Universität Stuttgart
IPVS – Institute for Parallel and Distributed Systems

Analytic Computing

Advanced Topics in Machine Learning
8 Learning - Part 1

Prof. Dr. Steffen Staab
Dr. Rafika Boutalbi
Zihao Wang
https://www.ipvs.uni-stuttgart.de/departments/ac/

Learning Objectives

Learning Tasks: Queries, Prediction, Knowledge discovery

Sufficient statistics: multinomial, Gaussian

Data Fragmentation and Overfitting

Maximum likelihood for complete observations of BNs

Maximum likelihood for complete observations of MNs

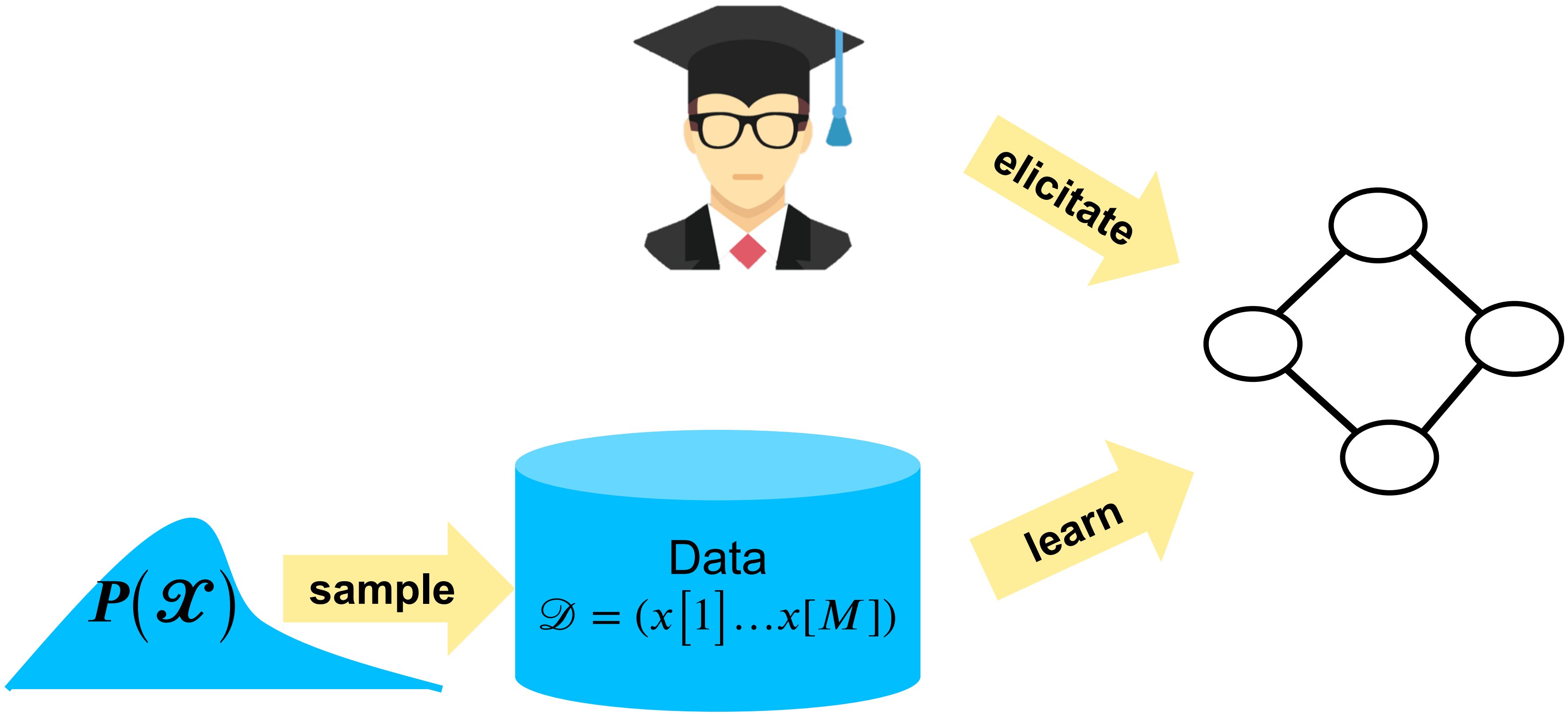
Log-linear models, feature functions

Disclaimer

Figures and examples not marked otherwise are taken from the book by Koller & Friedman

1 Learning Tasks in PGMs

Learning PGMs



true distribution (unknown)

Network

Why PGM Learning?

