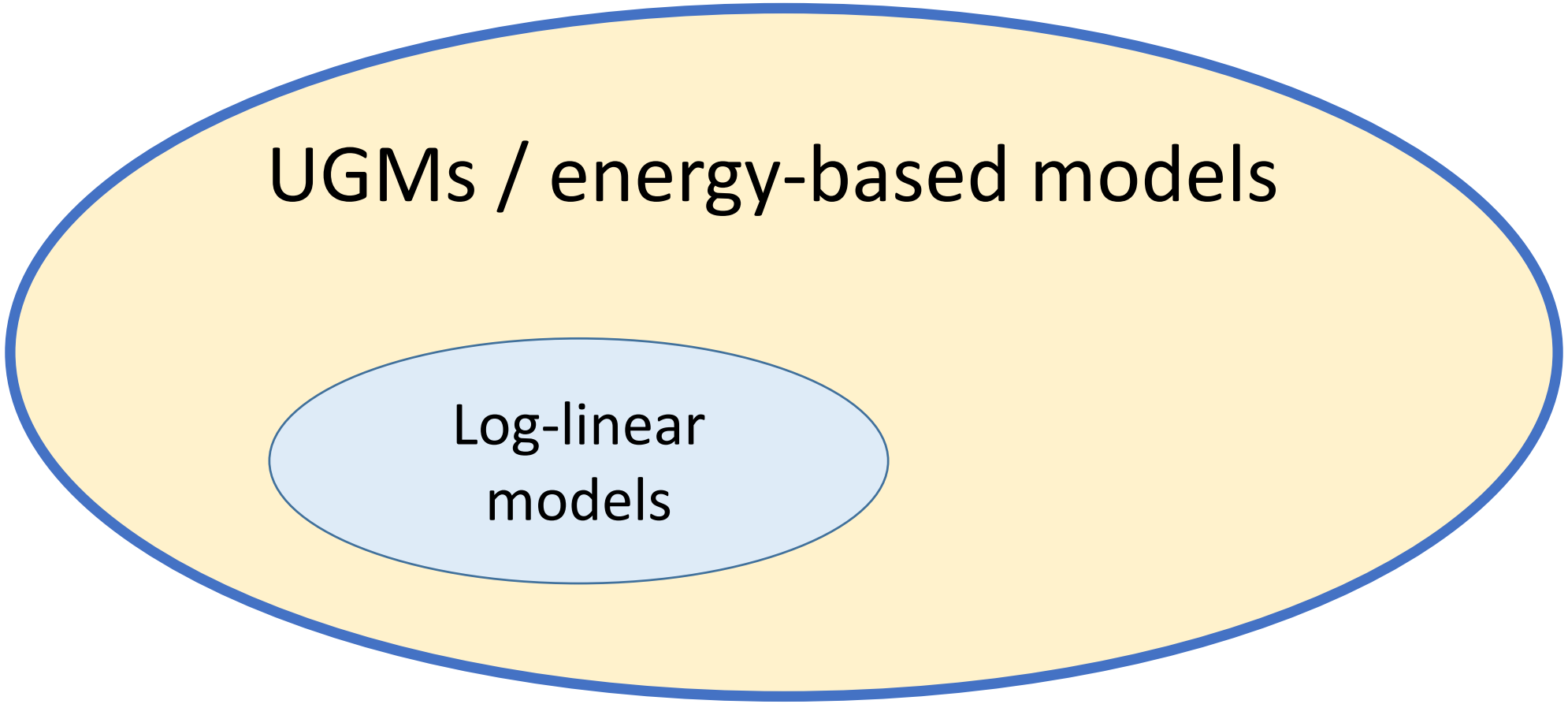
$$p(x_V) = \frac{1}{Z(\theta)} exp\left[\sum_C \theta_C^T f_C(x_C)\right]$$

This is known as a log-linear model or a Maximum Entropy model.

It can be proved that the maxent distribution is the same as
the maximum likelihood distribution from the closure of the set of log-linear RF distributions.

S. D. Pietra, V. D. Pietra, and J. Lafferty, "Inducing features of random fields", IEEE PAMI, 1997.

# Relationship between UGMs and other models



# Feature-based potential representation in log-linear models

- Consider an edge potential $\phi_{s,t}(x_s, x_t)$ associated with two discrete variables $x_s$ and $x_t$, both of which can take $K$ values.
- Define a feature vector of length $K^2$ as follows:

$$f_{s,t}(x_s, x_t) = [\cdots, 1(x_s = j, x_t = k), \cdots]^T, \qquad j, k = 1, \cdots, K$$

  with the associated weights:

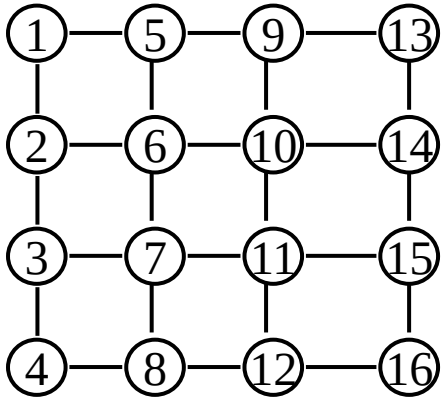
$$\theta_{s,t} = \left[\cdots, log\left(\phi_{s,t}(x_s = j, x_t = k)\right), \cdots\right]^T, \qquad j, k = 1, \cdots, K$$

- Then the tabular potential $\phi_{s,t}(x_s, x_t)$ can be represented as the log-liner form

$$\phi_{s,t}(x_s, x_t) = exp\left[\theta_{s,t}^T f_{s,t}(x_s, x_t)\right]$$

- Note: the log-linear form is more general because we can choose (or learn) the features.

# UGM Example - Ising model




- Consider a lattice of binary RV's, $x_i \in \{-1,1\}$

$$p\left(x_{1:N^2}\right) \propto \exp\left\{\sum_{(i,j)\in E} \psi\left(x_i, x_j\right)\right\} = \exp\left\{\beta \sum_{(i,j)\in E} x_i x_j\right\} \quad \beta > 0$$

  - $\beta$ : how much neighboring variables take identical values is favored.
  - Samples of Ising models on a lattice with different $\beta$:

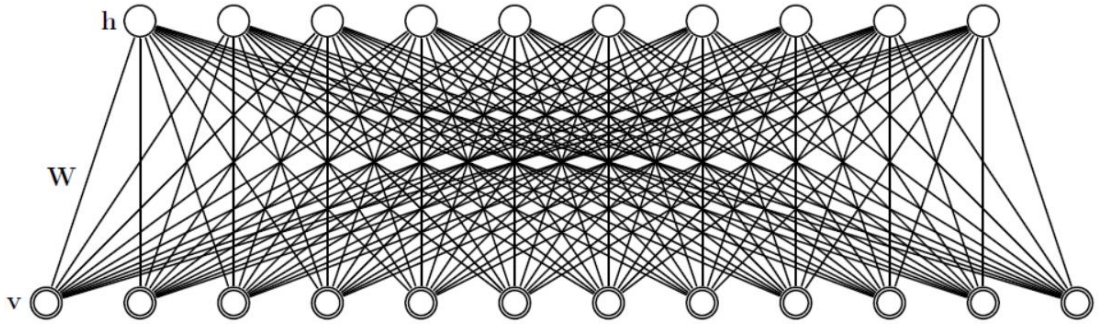
$\beta$ = 0.1 or 10 ?

$\beta$ = 1

$\beta$ = 0.1 or 10 ?

# Restricted Boltzmann Machines (RBMs)

- RBM is the main building block of a Deep Belief Network
- RBM is a two-layer MRF
  - Binary visible variables $v \in \{0,1\}^D$
  - Binary hidden variables $h \in \{0,1\}^F$
  - $\theta = \{W, b, a\}$




RBM: a stochastic version of a NN

$$p(v,h;\theta) = \frac{1}{Z(\theta)} exp[-E(v,h;\theta)]$$

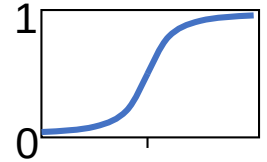
$$E(v,h;\theta) = -v^T W h - b^T v - a^T h$$

$$= -\sum_{i=1}^{D}\sum_{j=1}^{F} v_i W_{ij} h_j - \sum_{i=1}^{D} b_i v_i - \sum_{j=1}^{F} a_j h_j$$

$$p(h|v;\theta) = \prod_j p(h_j|v), \qquad p(h_j = 1|v) = \sigma\left(\sum_i W_{ij} v_i + a_j\right)$$

$$p(v|h;\theta) = \prod_i p(v_i|h), \qquad p(v_i = 1|h) = \sigma\left(\sum_j W_{ij} h_j + b_i\right)$$

Sigmoid function : $\sigma(x) = \frac{1}{1+e^{-x}}$

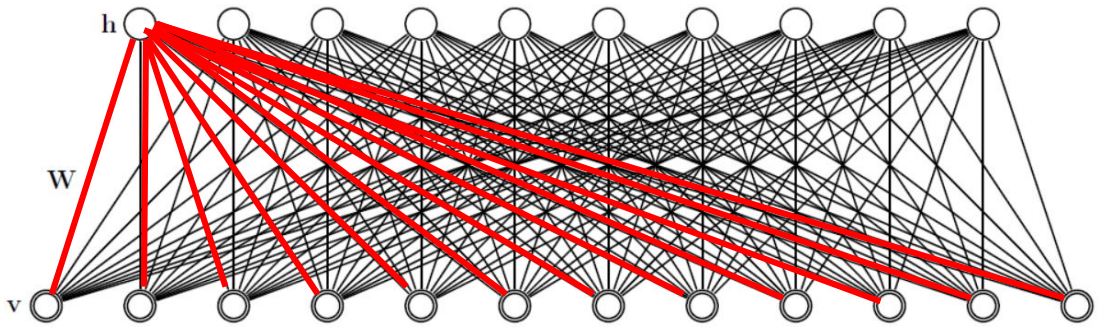



Ruslan Salakhutdinov. "Learning deep generative models". PhD thesis, University of Toronto, 2009.

# Learned features $W_{*j}$

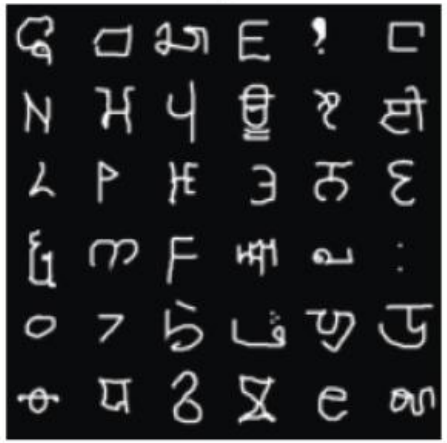
## Learned receptive fields for unit $h_j$

h: higher-level encoding of v



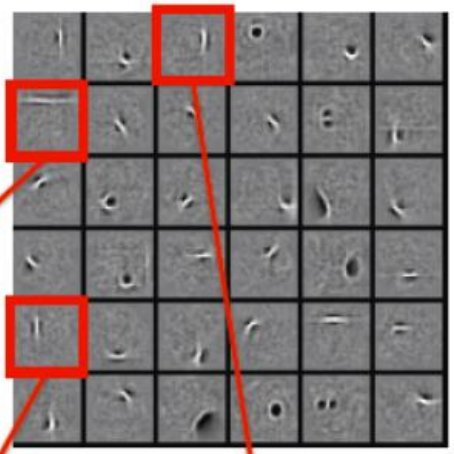

Observed Data

Subset of 25,000 characters

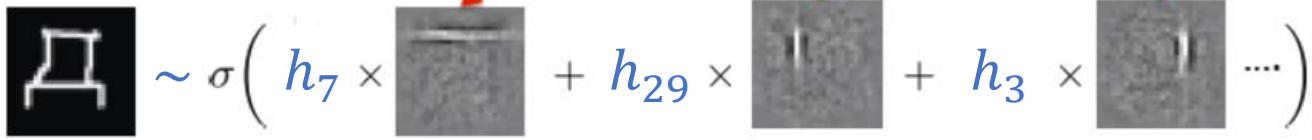


Learned W: "edges" /"parts"

Subset of 1000 features



$W_{*7}$ $W_{*29}$ $W_{*3}$



$$\sim \sigma\left( h_7 \times \quad + h_{29} \times \quad + h_3 \times \quad \cdots \right)$$

