

# Learning in Graphical Models

Andrea Passerini  
passerini@disi.unitn.it

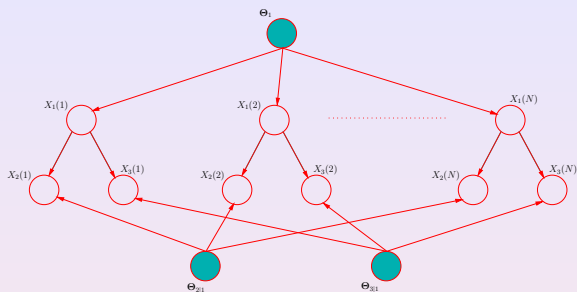
Machine Learning

## Parameter estimation

- We assume the structure of the model is given
- We are given a dataset of examples  $\mathcal{D} = \{\mathbf{x}(1), \dots, \mathbf{x}(N)\}$
- Each example  $\mathbf{x}(i)$  is a configuration for *all* (complete data) or *some* (incomplete data) variables in the model
- We need to estimate the parameters of the model (conditional probability distributions) from the data
- The simplest approach consists of learning the parameters maximizing the *likelihood* of the data:

$$\theta^{\max} = \operatorname{argmax}_{\theta} p(\mathcal{D}|\theta) = \operatorname{argmax}_{\theta} \mathcal{L}(\mathcal{D}, \theta)$$

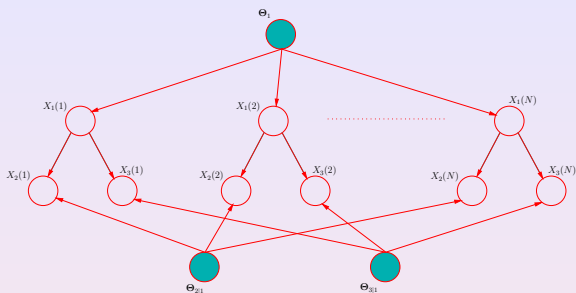
# Learning Bayesian Networks



Maximum likelihood estimation, complete data

$$p(\mathcal{D}|\theta) = \prod_{i=1}^N p(\mathbf{x}(i)|\theta) \quad \text{examples independent given } \theta$$
$$= \prod_{i=1}^N \prod_{j=1}^m p(\mathbf{x}_j(i)|pa_j(i), \theta) \quad \text{factorization for BN}$$

# Learning Bayesian Networks



Maximum likelihood estimation, complete data

$$p(\mathcal{D}|\boldsymbol{\theta}) = \prod_{i=1}^N \prod_{j=1}^m p(\mathbf{x}_j(i) | \text{pa}_j(i), \boldsymbol{\theta}) \quad \text{factorization for BN}$$

$$= \prod_{i=1}^N \prod_{j=1}^m p(\mathbf{x}_j(i) | \text{pa}_j(i), \boldsymbol{\theta}_{X_j | \text{pa}_j}) \quad \text{disjoint CPD parameters}$$

## Maximum likelihood estimation, complete data

- The parameters of each CPD can be estimated independently:

$$\theta_{X_j|Pa_j}^{\max} = \operatorname{argmax}_{\theta_{X_j|Pa_j}} \underbrace{\prod_{i=1}^N \rho(\mathbf{x}_j(i)|pa_j(i), \theta_{X_j|Pa_j})}_{\mathcal{L}(\theta_{X_j|Pa_j}, \mathcal{D})}$$

- A discrete CPD  $P(X|\mathbf{U})$ , can be represented as a table, with:
  - a number of rows equal to the number  $Val(X)$  of configurations for  $X$
  - a number of columns equal to the number  $Val(\mathbf{U})$  of configurations for its parents  $\mathbf{U}$
  - each table entry  $\theta_{x|\mathbf{u}}$  indicating the probability of a specific configuration of  $X = x$  and its parents  $\mathbf{U} = \mathbf{u}$

# Learning graphical models

## Maximum likelihood estimation, complete data

- Replacing  $p(x(i)|pa(i))$  with  $\theta_{x(i)|\mathbf{u}(i)}$ , the local likelihood of a single CPD becomes:

$$\begin{aligned}\mathcal{L}(\theta_{X|Pa}, \mathcal{D}) &= \prod_{i=1}^N p(x(i)|pa(i), \theta_{X|Pa_j}) \\ &= \prod_{i=1}^N \theta_{x(i)|\mathbf{u}(i)} \\ &= \prod_{\mathbf{u} \in \text{Val}(\mathbf{U})} \left[ \prod_{x \in \text{Val}(X)} \theta_{x|\mathbf{u}}^{N_{\mathbf{u},x}} \right]\end{aligned}$$

where  $N_{\mathbf{u},x}$  is the number of times the specific configuration  $X = x$ ,  $\mathbf{U} = \mathbf{u}$  was found in the data

## Maximum likelihood estimation, complete data

- A column in the CPD table contains a multinomial distribution over values of  $X$  for a certain configuration of the parents  $\mathbf{U}$
- Thus each column should sum to one:  $\sum_x \theta_{x|\mathbf{u}} = 1$
- Parameters of different columns can be estimated independently
- For each multinomial distribution, zeroing the gradient of the maximum likelihood and considering the normalization constraint, we obtain:

$$\theta_{x|\mathbf{u}}^{max} = \frac{N_{\mathbf{u},x}}{\sum_x N_{\mathbf{u},x}}$$