- Predictions of structured objects
  - sequences, graphs, trees
  - exploit correlations between several predicted variables
- Incorporate prior knowledge into model
  - Expert knowledge!
- Knowledge discovery
  - Explainability!
- One model, many tasks!
  - From cause to conclusion
  - from conclusion to plausible cause

Classification of challenges

- **Network structure known**
  - Induce only the factors
- **Network structure unknown**
  - Induce structure
  - Induce factors

- **All** random variables **completely observed**
- **All** random variables **observed**, but not completely
- **Some** random variables **completely observed**
  - others: **Latent Variables**
- **Some** random variables **observed**
  - others: **Latent Variables**

PGM Learning Task 1: Queries

Goal: Answer general probabilistic queries about new instances

- Targeted quality: Generalization
  - Metric: Test set likelihood: $P(\mathscr{D}' ; \mathscr{M})$

PGM Learning Task 2: Prediction

Goal: Specific prediction task on new instances

- Predict target variables $\boldsymbol{Y}$ from observed variables $\boldsymbol{X}$
  - Typically MAP Assignment
    - examples: image segmentation, speech recognition
- Targeted quality: correct assignment
  - conditional likelihood $\prod_m P(\boldsymbol{y}[m] \mid \boldsymbol{x}[m] \,;\, \mathscr{M})$
  - model evaluated on „true“ assignment over test data (“gold standard“)

PGM Learning Task 3: Knowledge Discovery

Goal: Knowledge discovery of $\mathscr{M}^*$

- Distinguish direct vs. indirect dependencies
- Possibly directionality of edges (causality!)
- Presence and location of hidden variables
  - e.g. confounding variables / factors
- Trained using likelihood
  - does not reflect structural accuracy
- Evaluate by comparing to prior knowledge

Overfitting

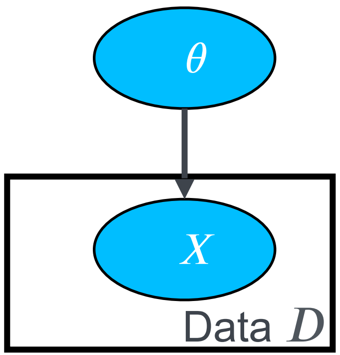
- Same problems as discussed in ML lecture
- Trade-off
  - more complex model better fits the training data
  - more complex model tends to overfit more
- Two kinds of overfitting
  - parameter overfitting
  - structure overfitting

regularization / penalizing model complexity

Question 1: Learning Tasks in PGMs

2 Maximum Likelihood for Bayesian Networks with Complete Observation of All Variables

Remember: MLE for Binary Coin Toss



- $\mathscr{X} = (X_1 \ldots X_m)$ with $\mathrm{val}\left(X_i\right) = \{0,1\}$, X_i are iid
- Which θ best explains $\mathscr{D} = \{x\left[1\right] \ldots x[M]\}$?
  - k times head up, $M - k$ times head down
- Likelihood:
  $L\left(\mathscr{D};\theta\right) = \prod_m P(x[m];\theta) = \theta^k (1-\theta)^{M-k}$
- Log-likelihood:
  $\log L\left(\mathscr{D};\theta\right) = k \log\theta + (M-k)\log(1-\theta)$
- Maximize by setting derivative to 0