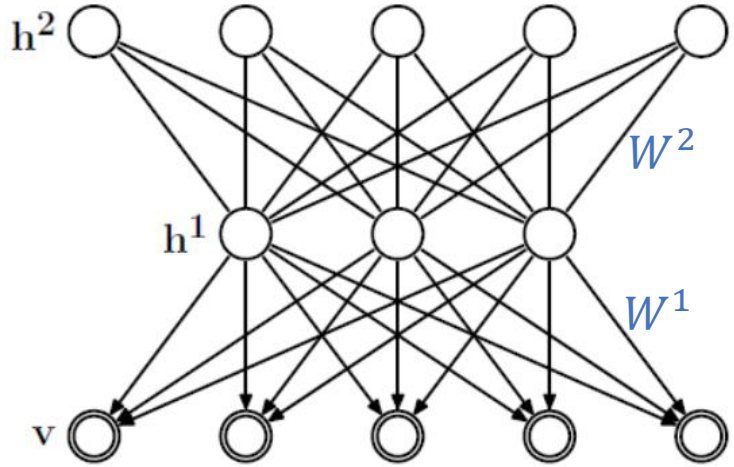
$$p(v|h;\theta) = \prod_i p(v_i|h),$$

$$p(v_i = 1|h) = \sigma\left(\sum_j W_{ij} h_j + b_i\right)$$

$$v|h \sim \sigma(h_1 \cdot W_{*1} + h_2 \cdot W_{*2} + \cdots + b)$$

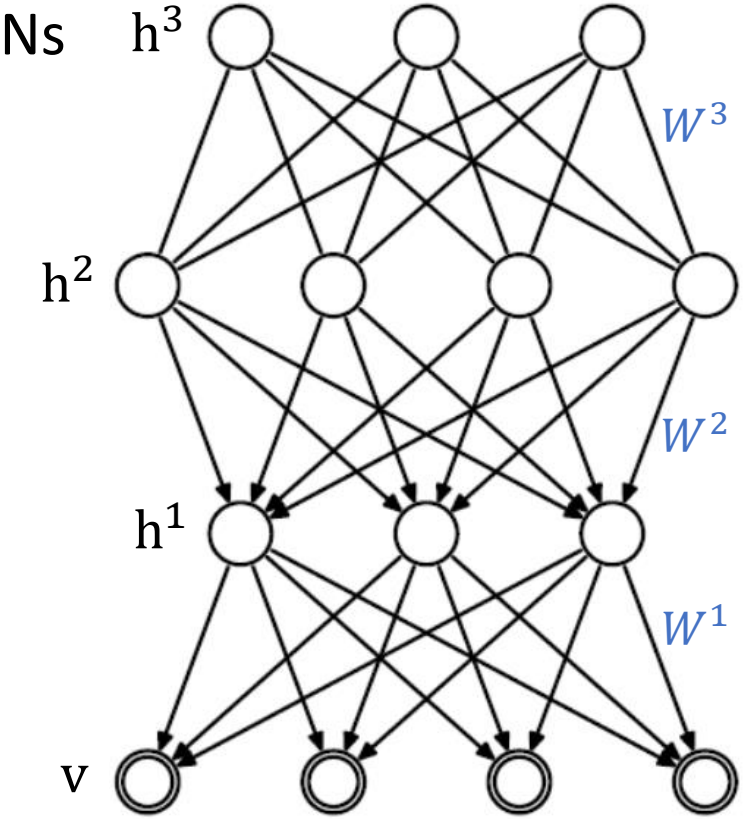
# Deep Belief Networks (DBNs)

- DBNs ignite Deep Learning, Science 2006
- DBN is a multilayer mixed directed and undirected model
  - Greedy layer-by-layer learning as pre-training for DNNs




$$p(v, h^1, h^2; \theta) = p(v|h^1; W^1)p(h^1, h^2; W^2)$$

$$\theta = \{W^1, W^2\}$$

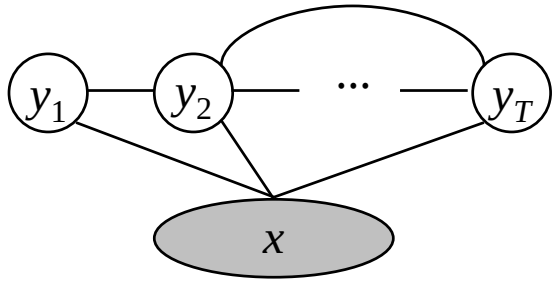



$$p(v, h^1, h^2, h^3; \theta) = p(v|h^1; W^1)p(h^1|h^2; W^2)p(h^2, h^3; W^3)$$

$$\theta = \{W^1, W^2, W^3\}$$

# Conditional Random Fields (CRFs)

- A CRF is a conditional distribution defined as a MRF




$$p(y|x) = \frac{1}{Z(x)} exp\left[\sum_C \psi_C(y_C, x)\right]$$

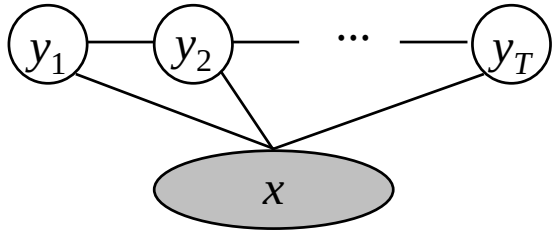
- $x$ is observed sequence, which is always given;
- $y$ is hidden sequence;
- $\psi_C(y_C, x)$ : Clique feature function.

J. Lafferty, A. McCallum, and F. Pereira, “Conditional random fields: Probabilistic models for segmenting and labeling sequence data”, ICML 2001.

# Linear-chain CRFs

for sequence tagging, e.g. POS tagging, shallow parser, Chinese word segmentation, …

$$p(y_{1:T} \mid x) \propto \exp\left\{\sum_{t=1}^{T-1} \psi_t(y_t, y_{t+1}, x) + \sum_{t=1}^{T} \psi_t(y_t, x)\right\}$$




Log-linear representation of tabular potentials

$$p(y_{1:T} \mid x) \propto \exp\left\{\sum_{t=1}^{T-1} \sum_{i} \lambda_i f_i(y_t, y_{t+1}, x, t) + \sum_{t=1}^{T} \sum_{j} \mu_j f_j(y_t, x, t)\right\}$$

Transition/edge features

$$\lambda_i f_i(y_t, y_{t+1}, x, t) = \lambda_i \cdot 1(y_t = prep, y_{t+1} = non)$$

State/node features

$$\mu_j f_j(y_t, x, t) = \mu_j \cdot 1(y_t = prep,\ x_t = \text{on})$$

$$\mu_j f_j(y_t, x, t) = \mu_j \cdot 1(y_t = adv,\ x_t \text{ ends in } ly)$$

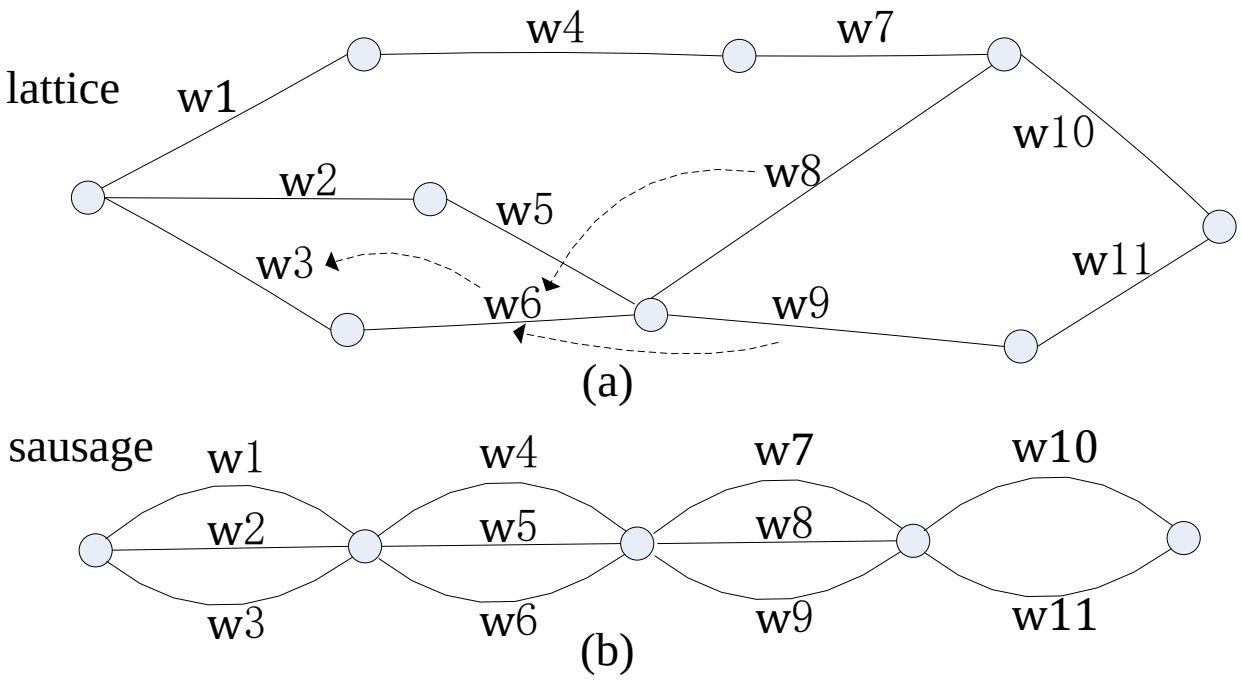
# Why UGMs ?

$$p(x_V) \triangleq \prod_{v \in V} p\left(x_v \mid x_{pa(v)}\right) \quad \text{vs} \quad p(x_V) = \frac{1}{Z} \prod_{C \in \mathcal{C}} \phi_C(x_C)$$

- Advantages over DGMs
  - Undirected modeling is more natural for co-occurrence, where fixing the directions of edges is awkward in a graphical model.
  - Avoid local normalization and acyclicity requirements
    - Potentially more powerful modeling capacity
      - e.g. CRFs overcome the label bias weakness.
    - Easily encode a much richer set of patterns/features
- Disadvantages over DGMs
  - Parameter learning in UGMs may be more computational expensive.
- The inference problem is (basically) the same in DGMs and UGMs.
  - UGMs are computational more efficient by avoiding softmax calculation

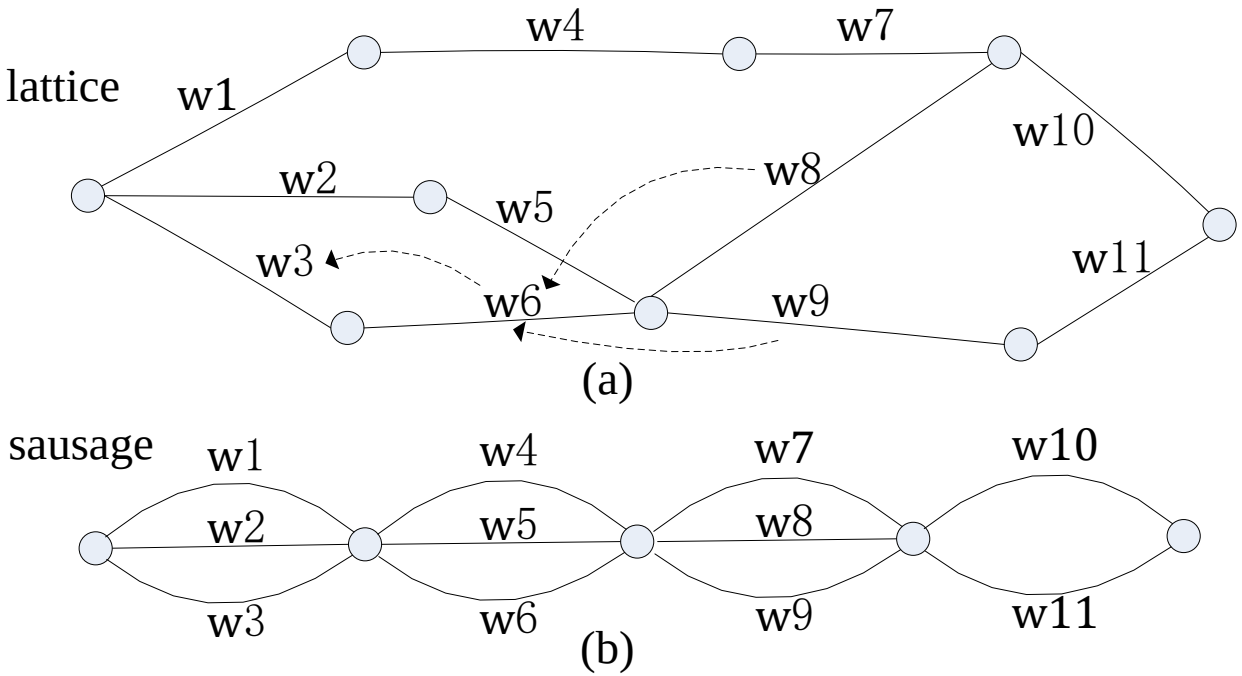
# Case study: CRF-based confidence measure (CM)

- Motivation
  - The use of forward-backward posterior probabilities as the confidence scores
  - Limitation: its performance for CMs cannot be improved easily.
  - Use CRFs to combine various relevant features !