- The maximum likelihood parameters are simply the fraction of times in which the specific configuration was observed in the data

## Adding priors

- ML estimation tends to overfit the training set
- Configuration not appearing in the training set will receive zero probability
- A common approach consists of combining ML with a prior probability on the parameters, achieving a maximum-a-posteriori estimate:

$$\theta^{\max} = \operatorname{argmax}_{\theta} p(\mathcal{D}|\theta)p(\theta)$$



## Dirichlet priors

- The conjugate (read natural) prior for a multinomial distribution is a Dirichlet distribution with parameters  $\alpha_{x|\mathbf{u}}$  for each possible value of  $x$
- The resulting maximum-a-posteriori estimate is:

$$\theta_{x|\mathbf{u}}^{\max} = \frac{N_{\mathbf{u},x} + \alpha_{x|\mathbf{u}}}{\sum_x (N_{\mathbf{u},x} + \alpha_{x|\mathbf{u}})}$$

- The prior is like having observed  $\alpha_{x|\mathbf{u}}$  imaginary samples with configuration  $X = x, \mathbf{U} = \mathbf{u}$

## Incomplete data

- With incomplete data, some of the examples miss evidence on some of the variables
- Counts of occurrences of different configurations cannot be computed if not all data are observed
- The full Bayesian approach of integrating over missing variables is often intractable in practice
- We need approximate methods to deal with the problem

## E-M for Bayesian nets in a nutshell

- Sufficient statistics (counts) cannot be computed (missing data)
- Fill-in missing data inferring them using current parameters (solve inference problem to get *expected* counts)
- Compute parameters maximizing likelihood (or posterior) of such expected counts
- Iterate the procedure to improve quality of parameters

## Expectation-Maximization algorithm

**e-step** Compute the expected sufficient statistics for the complete dataset, with expectation taken wrt the joint distribution for  $\mathbf{X}$  conditioned of the current value of  $\theta$  and the known data  $\mathcal{D}$ :

$$E_{p(\mathbf{X}|\mathcal{D},\theta)}[N_{ijk}] = \sum_{l=1}^n p(X_i(l) = x_k, Pa_i(l) = pa_j | \mathbf{X}_l, \theta)$$

- If  $X_i(l)$  and  $Pa_i(l)$  are observed for  $\mathbf{X}_l$ , it is either zero or one
- Otherwise, run Bayesian inference to compute probabilities from observed variables

# Learning with missing data: Expectation-Maximization

## Expectation-Maximization algorithm

**m-step** compute parameters maximizing likelihood of the complete dataset  $D_c$  (using expected counts):

$$\theta^* = \operatorname{argmax}_{\theta} p(D_c | \theta)$$

which for each multinomial parameter evaluates to:

$$\theta_{ijk}^* = \frac{E_{p(\mathbf{x}|D, \theta)}[N_{ijk}]}{\sum_{k=1}^{r_i} E_{p(\mathbf{x}|D, \theta)}[N_{ijk}]}$$

## Note

ML estimation can be replaced by maximum a-posteriori (MAP) estimation giving:

$$\theta_{ijk}^* = \frac{\alpha_{ijk} + E_{p(\mathbf{x}|D, \theta, S)}[N_{ijk}]}{\sum_{k=1}^{r_i} (\alpha_{ijk} + E_{p(\mathbf{x}|D, \theta, S)}[N_{ijk}])}$$

## Approaches

**constraint-based** test conditional independencies on the data and construct a model satisfying them

**score-based** assign a score to each possible structure, define a search procedure looking for the structure maximizing the score

**model-averaging** assign a prior probability to each structure, and average prediction over all possible structures weighted by their probabilities (full Bayesian, intractable)

# Appendix: Learning the structure

## Bayesian approach

- Let  $\mathcal{S}$  be the space of possible structures (DAGS) for the domain  $\mathbf{X}$ .
- Let  $\mathcal{D}$  be a dataset of observations
- Predictions for a new instance are computed marginalizing over both structures and parameters:

$$\begin{aligned} p(\mathbf{X}_{N+1}|\mathcal{D}) &= \sum_{S \in \mathcal{S}} \int_{\theta} P(\mathbf{X}_{N+1}, S, \theta | \mathcal{D}) d\theta \\ &= \sum_{S \in \mathcal{S}} \int_{\theta} P(\mathbf{X}_{N+1} | S, \theta, \mathcal{D}) P(S, \theta | \mathcal{D}) d\theta \\ &= \sum_{S \in \mathcal{S}} \int_{\theta} P(\mathbf{X}_{N+1} | S, \theta) P(\theta | S, \mathcal{D}) P(S | \mathcal{D}) d\theta \\ &= \sum_{S \in \mathcal{S}} P(S | \mathcal{D}) \int_{\theta} P(\mathbf{X}_{N+1} | S, \theta) P(\theta | S, \mathcal{D}) d\theta \end{aligned}$$