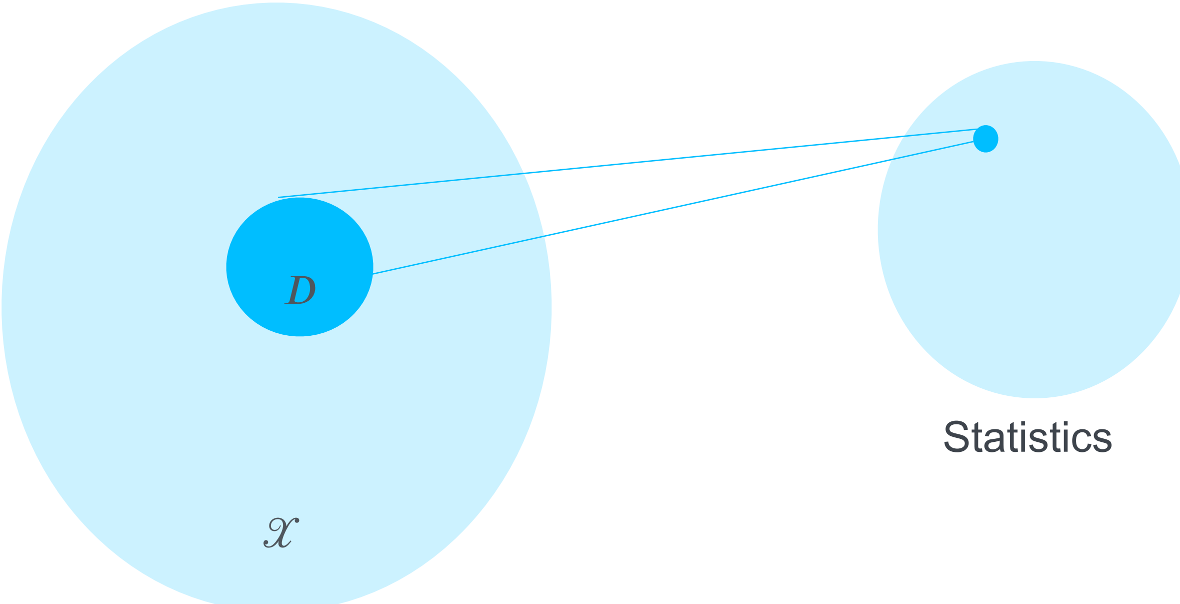
$$\frac{d \log L\left(\mathscr{D};\theta\right)}{d\theta} = \frac{k}{\theta} - \frac{M-k}{1-\theta} \overset{!}{=} 0$$

$$k(1-\theta) = (M-k)\theta$$

$$k - k\theta = M\theta - k\theta$$

$$\theta = \frac{k}{M}$$

Sufficient statistics



$$s(x) = \begin{cases} (1\ 0), & \text{for head up} \\ (0\ 1), & \text{for head down} \end{cases}$$

is sufficient statistics for binary coin toss

A function $s(\mathscr{D})$ is a sufficient statistics from instances to a vector in $\mathbb{R}^k$ if for any two datasets $\mathscr{D}$ and $\mathscr{D}'$ and any $\theta \in \Theta$ we have

$$\sum_{x \in \mathscr{D}} s(x) = \sum_{x \in \mathscr{D}'} s(x) \Longrightarrow L\left(\mathscr{D}; \theta\right) = L\left(\mathscr{D}'; \theta\right)$$

Sufficient Statistic for Multinomial

- Example: Bag of Words

  $\mathscr{D} = \{\text{dog}, \text{ cat}, \text{ dog}, \text{ bee}, \text{lion}, \text{dog}, \text{dog}, \text{cat}\}$

- For a dataset $\mathscr{D}$ over variable X with $\left|\text{val}(X)\right| = K$, the sufficient statistics are counts $< k_1, \ldots, k_K >$ where k_i is the number of times that $x[\ldots] = x^i$ in $\mathscr{D}$
  - $s\left(x^i\right) = (0\ldots.0\ 1\ 0\ \ldots 0)$, with 1 in i-th place
  - $L\left(\mathscr{D};\theta\right) = \prod_i \theta_i^{k_i}$
  - MLE: $\hat{\theta}_i = \dfrac{k_i}{M}$

Sufficient Statistic for Gaussian

- Gaussian distribution:

$$\boldsymbol{P}(X) \sim N\left(\mu, \sigma^2\right)$$

$$P(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} =$$

$$= \frac{1}{\sqrt{2\pi\sigma}} e^{\left(-x^2\frac{1}{2\sigma^2}+x\frac{\mu}{\sigma^2}-\frac{\mu^2}{2\sigma^2}\right)}$$

- Sufficient statistics for Gaussian:

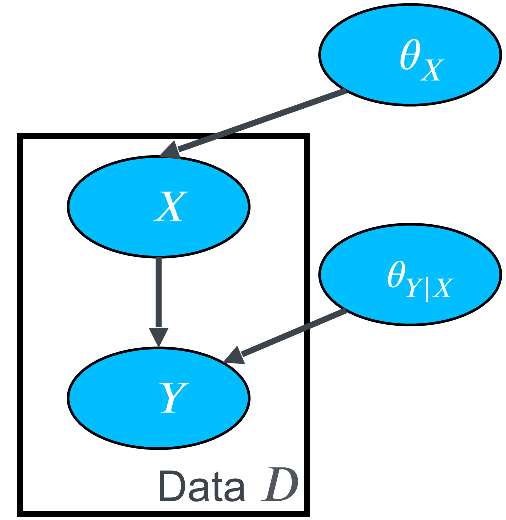
$$s(x) = < 1, x, x^2 >$$

Remember: Gaussian MLE

$$\hat{\mu} = \frac{1}{M}\sum_{m=1}^{M} x[m]$$

$$\hat{\sigma}^2 = \frac{1}{M}\sum_{m=1}^{M}\left(x[m] - \hat{\mu}\right)^2$$

MLE for Bayesian Networks



Parameters

$$\{\theta_x : x \in \mathrm{val}(X)\}$$

$$\{\theta_{y|x} : (y, x) \in \mathrm{val}(Y) \times \mathrm{val}(X)\}$$

$$L(\mathscr{D}; \Theta) = \prod_{m=1}^{M} P(x[m], y[m]; \Theta) = \prod_{m=1}^{M} P(x[m]; \Theta) P(y[m] \mid x[m]; \Theta) =$$

$$= \left(\prod_{m=1}^{M} P(x[m]; \Theta)\right)\left(\prod_{m=1}^{M} P(y[m] \mid x[m]; \Theta)\right)$$

$$= = \underbrace{\left(\prod_{m=1}^{M} P(x[m]; \theta_X)\right)}_{\text{optimized like a multinomial}}\underbrace{\left(\prod_{m=1}^{M} P(y[m] \mid x[m]; \theta_{Y|X})\right)}_{\text{also optimized like a multinomial}}$$

local likelihoods

Likelihood for Bayesian Network in General

$$L(\mathscr{D};\Theta) = \prod_{m=1}^{M} P(x[m];\Theta) =$$

$$= \prod_{m=1}^{M} \prod_{i} P(x_i[m] \mid \underbrace{\mathrm{pa}_{X_i}[m]}_{\text{value assignments to parents of } X_i \text{ in data item } x[m]};\Theta) =$$

$$= \prod_{i} \prod_{m=1}^{M} P(x_i[m] \mid \mathrm{pa}_{X_i}[m];\Theta) =$$

$$= \prod_{i} L_i(\mathscr{D};\Theta_i)$$

product of local likelihoods

with

$$L_i(\mathscr{D};\Theta_i) = P(x_i[m] \mid \mathrm{pa}_{X_i}[m];\theta_{X_i \mid \mathrm{pa}_{X_i}})$$

If $\theta_{X_i|\mathrm{pa}_{X_i}}$ are disjoint,
then MLE can be computed by maximizing each local likelihood separately

Likelihood for Table CPDs

Given:

- Data $\mathscr{D} = \left\{ \boldsymbol{x}[1] \ldots \boldsymbol{x}[M] \right\} \sim \mathbf{P}(\mathscr{X}),\ \mathscr{X} = (\mathrm{X}_1 \times \ldots \times X_l)$
- structure of tables representing $P(X_i \mid \mathrm{pa}_{\mathrm{X_i}})$, entries unknown
- $k(\boldsymbol{u}, x)$ counts how often value combination $\boldsymbol{u}, x$ is observed in D

Output: $\hat{\theta}_{X_i|\mathrm{pa}_{\mathrm{X_i}}}$ entries for all $P(X_i \mid \mathrm{pa}_{\mathrm{X_i}})$