**1. Reduce lattice to sausage (**a linear sequence of slices**) so that (linear-chain) CRFs can be used.**

# Case study: CRF-based confidence measure (CM)




## 2. Define the CRF over sausage

Given the sausage $y$, the reliability of the word candidate w4 is $p(q_2$=w4 $|y)$.

$$p(q|y) \propto exp\left\{\sum_{n=1}^{N}\phi_n(q_n,y)+\sum_{n=2}^{N}\psi_n(q_{n-1},q_n,y)\right\}$$

Z. Ou, H. Luo. "CRF-based Confidence Measures of Recognized Candidates for Lattice-based Audio Indexing." ICASSP 2012.

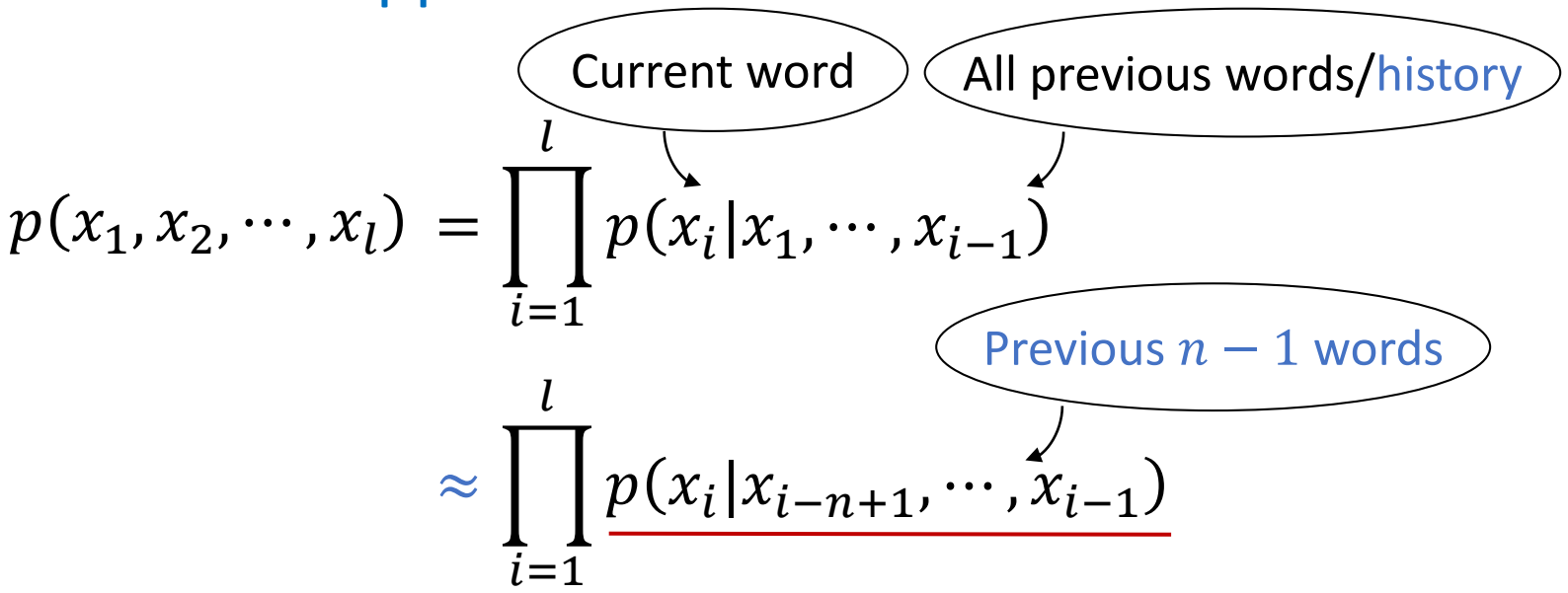
# Case study

Trans-dimensional Random Field Language Models (TRF LMs) – brand new

- State-of-the-art LMs - review
  - N-gram LMs
  - Neural network LMs
- Motivation - why
- Model formulation - what
- Model Training - breakthrough
- Experiment results - evaluation
- Summary

# N-gram LMs

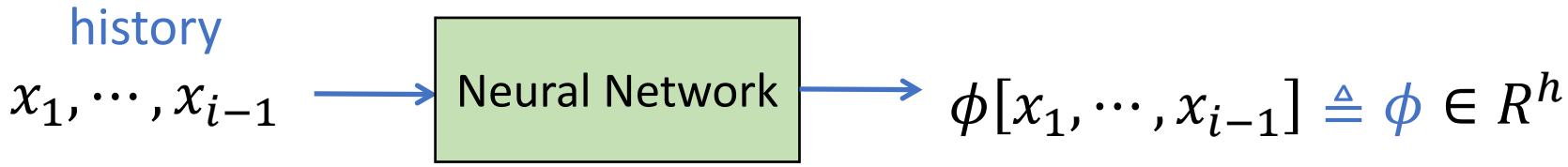
- Language modeling (LM) is to determine the joint probability of a sentence, i.e. a word sequence.
- Dominant: Conditional approach




- Using Markov assumption leads to the N-gram LMs
  - One of the state-of-the-art LMs

# Neural network LMs

- Another state-of-the-art LMs




$$p(x_i|x_1, \cdots, x_{i-1}) \approx p(x_i|\phi[x_1, \cdots, x_{i-1}])$$

$$p(x_i = k|x_1, \cdots, x_{i-1}) \approx \frac{\phi^T w_k}{\sum_{k=1}^{V} \phi^T w_k}$$ where $V$ is lexicon size, $w_k \in R^h$

☹ Computational very expensive in both training and testing [1]

e.g. $V = 10k \sim 100k, h = 250$

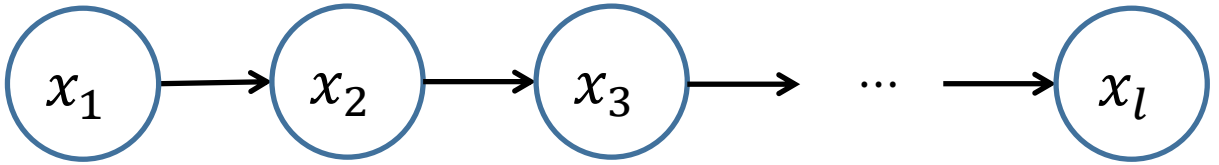
[1] Partly alleviated by using un-normalized models, e.g. through noise contrastive estimation training.

# TRF LMs – Motivation (1)

$$p(x_1, x_2, \cdots, x_l) = ?$$

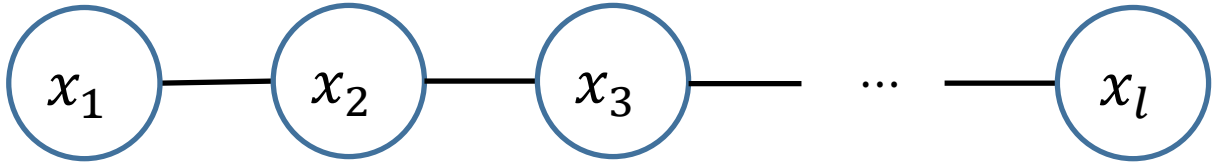
## Dominant:

Conditional approach / Directed




## Alternative:

Random field approach / Undirected




☹ Difficulty in model training

☺ A rule in language cognition: employ context for reading and writing

The cat is on the table.

The cat is in the house.

# TRF LMs – Motivation (2)

- Drawback of N-gram LMs
  - N-gram is only one type of linguistic feature/property/constraint
  - meeting on Monday

$$P(w_i = Monday | w_{i-2} = meeing, w_{i-1} = on)$$

  - What if the training data only contain 'meeting on Monday' ?
  - New feature 'meeting on DAY-OF-WEEK', using class
  - New feature 'party on *** birthday', using skip
  - New features ….

**F. Jelinek, 1932 – 2010**

- 1985: Every time I fire a linguist, the performance of the speech recognizer goes up.
- 1995: put language back into language modeling.

# TRF LMs – Formulation

- Intuitive idea
  - Features $(f_i, i = 1,2, \dots, F)$ can be defined flexibly, beyond the n-gram features.
  - Each feature brings a contribution to the sentence probability $p(x)$
- Formulation

$$p(x) = \frac{1}{Z} \exp\left( \sum_{i=1}^{F} \lambda_i f_i(x) \right), x \triangleq (x_1, x_2, \cdots, x_l)$$

$$f_i(x) = \begin{cases} 1, & \text{'meeting on DAY–OF–WEEK' appears in } x \quad \Rightarrow \lambda_i \text{ is activated} \\ 0, & \text{Otherwise} \qquad\qquad\qquad\qquad\quad \Rightarrow \lambda_i \text{ is removed} \end{cases}$$

☺ More flexible features, beyond the n-gram features, can be well supported in RFLMs.

☺ Computational very efficient in computing sentence probability.

# TRF LMs – Breakthrough in training (1)

- Propose Joint Stochastic Approximation (SA) Training Algorithm
  - Simultaneously updates the model parameters and normalization constants

**Algorithm 1** Joint stochastic approximation

**Input:** training set

set initial values $\lambda^{(0)} = (0, \ldots, 0)^T$ and $\zeta^{(0)} = \zeta^*(\lambda^{(0)}) - \zeta_1^*(\lambda^{(0)})$

**for** $t = 1, 2, \ldots, t_{max}$ **do**

  set $B^{(t)} = \emptyset$

  set $(L^{(t,0)}, X^{(t,0)}) = (L^{(t-1,K)}, X^{(t-1,K)})$

  ***Step I: MCMC sampling***

  **for** $k = 1 \rightarrow K$ **do**

    sampling (See Algorithm 3) $(L^{(t,k)}, X^{(t,k)}) = SAMPLE(L^{(t,k-1)}, X^{(t,k-1)})$

    set $B^{(t)} = B^{(t)} \cup \{(L^{(t,k)}, X^{(t,k)})\}$

  **end for**

  ***Step II: SA updating***

  Compute $\lambda^{(t)}$ based on (13)

  Compute $\zeta^{(t)}$ based on (14) and (15)

**end for**

# TRF LMs – Breakthrough in training (2)

- Propose Trans-dimensional mixture sampling
  - Sampling from $p(l, x^l; \lambda, \zeta)$, a mixture of RFs on subspaces of different dimensions.
  - Formally like RJ-MCMC (Green, 1995).

**function** SAMPLING($(L^{(t-1)}, X^{(t-1)})$)
  set $k = L^{(t-1)}$
  set $L^{(t)} = k$
  set $X^{(t)} = X^{(t-1)}$
  ***Stage I: Local jump***
  generate $j \sim \Gamma(k, \cdot)$
  **if** $j = k + 1$ **then**
    generate $Y \sim g_{k+1}(y|X^{(t-1)})$ (equ.24)
    set $L^{(t)} = j$ and $X^{(t)} = \{X^{(t-1)}, Y\}$ with probability equ.22
  **end if**
  **if** $j = k - 1$ **then**
    set $L^{(t)} = j$ and $X^{(t)} = X^{(t-1)}_{1:k-1}$ with probability equ.23
  **end if**
  ***Stage II: Markov move***
  **for** $i = 1 \rightarrow L^{(t)}$ **do**
    $a \sim p(L^{(t)}, \{X^{(t)}_{1:i-1}, \cdot, X^{(t)}_{i+1:L^{(t)}}\}; \Lambda, \zeta)$
    $X^{(t)}_i \leftarrow a$
  **end for**
  **return** $(L^{(t)}, X^{(t)})$
**end function**

# Experiment results

- Benchmarking experiments
  - Speech recognition on PTB-WSJ dataset
  - Speech recognition on ChiME-4 dataset
  - Mandarin speech recognition on Toshiba dataset
- TRF LMs significantly outperform KN n-gram LMs (10%+ WER relative reduction), and perform better than RNN LMs and close to LSTM LMs but with much faster speed in computing sentence probabilities (0.16 sec. CPU vs 40 sec. GPU).
- Interpolated TRF and LSTM is better than Interpolated KN5 and LSTM.