## Problem

Averaging over all possible structures is too expensive

## Model selection

- Choose a *best* structure  $S^*$  and assume  $P(S^*|\mathcal{D}) = 1$
- Approaches:
  - Score-based:
    - Assign a score to each structure
    - Choose  $S^*$  to maximize the score
  - Constraint-based:
    - Test conditional independencies on data
    - Choose  $S^*$  that satisfies these independencies



## Structure scores

- Maximum-likelihood score:

$$S^* = \operatorname{argmax}_{S \in \mathcal{S}} p(\mathcal{D} | S)$$

- Maximum-a-posteriori score:

$$S^* = \operatorname{argmax}_{S \in \mathcal{S}} p(\mathcal{D} | S) p(S)$$

## Maximum likelihood approximation

- The easiest solution is to approximate  $P(\mathcal{D}|\mathcal{S})$  with the maximum-likelihood score *over the parameters*:

$$P(\mathcal{D}|\mathcal{S}) \approx \max_{\theta} P(\mathcal{D}|\mathcal{S}, \theta)$$

- Unfortunately, this boils down to adding a connection between two variables if their empirical mutual information over the training set is non-zero (proof omitted)
- Because of noise, empirical mutual information between any two variables is almost never exactly zero  $\Rightarrow$  fully connected network

# Computing $P(\mathcal{D}|S) \equiv P_S(\mathcal{D})$ : Bayesian-Dirichlet scoring

## Simple case: setting

- $X$  is a single variable with  $r$  possible realizations ( $r$ -faced die)
- $S$  is a single node
- Probability distribution is a multinomial with Dirichlet priors  $\alpha_1, \dots, \alpha_r$ .
- $\mathcal{D}$  is a sequence of  $N$  realizations (die tosses)

# Computing $P_S(\mathcal{D})$ : Bayesian-Dirichlet scoring

## Simple case: approach

- Sort  $\mathcal{D}$  according to outcome:

$$\mathcal{D} = \{x^1, x^1, \dots, x^1, x^2, \dots, x^2, \dots, x^r, \dots, x^r\}$$

- Its probability can be decomposed as:

$$P_S(\mathcal{D}) = \prod_{t=1}^N P_S(X(t) | \underbrace{X(t-1), \dots, X(1)}_{\mathcal{D}(t-1)})$$

- The prediction for a new event given the past is:

$$P_S(X(t+1) = x^k | \mathcal{D}(t)) = E_{P_S(\theta | \mathcal{D}(t))}[\theta_k] = \frac{\alpha_k + N_k(t)}{\alpha + t}$$

where  $N_k(t)$  is the number of times we have  $X = x^k$  in the first  $t$  examples in  $\mathcal{D}$

# Computing $P_S(\mathcal{D})$ : Bayesian-Dirichlet scoring

## Simple case: approach

$$\begin{aligned} P_S(\mathcal{D}) &= \frac{\alpha_1}{\alpha} \frac{\alpha_1 + 1}{\alpha + 1} \cdots \frac{\alpha_1 + N_1 - 1}{\alpha + N_1 - 1} \\ &\quad \cdot \frac{\alpha_2}{\alpha + N_1} \frac{\alpha_2 + 1}{\alpha + N_1 + 1} \cdots \frac{\alpha_2 + N_2 - 1}{\alpha + N_1 + N_2 - 1} \cdots \\ &\quad \cdot \frac{\alpha_r}{\alpha + N_1 + \cdots + N_{r-1}} \cdots \frac{\alpha_r + N_r - 1}{\alpha + N - 1} \\ &= \frac{\Gamma(\alpha)}{\Gamma(\alpha + N)} \prod_{k=1}^r \frac{\Gamma(\alpha_k + N_k)}{\alpha_k} \end{aligned}$$

where we used the Gamma function ( $\Gamma(x + 1) = x\Gamma(x)$ ):

$$\alpha(1 + \alpha) \cdots (N - 1 + \alpha) = \frac{\Gamma(N + \alpha)}{\Gamma(\alpha)}$$

# Computing $P_S(\mathcal{D})$ : Bayesian-Dirichlet scoring

## General case

$$P_S(\mathcal{D}) = \prod_i \prod_j \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + N_{ij})} \prod_{k=1}^r \frac{\Gamma(\alpha_{ijk} + N_{ijk})}{\alpha_{ijk}}$$

where

- $i \in \{1, \dots, n\}$  ranges over nodes in the network
- $j \in \{1, q_i\}$  ranges over configurations of  $X_i$ 's parents
- $k \in \{1, r_i\}$  ranges over states of  $X_i$

## Note

The score is **decomposable**: it is the product of independent scores associated with the distribution of each node in the net

## Approach

- Discrete search problem: NP-hard for nets whose nodes have at most  $k > 1$  parents.
- Heuristic search strategies employed:
  - Search space: set of DAGs
  - Operators: add, remove, reverse one arc
  - Initial structure: e.g. random, fully disconnected, ...
  - Strategies: hill climbing, best first, simulated annealing

## Note

Decomposable scores allow to recompute local scores only for a single move