Approach: local likelihood function

$$L_{X_i}\left(\mathscr{D}; \boldsymbol{\theta}_{X_i|\mathrm{pa}_{\mathrm{X_i}}}\right) = \prod_{m=1}^{M} \theta_{x_i[m]|\mathrm{pa}_{\mathrm{X_i}}[\mathrm{m}]} =$$

$$= \prod_{\boldsymbol{u} \in \mathrm{val}(\mathrm{pa}_{X_i})} \left[\prod_{x \in \mathrm{val}(X_i)} \left(\theta_{x|\boldsymbol{u}}\right)^{k(\boldsymbol{u},x)} \right]$$

Multinomial MLE:

$$\hat{\theta}_{x|\boldsymbol{u}} = \frac{k(\boldsymbol{u}, x)}{k(\boldsymbol{u})}$$

MLE for Linear Gaussian Bayesian Network

$$P(X_i \mid \boldsymbol{u}) = \mathcal{N}\left(\beta_0 + \beta_1 u_1 + \ldots + \beta_l u_l; \sigma^2\right)$$

Closed Form Solution!

$$\ell_X\left(\mathcal{D}; \theta_{X|\mathrm{pa}_X}\right) = \log L_X(\mathcal{D}; \theta_{X|\mathrm{pa}_X}) =$$

$$= \sum_{m=1}^{M} \left[-\frac{1}{2} \log\left(2\pi\sigma^2\right) - \frac{1}{2\sigma^2} \left(\beta_0 + \beta_1 u_1[m] + \ldots + \beta_l u_l[m]\right)^2 \right]$$

Computing the partial derivatives $\dfrac{\partial}{\partial \beta_i}$ and setting the result to 0

leads to a set of linear equations that can be solved.

Data Fragmentation and Overfitting

- Data $\mathscr{D} = \left\{ \boldsymbol{x}[1] \ldots \boldsymbol{x}[M] \right\}$
- $\hat{\theta}_{x|\boldsymbol{u}}$ is estimated based on $\dfrac{M}{k(\boldsymbol{u})}$ data points
  - Possible value assignments $\boldsymbol{u}$ to pa_{X_i} grows
    - exponentially with number of parents
    - larger set of values $\longrightarrow$ larger basis for the exponent

Data Fragmentation leads to Overfitting.
Simpler network structures may prevent fragmentation and overfitting.

Example:
Naive Bayes classifier avoids complex structure

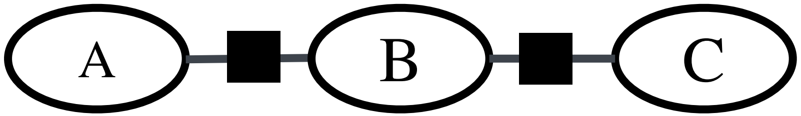
Summary

- For Bayesian Networks with disjoint sets of parameters in CPD, likelihood decomposes as product of local likelihood functions
- For table CPDs, local likelihood further decomposes as product of likelihood for multinomials – one for each parent combination
- For networks with shared CPDs (e.g. HMMs), statistics accumulate over all uses of CPDs

Question 2: Maximum Likelihood for BNs

3 Maximum Likelihood for Log-linear Models

Log-likelihood for Markov Networks



$$P(a,b,c;\Theta) = \frac{1}{Z(\Theta)}\phi_1(a,b)\phi_2(b,c)$$

log-likelihood

$$\ell\left(\mathcal{D};\theta\right) =$$

$$= \sum_{m=1}^{M}\left(\ln\phi_1(a[m],b[m]) + \ln\phi_2(b[m],c[m]) - \ln Z(\Theta)\right) =$$