- Bin Wang, Zhijian Ou, Zhiqiang Tan, “Trans-dimensional Random Fields for Language Modeling”, ACL 2015.
- Bin Wang, Zhijian Ou, Yong He, and Akinori Kawamura, "Model Interpolation with Trans-dimensional Random Field Language Models for Speech Recognition", *arXiv* 2016.
- Hongyu Xiang, Bin Wang and Zhijian Ou. “The THU-SPMI CHiME-4 system : Lightweight design with advanced multi-channel processing, feature enhancement, and language modeling”. CHiME Workshop, 2016,9.

Once said in : J. Goodman, “A bit of progress in language modeling”, Computer Speech & Language, 2001.

## 11.2 All hope abandon, ye who enter here

In this section,[10] we argue that meaningful, practical reductions in word error rate are hopeless. We point out that trigrams remain the de facto standard not because we don't know how to beat them, but because no improvements justify the cost. We claim with little evidence that entropy is a more meaningful measure of progress than perplexity, and that entropy improvements are small. We conclude that most language modeling research, including ours, by comparing to a straw man baseline and ignoring the cost of implementations, presents the illusion of progress without the substance. We go on to describe what, if any, language modeling research is worth the effort.

Begin long rambling cynical diatribe – no results or particularly novel ideas. Grad students thinking about research in language modeling *should* read this section

### 11.2.1 Practical word error rate reductions are hopeless

Most language modeling improvements require significantly more space than the trigram baseline compared to, and also typically require significantly more

Now we can beat n-gram significantly by RNN (Mikolov, 2010) and TRF (2015) …

# Contents

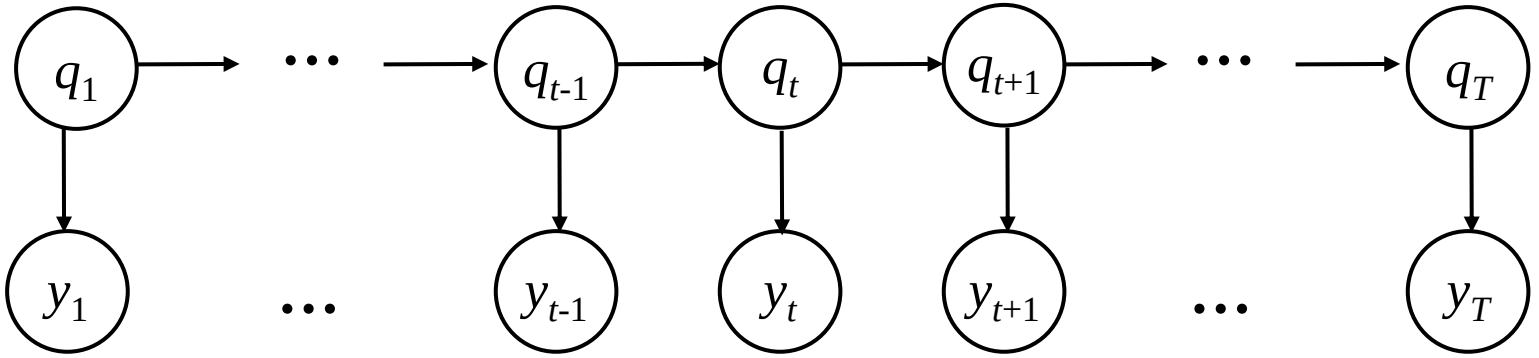
1. Semantics of DGMs and UGMs
   - RBMs, DBNs, CRFs, TRFs
2. Exact inference - variable elimination
3. Approximate inference – variational
4. Approximate inference – Monte Carlo
5. Learning
6. Summary

# Inference

Calculate $p(x_Q \mid X_E = x_E)$ $\quad e.g.\ p(q_t \mid y_{1:T}) = ?$

query nodes

evidence



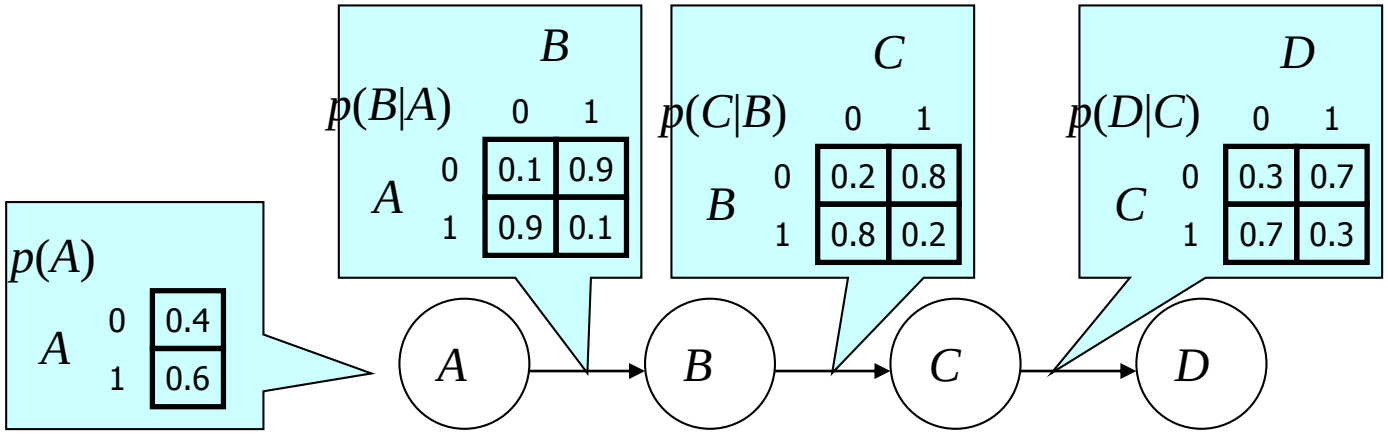

$$p(x_Q \mid x_E) \propto p(x_Q, x_E) = \sum_{x_{V \setminus (Q,E)}} p(x_Q, x_E, x_{V \setminus (Q,E)})$$

Sum-product !

$$p(q_t \mid y_{1:T}) \propto p(q_t, y_{1:T}) = \sum_{q_{1:T \setminus \{t\}}} p(q_{1:T}, y_{1:T}) = \sum_{q_{1:T \setminus \{t\}}} p(q_1) \cdot \prod_{t=1}^{T-1} p(q_{t+1} \mid q_t) \cdot \prod_{t=1}^{T} p(y_t \mid q_t)$$

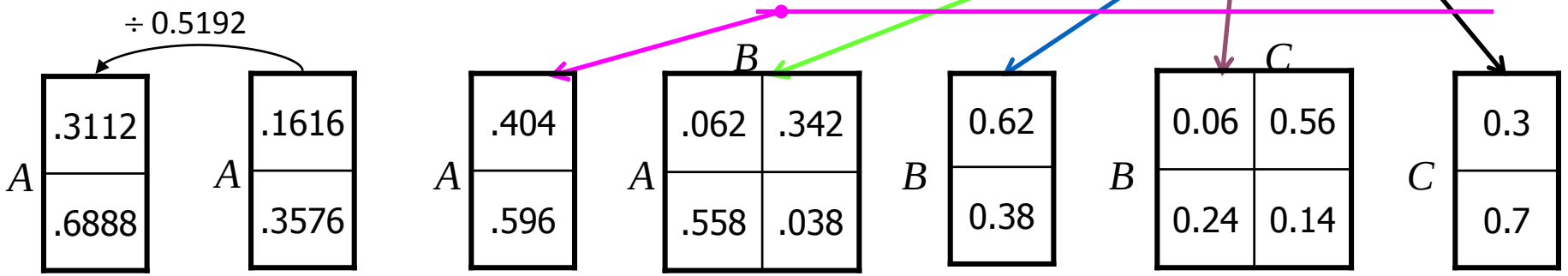
given $(\pi,\ A,\ B)$

# Inference example




$$p(A \mid D=0) \propto p(A, D=0) = \sum_B \sum_C p(A)\, p(B \mid A)\, p(C \mid B)\, p(D=0 \mid C)$$

$$= p(A) \sum_B p(B \mid A) \sum_C p(C \mid B)\, p(D=0 \mid C)$$




Two basic operations: product, marginalization

# Comment

- Exponential reduction in computation !
  - Elimination/marginalization/summation with respect to a variable should be performed as early as possible, i.e. Move the sum to the rightmost !
  - Cache intermediate results.
- Variable elimination (bucket elimination)
  - A systematic formulation of the heuristic operations.

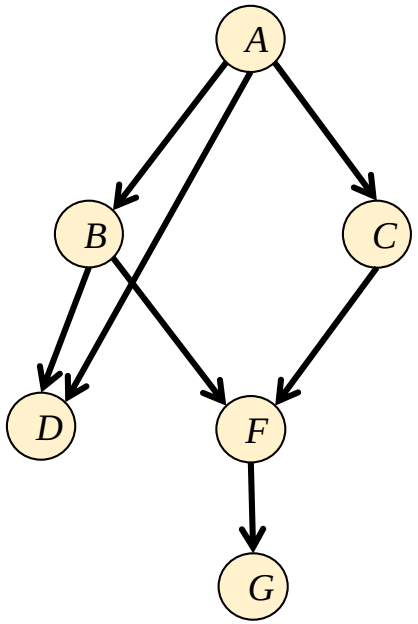

- R. Dechter, "Bucket Elimination: A Unifying Framework for Probabilistic Inference", UAI 1996.
- S. M. Aji and R. J. McEliece, “The generalized distributive law,” IEEE Trans. Information Theory, 2000.

# Bucket elimination *elim-bel* (Dechter 1996)

$$p(a \mid g=1) \propto \sum_c \sum_b \sum_f \sum_d \sum_g p(a)\, p(c \mid a)\, p(b \mid a)\, p(f \mid b,c)\, p(d \mid a,b)\, p(g \mid f)\, \delta(g=1)$$

Given an **elimination ordering** of the variables (e.g. $a, c, b, f, d, g$), beginning with the query variable.

1. Initialize. For a function, find the its argument which is to be eliminated earliest, then place the function to the bucket corresponding to this argument;
2. Processing each bucket / eliminate each variable according to the order;
3. Return: posterior ∝ all functions in the query variable's bucket.




$\text{bucket}_G$ : $p(g \mid f)$ $\delta(g=1)$

$\text{bucket}_D$ : $p(d \mid a,b)$

$\text{bucket}_F$ : $p(f \mid b,c)$ $\lambda_G(f) = p(g=1 \mid f)$

$\text{bucket}_B$ : $p(b \mid a)$ $\lambda_D(a,b)$ $\lambda_F(b,c)$

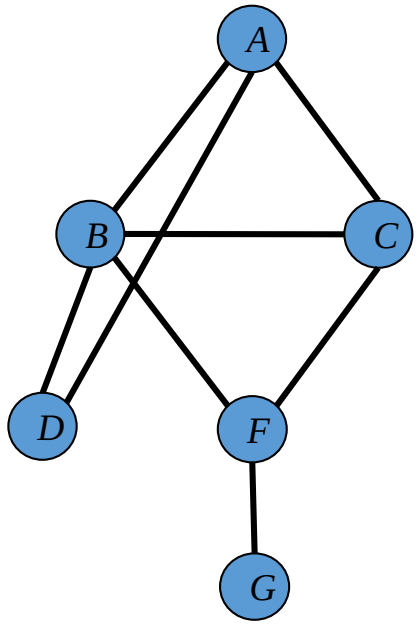
$\text{bucket}_C$ : $p(c \mid a)$ $\lambda_B(a,c)$

$\text{bucket}_A$ : $p(a)$ $\lambda_C(a)$

# Bucket elimination *for UGMs*

$$p(a,b,c,d,f,g) = \frac{1}{Z}\phi(a,b,c)\phi(b,c,f)\phi(a,b,d)\phi(f,g)$$

$$p(a \mid g=1) \propto \sum_c \sum_b \sum_f \sum_d \sum_g \phi(a,b,c)\phi(b,c,f)\phi(a,b,d)\phi(f,g)\delta(g=1)$$




$\text{bucket}_G$ : $\phi(f,g)$ $\delta(g=1)$

$\text{bucket}_D$ : $\phi(a,b,d)$

$\text{bucket}_F$ : $\phi(b,c,f)$ $\lambda_G(f) = \phi(f,g=1)$

$\text{bucket}_B$ : $\phi(a,b,c)$ $\lambda_D(a,b)$ $\lambda_F(b,c)$

$\text{bucket}_C$ : $\lambda_B(a,c)$

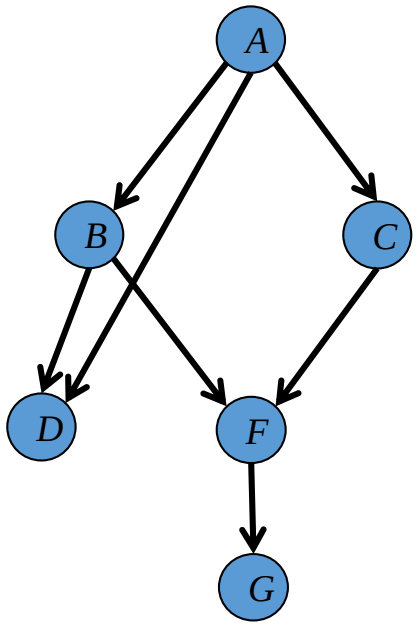
$\text{bucket}_A$ : $\lambda_C(a)$

# Finding MPE (Most Probable Explanation) *elim-mpe* (Dechter 1996)

$$MPE = \arg\max_{a,c,b,f,d} p(a,c,b,f,d \mid g=1)$$

$$\max_{a,c,b,f,d} p(a)\,p(c \mid a)\,p(b \mid a)\,p(f \mid b,c)\,p(d \mid a,b)\,p(g=1 \mid f)$$

$\sum$ is replaced by max




$D$: $\max_d p(d \mid a,b)$

$F$: $p(f \mid b,c)$ $p(g=1 \mid f)$

$B$: $p(b \mid a)$ $\lambda_D(a,b)$ $\lambda_F(b,c)$

$C$: $p(c \mid a)$ $\lambda_B(a,c)$

$A$: $p(a)$ $\lambda_C(a)$

$MPE$

$$D^*(a,b) = \arg\max_d p(d \mid a,b)$$

$$F^*(b,c) = \arg\max_f p(f \mid b,c)\,p(g=1 \mid f)$$

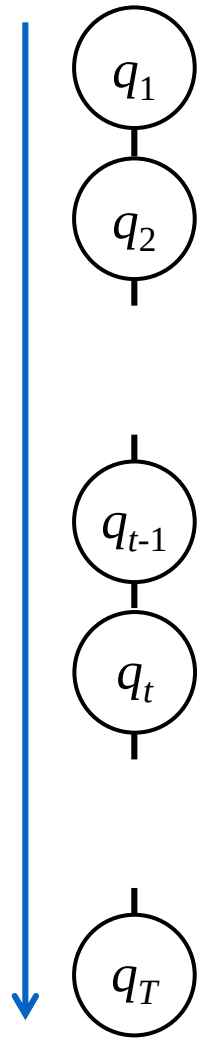
$$B^*(a,c) = \arg\max_b p(b \mid a)\,\lambda_D(a,b)\,\lambda_F(b,c)$$

$$C^*(a) = \arg\max_c p(c \mid a)\,\lambda_B(a,c)$$

$$A^* = \arg\max_a p(a)\,\lambda_C(a)$$

# Viterbi algorithm: MPE

$$\max_{q_{1:T}} p(q_{1:T} \mid y_{1:T}) \propto \max_{q_T \cdots q_1} p(q_1) \cdot \prod_{t=1}^{T-1} p(q_{t+1} \mid q_t) \cdot \prod_{t=1}^{T} p(y_t \mid q_t)$$




$$p(q_1)\, p(q_2 \mid q_1)\, p(y_1 \mid q_1)$$

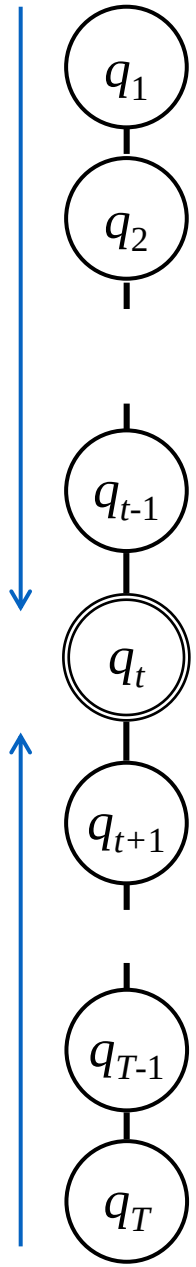
$$p(q_t \mid q_{t-1})\, p(y_{t-1} \mid q_{t-1})\, \lambda(q_{t-1})$$

$$\lambda(q_t) = \max_{q_{t-1}} p(q_t \mid q_{t-1})\, \underbrace{p(y_{t-1} \mid q_{t-1})\, \lambda(q_{t-1})}$$

$$\delta(q_t) = p(y_t \mid q_t) \max_{q_{t-1}} p(q_t \mid q_{t-1})\, \delta(q_{t-1})$$

$$p(y_T \mid q_T)\, \lambda(q_T)$$

# Forward-backward algorithm




$$p(q_1)\,p(q_2 \mid q_1)\,p(y_1 \mid q_1)$$

$$p(q_t \mid y_{1:T}) \propto \sum_{q_{1:T} \setminus q_t} p(q_1) \cdot \prod_{t=1}^{T-1} p(q_{t+1} \mid q_t) \cdot \prod_{t=1}^{T} p(y_t \mid q_t)$$

$$p(q_t \mid y_{1:T}) \propto \underbrace{p(y_t \mid q_t)\,\lambda(q_t)}_{\alpha(q_t)} \cdot \beta(q_t)$$

$$p(q_{t-1} \mid q_{t-2})\,p(y_{t-1} \mid q_{t-1})\,\lambda(q_{t-1})$$

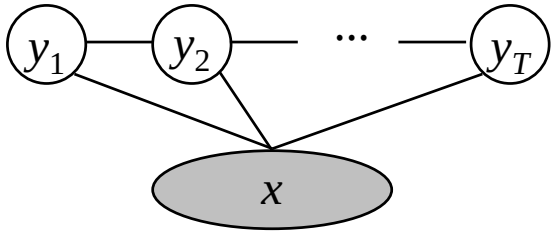
$$p(y_t \mid q_t)\,\lambda(q_t) \qquad \beta(q_t) = \sum_{q_{t+1}} p(q_{t+1} \mid q_t)\,p(y_{t+1} \mid q_{t+1})\,\beta(q_{t+1})$$

$$p(q_{t+1} \mid q_t)\,p(y_{t+1} \mid q_{t+1})\,\beta(q_{t+1})$$

$$p(q_{T-1} \mid q_{T-2})\,p(y_{T-1} \mid q_{T-1})\,\beta(q_{T-1})$$

$$p(q_T \mid q_{T-1})\,p(y_T \mid q_T)$$

# Linear-chain CRFs




$$p(y_{1:T} \mid x) \propto \exp\left\{\sum_{t=1}^{T-1}\psi_t(y_t, y_{t+1}, x) + \sum_{t=1}^{T}\psi_t(y_t, x)\right\}$$

Likelihood calculation——Forward-backward Algorithm

$$p(q_t \mid y) \propto \sum_{q_{1:T}\setminus q_t} \exp\left\{\sum_{t=1}^{T-1}\psi_t(q_t, q_{t+1}, y) + \sum_{t=1}^{T}\psi_t(q_t, y)\right\}$$

Decoding (recognition)——Viterbi Algorithm

$$\max_{q_{1:T}} p(q_{1:T} \mid y) \propto \max_{q_T \cdots q_1} \exp\left\{\sum_{t=1}^{T-1}\psi_t(q_t, q_{t+1}, y) + \sum_{t=1}^{T}\psi_t(q_t, y)\right\}$$

# Complexity analysis

- Complexities (time and space) vary greatly for different elimination ordering.
- Complexity of processing a bucket

$$\lambda_i(\cdot) = \sum_{x_i} \prod_{h \in \text{bucket}_i} h \qquad scope(\lambda_i) = \left[ \bigcup_{h \in \text{bucket}_i} scope(h) \right] - \{X_i\}$$

$$\lambda_B(a,c) = \sum_b p(b \mid a)\, \lambda_D(a,b)\, \lambda_F(b,c)$$

The scope of a function is the set of its arguments

  - Time complexity for computing message functions $r^{|scope(\lambda_i)|+1}$
  - Space complexity for saving message functions $r^{|scope(\lambda_i)|}$

(1) Finding good elimination ordering → triangulation.

(2) Reducing redundant computation for multiple runs of elimination for different queries over the same model → Junction Tree algorithm.

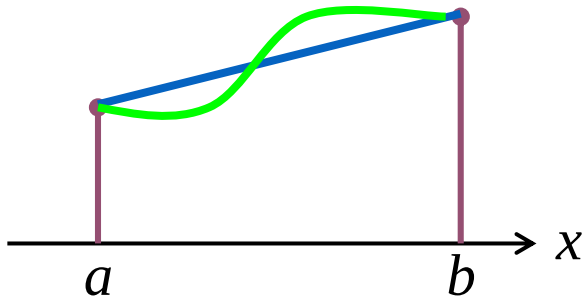
# Contents

1. Semantics of DGMs and UGMs
   - RBMs, DBNs, CRFs, TRFs
2. Exact inference - variable elimination
3. Approximate inference – variational
4. Approximate inference – Monte Carlo
5. Learning
6. Summary

# Variational principle

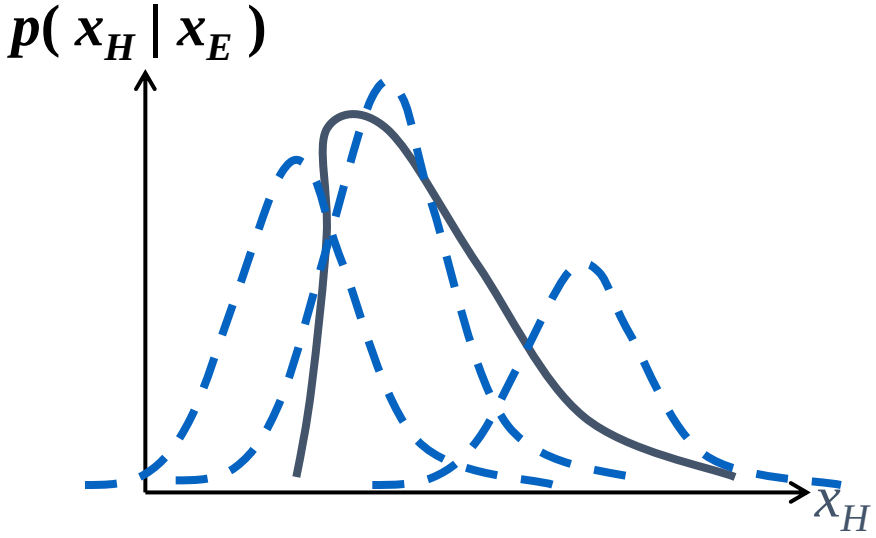
- Variational principle : Optimization over functions

$$\max_f J(f) = \int_a^b \sqrt{1+\dot{f}^2}\,dx$$




- Variational inference
  - Find a function $q(x_H)$ from some simpler and tractable family to approximate the target $p(x_H|x_E)$

$$\hat{q}(x_H) = \arg\min_q \underbrace{KL\left(q(x_H)\,\|\,p(x_H \mid x_E)\right)}_{J(q(x_H))}$$



# Minimize the (exclusive) KL divergence

$$KL(q \| p) = \sum_{x_H} q(x_H) \log \frac{q(x_H)}{p(x_H \mid x_E)}$$

fixed maximise minimise

$$\log p(x_E) = L(q) + KL(q \| p)$$

$$L(q) = \sum_{x_H} q(x_H) \log \frac{p(x_H, x_E)}{q(x_H)}$$

$$\log p(x_E) \geq \sum_{x_H} q(x_H) \log \frac{p(x_H, x_E)}{q(x_H)} = \sum_{x_H} q(x_H) \log p(x_H, x_E) + H\left[q(x_H)\right]$$