$$= \sum_{a,b} k(A=a, B=b)\ln\phi_1(a,b) +$$

$$+ \sum_{b,c} k(B=b, C=c)\ln\phi_2(b,c) - M\ln Z(\Theta)$$

$$Z(\Theta) = \sum_{a,b,c}\phi_1(a,b)\phi_2(b,c)$$

Partition function couples parameters

- No decomposition of likelihood
- No closed form solution

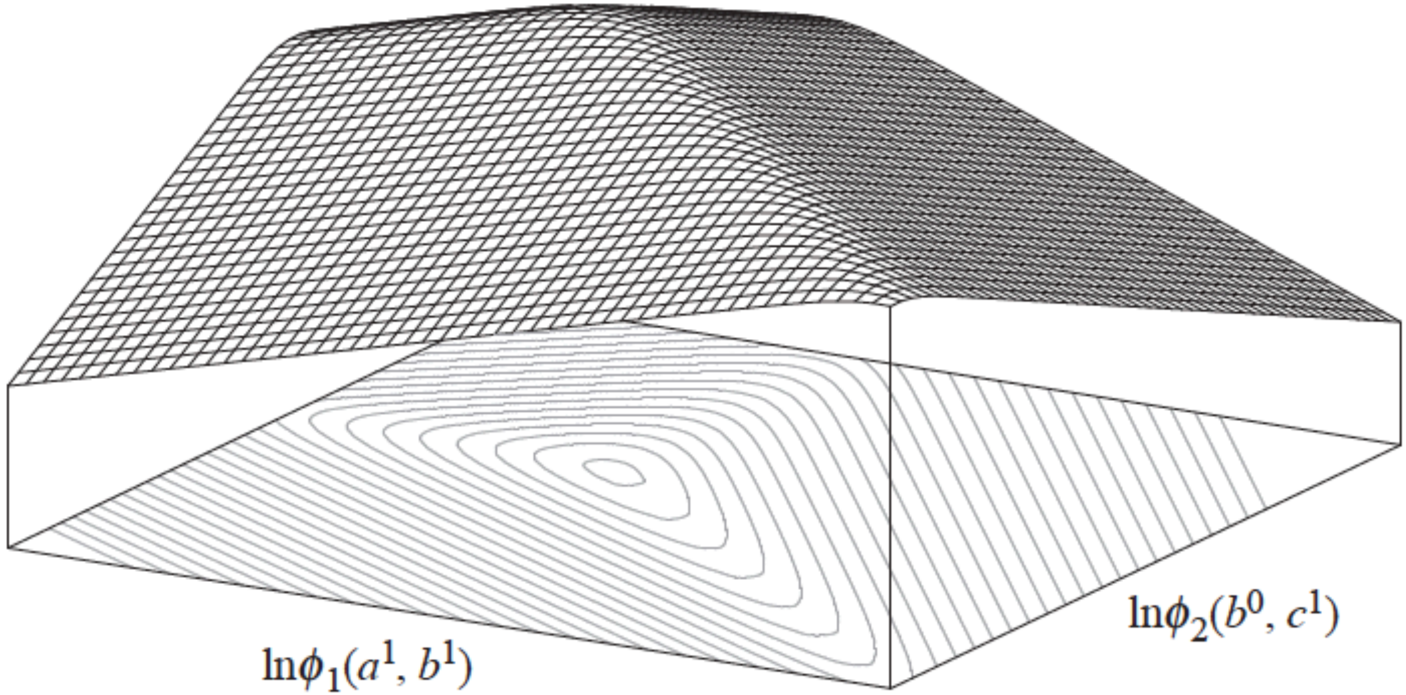


Figure 20.1 Log-likelihood surface for the Markov network A—B—C, as a function of $\ln \phi_1(a^1, b^1)$ **(**x**-axis) and** $\ln \phi_2(b^0, c^1)$ **(**y**-axis)**; all other parameters in both potentials are set to 1. Surface is viewed from the $(+\infty, +\infty)$ point toward the $(-, -)$ quadrant. The data set $\mathcal{D}$ has $M = 100$ instances, for which $M[a^1, b^1] = 40$ and $M[b^0, c^1] = 40$. (The other sufficient statistics are irrelevant, since all of the other log-parameters are 0.)

Log-linear models

Already used for CRFs

A distribution $P(\mathscr{X})$ is a log-linear model over a Markov Network $\mathscr{H}$ if it is associated with:

- a set of features $\mathscr{F} = \{f_1(D_1), \ldots f_l(D_l)\}$,
  - where each D_i is a complete subgraph in $\mathscr{H}$
  - where each feature $f_i : \mathrm{val}(D_i) \rightarrow \mathbb{R}$
- a set of weights $w_1, \ldots, w_l$

such that

$$P(X_1, \ldots, X_n) = \frac{1}{Z} e^{\left(-\sum_{i=1}^{l} w_i f_i(D_i)\right)}$$

The log-linear model expresses the factor product as the sum of features in an exponential. For some distributions this representation is more compact.