Variational lower-bound

Inclusive KL divergence $KL(p \| q)$ → Expectation Propagation (Minka, 2001)

# Mean-field variational inference

- $$L(q) = \sum_{x_H} q(x_H) \log \frac{p(x_H, x_E)}{q(x_H)} = H(q) + \underbrace{\sum_{x_H} q(x_H) \log p(x_H, x_E)}_{E_q\left[\log p(x_H, x_E)\right]}$$
- $$\arg\max_{q:\, q(x_H) = \prod_{i \in H} q(x_i)} L(q) = ?$$

$$\underbrace{\sum_{x_H \setminus x_k} q(x_H \setminus x_k) \log p(x_H, x_E)}$$

- Mean-field update formula

$$\log q(x_k) = E_q\left[\log p(x_H, x_E) \mid x_k\right] + const, \quad k \in H$$

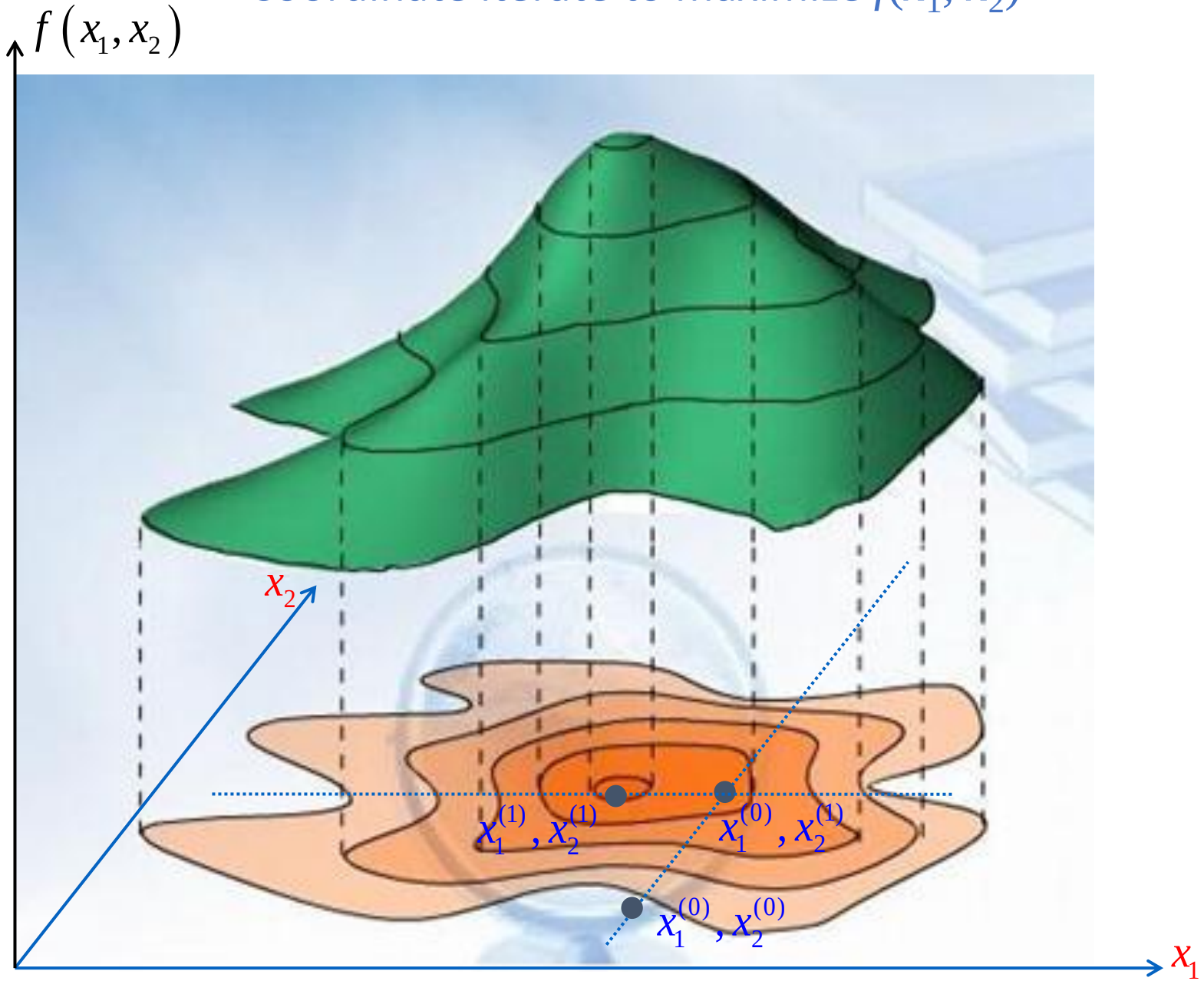
$$q(x_k) \propto \exp\left\{E_q\left[\log p(x_H, x_E) \mid x_k\right]\right\}$$

Iterate: $q^{(0)}(x_1), q^{(0)}(x_2), q^{(0)}(x_3), \cdots, q^{(0)}(x_K)$

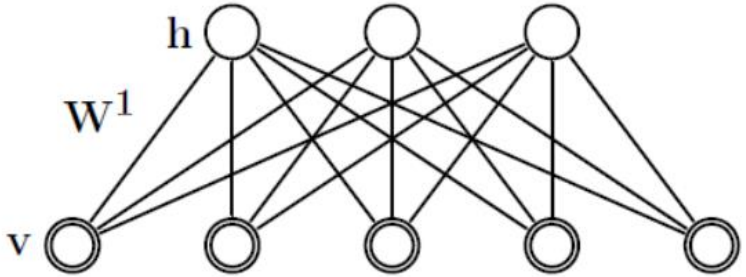
$q^{(1)}(x_1), q^{(0)}(x_2), q^{(0)}(x_3), \cdots, q^{(0)}(x_K)$

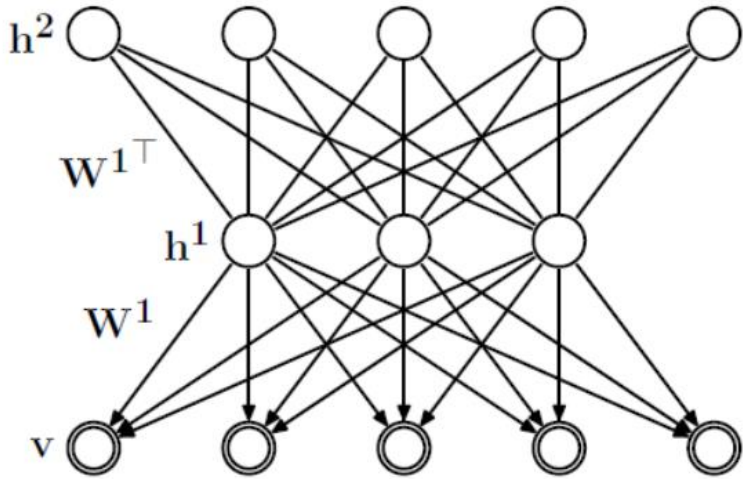
$q^{(1)}(x_1), q^{(1)}(x_2), q^{(0)}(x_3), \cdots, q^{(0)}(x_K)$

# Coordinate iterate to maximize $f(x_1, x_2)$



# Greedy learning of DBNs




- Key: $p\left(v, h^1; W^1, W^2 = {W^1}^T\right)$ defined by the right DBN with tied weights is identical to $p(v, h^1; W^1)$ defined by the left RBM.

$$logp(v; W^1, W^2) \geq \sum_{h^1} Q(h^1|v)[logp(h^1; W^2) + logp(v|h^1; W^1)] + H\big(Q(h^1|v)\big)$$

Take equality when setting $W^2 = {W^1}^T$ and $Q(h^1|v) = p(h^1|v; W^1)$

Greedy learning is to freeze $W^1$ and maximize the variational lower bound with respect to $W^2$ :

$$\sum_{h^1} p(h^1|v; W^1) logp(h^1; W^2)$$

# Contents

1. Semantics of DGMs and UGMs
   - RBMs, DBNs, CRFs, TRFs
2. Exact inference - variable elimination
3. Approximate inference – variational
4. Approximate inference – Monte Carlo
5. Learning
6. Summary

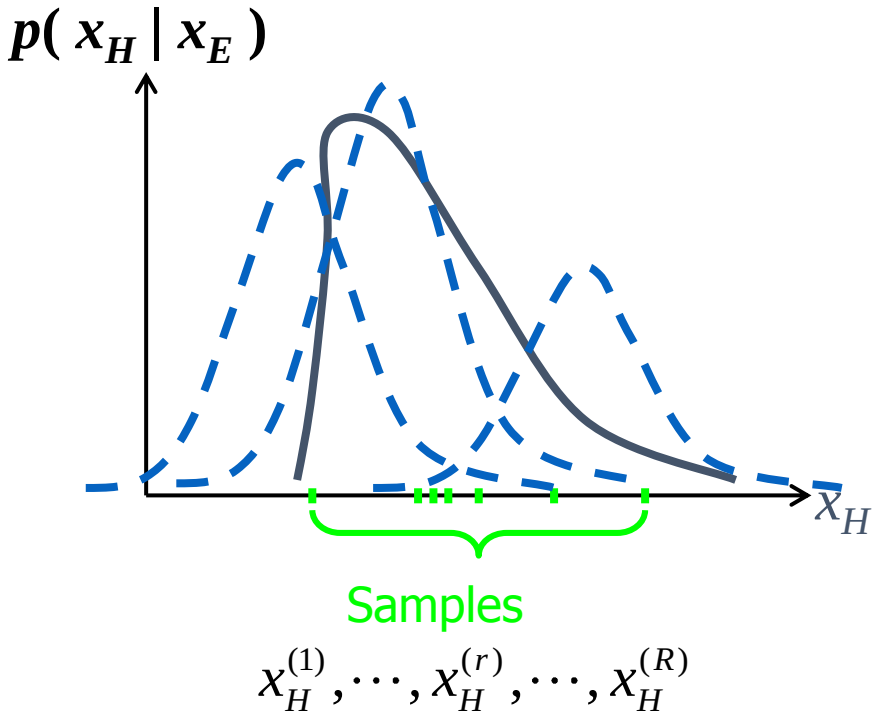
# Inference by Monte Carlo

- Monte Carlo methods are a broad class of computational algorithms that rely on repeated random sampling to obtain numerical results.
  - Evaluate integral $E_{p(x)}\left[\phi(x)\right] = \int \phi(x)\, p(x) dx$
  - Draw samples $x^{(r)} \sim p(x), r = 1, \cdots, R$

$$\int \phi(x)\, p(x) dx \approx \frac{1}{R} \sum_{r=1}^{R} \phi\left(x^{(r)}\right)$$

Probability approximated as frequencies

$$p(\bar{x}) = \int 1(x = \bar{x})\, p(x) dx \approx \frac{1}{R} \sum_{r=1}^{R} 1\left(x^{(r)} = \bar{x}\right)$$




  - The variance of the estimator decreases as $O(1/R)$.
  - Two broad classes: Markov Chain Monte Carlo (MCMC), Importance sampling

# MCMC example: Metropolis–Hastings algorithm

We want to Draw samples from the target distribution $p(x)$ ?

Solution: Construct a Markov chain that has $p(x)$ as the stationary distribution.

1. Random initialize $x_0$

2. For $t = 1, \cdots$

Generates $x^*$ from proposal/transition kernel $q(x^*|x_{t-1})$,

Accept $x_t = x^*$ with probability $min\left\{1, \frac{p(x^*)q(x_{t-1}|x^*)}{p(x_{t-1})q(x^*|x_{t-1})}\right\}$,

otherwise set $x_t = x_{t-1}$

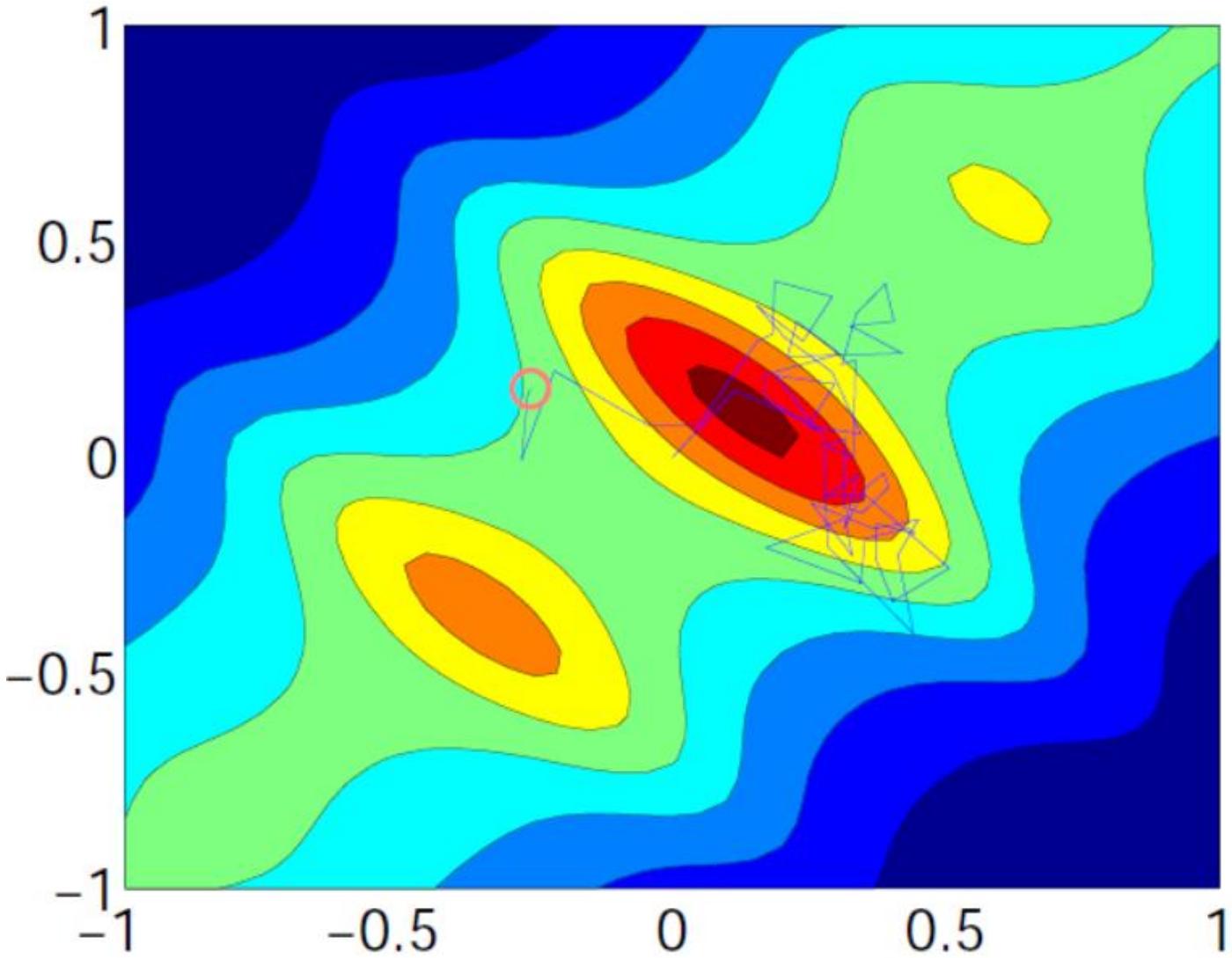
Burn-in : first few samples are discarded.

- For an irreducible & ergodic Markov chain, there exist stationary distribution $\pi$, which satisfies equation $\pi=\pi \mathrm{P}$.
- A sufficient condition: satisfy the detailed balance equation

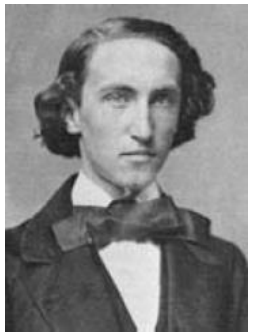
$$\pi_i \, \mathrm{P}_{ij} = \pi_j \, \mathrm{P}_{ji}$$

# Metropolis–Hastings example

e.g. $q(x^*|x_{t-1}) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x^*-x_{t-1})^2}{2\sigma^2}}$



# MCMC example: Gibbs sampling



J.W. Gibbs, 1839-1903

- Draw samples from $p(z) = p(z_1, \cdots, z_M)$

1. *Initialize* $\{z_1^{(1)}, z_2^{(1)}, z_3^{(1)}, z_4^{(1)}\}$

2. A sweep generates a full sample of $z^{(t)} = \{z_1^{(t)}, z_2^{(t)}, z_3^{(t)}, z_4^{(t)}\}$

   – *Sample* $z_1^{(2)} \sim p(z_1 \mid z_2^{(1)}, z_3^{(1)}, z_4^{(1)}, E)$

   – *Sample* $z_2^{(2)} \sim p(z_2 \mid z_1^{(2)}, z_3^{(1)}, z_4^{(1)}, E)$

   – *Sample* $z_3^{(2)} \sim p(z_3 \mid z_1^{(2)}, z_2^{(2)}, z_4^{(1)}, E)$

   – *Sample* $z_4^{(2)} \sim p(z_4 \mid z_1^{(2)}, z_2^{(2)}, z_3^{(2)}, E)$

# Mean-field inference vs Gibbs sampling

$$\log q(x_k) = E_q\left[\log p(x_H, x_E) \mid x_k\right] + const$$

Iterate :

$$q(x_1), q(x_2), q(x_3), \cdots, q(x_K)$$
$$\hat{q}(x_1), q(x_2), q(x_3), \cdots, q(x_K)$$
$$\hat{q}(x_1), \hat{q}(x_2), q(x_3), \cdots, q(x_K)$$

$$x_k - sampling\ from\ p\left(x_k \mid x_{H\setminus\{k\}}, x_E\right)$$
$$x_k - sampling\ from\ p(x_H, x_E)$$

Iterate :

$$x_1,\ x_2,\ x_3,\ \cdots,\ x_K$$
$$\hat{x}_1,\ x_2,\ x_3,\ \cdots,\ x_K$$
$$x_1,\ \hat{x}_2,\ x_3,\ \cdots,\ x_K$$

# Contents

1. Semantics of DGMs and UGMs
   - RBMs, DBNs, CRFs, TRFs
2. Exact inference - variable elimination
3. Approximate inference – variational
4. Approximate inference – Monte Carlo
5. Learning
6. Summary

# Training of UGMs in general

$$p(x;\theta) = \frac{1}{Z(\theta)} \exp[Q(x;\theta)]$$

Normalization constant:

$$Z(\theta) = \sum_x \exp[Q(x;\theta)]$$

- Maximum likelihood (ML) training

The scaled log-likelihood of observed $\{x_i, i = 1, \cdots, N\}$

$$l(\theta) \triangleq \frac{1}{N}\sum_{i=1}^{N} logp(x_i;\theta) = \left[\frac{1}{N}\sum_{i=1}^{N} Q(x_i;\theta)\right] - logZ(\theta)$$

$$\frac{\partial l(\theta)}{\partial \theta} = E_{\tilde{p}(x)}\left[\frac{\partial Q(x;\theta)}{\partial \theta}\right] - E_{p(x;\theta)}\left[\frac{\partial Q(x;\theta)}{\partial \theta}\right] = 0$$

Maximum Entropy

Expectation under empirical distribution $\tilde{p}(x) = \frac{1}{N}\sum_{i=1}^{N} 1(x = x_i)$

Expectation under model distribution $p(x;\theta)$

# Training of UGMs - overview

- Roughly speaking, two types of approximate methods
- Gradient methods
  - Make explicit use of the gradient: Gradient descent, conjugate gradient, L-BFGS.
  - Stochastic approximation (SA)
  - Stochastic maximum likelihood (SML)
  - Persistent contrastive divergence (PCD)
- Lower bound methods
  - Generalized iterative scaling (GIS)
  - Improved iterative scaling (IIS)
  - Mostly studied in the context of maximum entropy (maxent) parameter estimation of log-linear models.
- In practice the gradient methods are shown to be much faster than the lower bound methods

# Comparison on learning CRFs

- Div: the relative entropy between the fitted model and the training data
- Iter: Iteration number
- Evals: the number of calculating log-likelihood and gradient
- Time: the total time.

| Dataset | Method | Div. | Iter | Evals | Time (secs) |
|---|---|---|---|---|---|
| rules | gis | $5.19\times10^{-2}$ | 1201 | 1202 | 23.04 |
| | iis | $5.14\times10^{-2}$ | 923 | 924 | 42.48 |
| | steepest ascent | $5.13\times10^{-2}$ | 212 | 331 | 6.16 |
| | conjugate gradient (fr) | $5.07\times10^{-2}$ | 74 | 196 | 3.74 |
| | conjugate gradient (prp) | $5.08\times10^{-2}$ | 63 | 154 | 2.87 |
| | limited memory variable metric | $5.07\times10^{-2}$ | 70 | 76 | 1.44 |
| lex | gis | $1.61\times10^{-3}$ | 370 | 371 | 36.29 |
| | iis | $1.52\times10^{-3}$ | 241 | 242 | 102.18 |
| | steepest ascent | $3.47\times10^{-3}$ | 1041 | 1641 | 139.10 |
| | conjugate gradient (fr) | $1.39\times10^{-3}$ | 166 | 453 | 39.03 |
| | conjugate gradient (prp) | $1.62\times10^{-3}$ | 150 | 382 | 32.46 |
| | limited memory variable metric | $1.49\times10^{-3}$ | 136 | 143 | 17.25 |
| summary | gis | $1.83\times10^{-3}$ | 1446 | 1447 | 125.46 |
| | iis | $1.07\times10^{-3}$ | 626 | 627 | 208.22 |
| | steepest ascent | $2.64\times10^{-3}$ | 1163 | 3503 | 227.30 |
| | conjugate gradient (fr) | $1.01\times10^{-4}$ | 175 | 948 | 60.91 |
| | conjugate gradient (prp) | $7.30\times10^{-4}$ | 93 | 428 | 27.81 |
| | limited memory variable metric | $3.98\times10^{-5}$ | 81 | 89 | 10.38 |
| shallow | gis | $3.57\times10^{-2}$ | 3428 | 3429 | 27103.62 |
| | iis | $3.50\times10^{-2}$ | 3216 | 3217 | 71053.24 |
| | steepest ascent † | — | — | — | — |
| | conjugate gradient (fr) | $2.91\times10^{-2}$ | 1094 | 6056 | 46958.87 |
| | conjugate gradient (prp) | $4.13\times10^{-2}$ | 421 | 2170 | 16477.84 |
| | limited memory variable metric | $3.26\times10^{-2}$ | 429 | 444 | 3408.30 |


- T. Tieleman, “Training restricted boltzmann machines using approximations to the likelihood gradient”, ICML 2008.
- R. Malouf, “A comparison of algorithms for maximum entropy parameter estimation”, in Proc. Conference on Natural Language Learning (CoNLL), 2002.

# Training of log-linear models

$$p(x;\theta) = \frac{1}{Z(\theta)} exp\left[\sum_C \theta_C^T f_C(x)\right]$$ where $C$ indexes the cliques.

$$\frac{\partial l(\theta)}{\partial \theta_C} = E_{\tilde{p}(x)}[f_C(x)] - E_{p(x;\theta)}[f_C(x)] = 0$$ Statistics matching

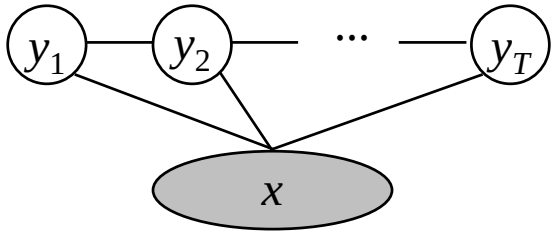
Empirical statistics of features

Expected statistics of features

- $l(\theta)$ is convex in $\theta$, so it has a unique global maximum which we can find using gradient-based optimizers. ☺
- The exact calculation of the gradient is intractable in general, involving high-dimensional integration. ☹

# Training of log-linear models - example

$$p(y_{1:T} \mid x) \propto \exp\left\{\sum_{t=1}^{T-1}\sum_{i}\lambda_i f_i(y_t, y_{t+1}, x, t) + \sum_{t=1}^{T}\sum_{j}\mu_j f_j(y_t, x, t)\right\}$$




- Maximum conditional likelihood (MCL)

$$\frac{\partial p(y_{1:T}|x;\theta)}{\partial \mu_j} = \sum_{t=1}^{T} f_j(y_t, x, t) - E_{p(y|x;\theta)}\left[\sum_{t=1}^{T} f_j(y_t, x, t)\right]$$

Statistics matching

$$= \sum_{t=1}^{T} f_j(y_t, x, t) - \sum_{t=1}^{T} E_{p(y_t|x;\theta)}[f_j(y_t, x, t)]$$

- The above gradient involves only one training instance $y_{1:T}|x$.
- The gradient of scaled conditional likelihood is sum of gradients for all training instances.

# Training of partially observed UGMs

$$p(x,h;\theta)=\frac{1}{Z(\theta)}\exp[Q(x,h;\theta)]$$

Normalization constant:

$$Z(\theta)=\sum_{x,h}\exp[Q(x,h;\theta)]$$

- Maximum likelihood (ML) training

Scaled log-likelihood of observed $\{x_i, i=1,\cdots,N\}$

$$l(\theta)\triangleq\frac{1}{N}\sum_{i=1}^{N}logp(x_i;\theta)=\left[\frac{1}{N}\sum_{i=1}^{N}log\sum_{h}\exp[Q(x_i,h;\theta)]\right]-logZ(\theta)$$

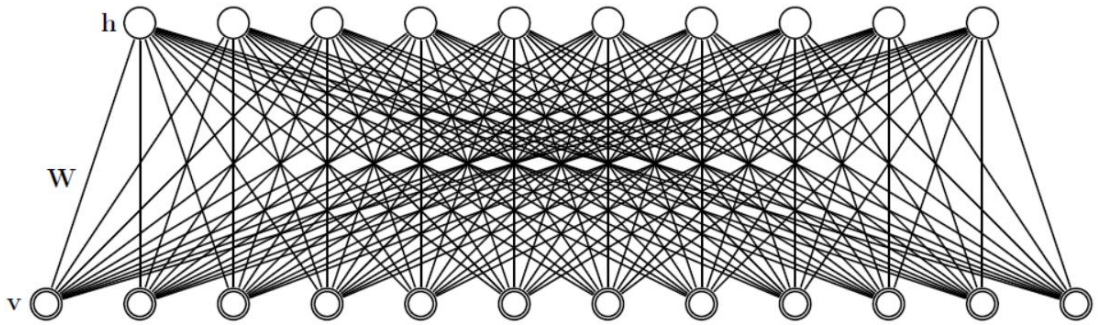
$$\frac{\partial l(\theta)}{\partial\theta}=E_{\tilde{p}(x)p(h|x)}\left[\frac{\partial Q(x,h;\theta)}{\partial\theta}\right]-E_{p(x,h;\theta)}\left[\frac{\partial Q(x,h;\theta)}{\partial\theta}\right]=0$$

Expectation under empirical distribution $\tilde{p}(x)p(h|x)$

Expectation under model distribution $p(x,h;\theta)$

# Training of partially observed UGMs - example

- RBM is a two-layer MRF
  - Binary visible variables $v \in \{0,1\}^D$
  - Binary hidden variables $h \in \{0,1\}^F$
  - $\theta = \{W, b, a\}$




$$p(v,h;\theta) = \frac{1}{Z(\theta)} exp[Q(v,h;\theta)]$$

$$Q(v,h;\theta) = v^T W h + b^T v + a^T h$$

$$= \sum_{i=1}^{D}\sum_{j=1}^{F} v_i W_{ij} h_j + \sum_{i=1}^{D} b_i v_i + \sum_{j=1}^{F} a_j h_j$$

$$\begin{cases} \frac{\partial l(\theta)}{\partial W} = E_{p_{emp}}[vh^T] - E_{p_{model}}[vh^T] \\ \frac{\partial l(\theta)}{\partial a} = E_{p_{emp}}[h] - E_{p_{model}}[h] \\ \frac{\partial l(\theta)}{\partial b} = E_{p_{emp}}[v] - E_{p_{model}}[v] \end{cases}$$

# Training of UGMs in general

gradient = empirical expectation – model expectation

1. Approximate the model expectations using Monte Carlo sampling.
   - We can use MCMC to generate the samples, but running MCMC to convergence at each step of the inner loop would be extremely slow.
   - Fortunately, it was shown by Younes (1989) that we can start the MCMC chain at its previous value, and just take a few steps.
2. We can combine this with stochastic gradient descent (SGD), which takes samples from the empirical distribution.

Both two ideas/tricks essentially follows in the framework of Stochastic Approximation (SA).


- Robbins and Monro (1951). A stochastic approximation method. Ann. Math. Stat.
- L. Younes, “Parametric inference for imperfectly observed gibbsian fields,” Probability Theory and Related Fields, 1989.

# Stochastic Approximation (SA)

Problem: The objective is to find a solution $\theta$ to $E_{Y\sim f(\cdot;\,\theta)}[H(Y;\theta)] = \alpha$,

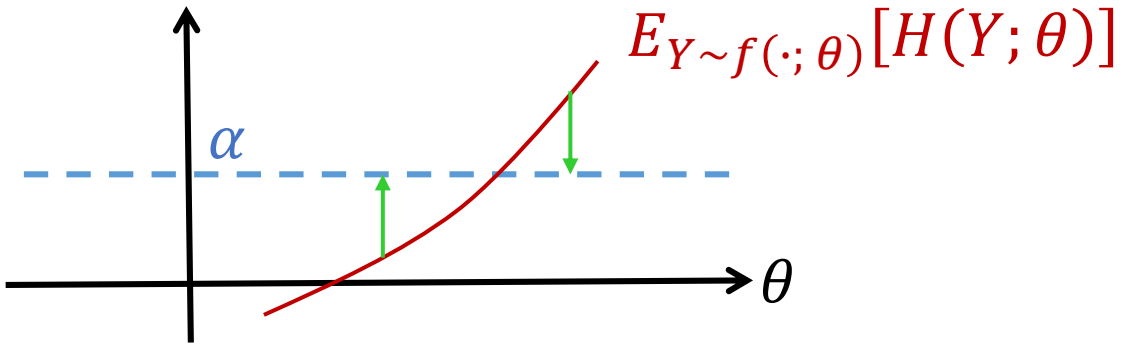
where $\theta \in R^d$, noisy observation $H(Y;\theta) \in R^d$

Method:

(1) Sampling: Generate $Y_t \sim K(Y_{t-1}, \cdot\,; \theta_{t-1})$, a Markov transition kernel that admits $f(\cdot;\,\theta_{t-1})$ as the invariant distribution.

(2) Updating: Set $\theta_t = \theta_{t-1} + \gamma_t\{H(Y_t;\theta_{t-1}) - \alpha\}$

$e.g.\ \gamma_t = \dfrac{1}{t_0 + t}$




- Robbins and Monro (1951). A stochastic approximation method. Ann. Math. Stat.
- Chen (2002), Stochastic Approximation and Its Applications, Kluwer Academic Publishers.

# Training of partially observed UGMs – SA algorithm

Training data : Observed $\{x_i, i = 1, \cdots, N\}$

$$\frac{\partial l(\theta)}{\partial \theta} = E_{\tilde{p}(v,z;\theta)}\left[\frac{\partial Q(v,z;\theta)}{\partial \theta}\right] - E_{p(x,h;\theta)}\left[\frac{\partial Q(x,h;\theta)}{\partial \theta}\right] = 0$$

$$Y = \begin{pmatrix} v \\ z \\ x \\ h \end{pmatrix} \sim f(\cdot;\, \theta) = \tilde{p}(v,z;\theta)p(x,h;\theta) \qquad \tilde{p}(v,z) = \frac{1}{N}\sum_{i=1}^{N} 1(v = x_i) \cdot p(z|v)$$

(1) Initialize $\theta_0$ randomly;

(2) For iteration $t = 1, \cdots,$ do

- Draw a empirical minibatch of size $B$ $\{(v^{(i)}, z^{(i)}), i = 1, \cdots, B\}$ according to $\tilde{p}(v, z; \theta_{t-1})$;

  Draw a Monte Carlo minibatch $\{(x^{(i)}, h^{(i)}), i = 1, \cdots, B\}$ by continuously taking $B$ steps using a Markov transition kernel that admits $p(x, h; \theta_{t-1})$ as the invariant distribution.
- Updating:

$$\theta_t = \theta_{t-1} + \gamma_t \frac{1}{B}\left\{\left(\sum_{i=1}^{B} \frac{\partial Q(v^{(i)}, z^{(i)}; \theta)}{\partial \theta}\right)\Bigg|_{\theta=\theta_{t-1}} - \left(\sum_{i=1}^{B} \frac{\partial Q(x^{(i)}, h^{(i)}; \theta)}{\partial \theta}\right)\Bigg|_{\theta=\theta_{t-1}}\right\}$$

# Connection of SA with other gradient methods

- Robbins and Monro 1951.
- aka Stochastic Maximum Likelihood (SML), (Younes 1989).
- This was independently discovered by Tieleman in 2008, who called it persistent contrastive divergence (PCD).
- In regular contrastive divergence (CD), proposed by Hinton 2002, we restart the Markov chain at the training data rather than at the previous state. This will not converge to the MLE.
- “Clearly, the widely used practice of CD1 learning is a rather poor “substitute” for maximum likelihood learning. “ (Salakhutdinov phd thesis 2009).

# GIS and IIS for learning log-linear models

$$p(x;\lambda) = \frac{1}{Z}\exp\left\{\sum_{i=1}^{F}\lambda_i f_i(x)\right\}$$

$$\tilde{p}[f_i] - \sum_x p(x) f_i(x) e^{\Delta\lambda_i f_\#(x)} = 0 \qquad f_\#(x) = \sum_{i=1}^{F} f_i(x)$$

- Generalized iterative scaling (GIS)
  - Introduce an extra feature $f_{F+1}(x) = S - \sum_{i=1}^{F} f_i(x)$
  - Then we have $f_\#(x) = \sum_{i=1}^{F+1} f_i(x) = S$ is a constant.

$$\Delta\lambda_i = \frac{1}{S}\log\frac{\tilde{p}[f_i]}{p[f_i]} \qquad i = 1, \ldots, F, F+1$$

- Improved iterative scaling (IIS)

$$\sum_{m=1}^{M} \sum_{\{x|f_\#(x)=m\}} p(x) f_i(x) \beta_i^m = \tilde{p}[f_i] \qquad \beta_i = e^{\Delta\lambda_i}$$

Use Newton Method to solve the polynomial

# Summary

- **General and basic concepts of UGMs**
  1. Semantics of DGMs and UGMs
     - RBMs, DBNs, CRFs, TRFs
  2. Exact inference - variable elimination
  3. Approximate inference – variational
  4. Approximate inference – Monte Carlo
  5. Learning

# Further study

- Daphne Koller, Nir Friedman. “Probabilistic graphical models : principles and techniques”. MIT Press, 2009.
  - Detailed, 1231 pages.
- R. G. Cowell, A. P. Dawid, S. L. Lauritzen and D. J. Spiegelhalter. "Probabilistic Networks and Expert Systems". Springer-Verlag. 1999.
  - One of the best book available, although the treatment is restricted to exact inference.
- J. Pearl. "Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference". Morgan Kaufmann. 1988.
  - The book that got it all started! A very insightful book, still relevant today.
- S. Lauritzen. "Graphical Models". Oxford. 1996.
  - The definitive mathematical exposition of the theory of graphical models.
- M. I. Jordan (ed). "Learning in Graphical Models". MIT Press. 1999.
  - Loose collection of papers on machine learning, many related to graphical models. One of the few books to discuss approximate inference.
- Christopher M. Bishop. “Pattern Recognition and Machine Learning”. Springer 2006.
  - Comprehensive, good reference.
- D. J. MacKay. “Information Theory, Inference, and Learning Algorithms”. Cambridge Univ. Press, 2003.
  - Information theory, coding.
- Kevin P. Murphy. “Machine Learning: A Probabilistic Perspective”. MIT Press, 2012.
  - Detailed, 1098 pages.
  - Matlab code.

# Thanks for your attention !

Thanks to my collaborators and students :

Bin Wang, Zhiqiang Tan, Hongyu Xiang, Yinpei Dai