Log-likelihood for log-linear model

$$\ell\left(\mathscr{D};\theta\right) = \sum_{i=1}^{l} \theta_i \left(\sum_{m=1}^{M} f_i(x[m]) \right) - M \ln Z(\theta)$$

log-sum-exp:

$$\ln Z(\theta) = \ln \sum_{\boldsymbol{x} \in \mathscr{X}} e^{\left(\sum_i \theta_i f_i(\boldsymbol{x}) \right)}$$

exponentially large space

Log-Partition Function

Theorem:

$$\frac{\partial}{\partial \theta_i} \ln Z(\theta) = \mathbb{E}_\theta\left[f_i\right] = \sum_{\boldsymbol{x} \in \mathscr{X}} P_\theta(\boldsymbol{x}) f_i(\boldsymbol{x})$$

$$\frac{\partial^2}{\partial \theta_i \partial \theta_j} \ln Z(\theta) = \mathrm{Cov}_\theta\left[f_i; f_j\right]$$

The Hessian of the log-sum-exp is the covariance matrix, which is always positive semi-definite and therefore weakly convex

$$\frac{\partial}{\partial \theta_i} \ln Z(\theta) = \mathbb{E}_\theta\left[f_i\right]$$

Proof Part 1:

$$\frac{\partial}{\partial \theta_i} \ln Z(\theta) = \frac{1}{Z(\theta)} \sum_{\boldsymbol{x} \in \mathcal{X}} \frac{\partial}{\partial \theta_i} e^{\left(\sum_j \theta_j f_j(\boldsymbol{x})\right)} =$$

$$= \frac{1}{Z(\theta)} \sum_{\boldsymbol{x} \in \mathcal{X}} f_i(\boldsymbol{x}) e^{\left(\sum_j \theta_j f_j(\boldsymbol{x})\right)} = \sum_{\boldsymbol{x} \in \mathcal{X}} \frac{1}{Z(\theta)} e^{\left(\sum_j \theta_j f_j(\boldsymbol{x})\right)} f_i(\boldsymbol{x}) =$$

$$= \sum_{\boldsymbol{x} \in \mathcal{X}} P_\theta(\boldsymbol{x}) f_i(\boldsymbol{x}) = \mathbb{E}_\theta\left[f_i\right]$$

Proof Part 2 (Koller & Friedman,pp. 948)

Optimizing $\ell\left(\mathscr{D};\theta\right)$

Consider: $\ell\left(\mathscr{D};\theta\right) = \sum_{i=1}^{l} \theta_i \left(\sum_{m=1}^{M} f_i(x[m]) \right) - M \ln Z(\theta)$

The first term is linear in θ.

The second term is concave in θ („- convex“).

$\Longrightarrow$ The overall sum is concave

$\Longrightarrow$ local optimum is global optimum

$\Longrightarrow$ easy to optimize using gradient ascent

Maximum Likelihood Estimation

$$\frac{1}{M}\ell(\mathscr{D};\theta) = \sum_{i=1}^{l} \theta_i \left(\frac{1}{M} \sum_{m=1}^{M} f_i(x[m]) \right) - \ln Z(\theta)$$

$$\frac{\partial}{\partial \theta_i} \frac{1}{M}\ell(\mathscr{D};\theta) = \mathbb{E}_{\mathscr{D}}\left[f_i(\mathscr{X})\right] - \mathbb{E}_{\theta}\left[f_i\right]$$

Expectation of f_i in $\mathscr{D}$

Expectation of f_i in $P_{\Phi}(\mathscr{X})$

Theorem: $\hat{\theta}$ is the MLE if and only if $\mathbb{E}_{\mathscr{D}}\left[f_i(\mathscr{X})\right] = \mathbb{E}_{\hat{\theta}}\left[f_i\right]$

Computation of Gradient Ascent

$$\frac{\partial}{\partial \theta_i} \frac{1}{M} \ell\left(\mathscr{D}; \boldsymbol{\theta}\right) = \mathbb{E}_{\mathscr{D}}\left[f_i(\mathscr{X})\right] - \mathbb{E}_{\boldsymbol{\theta}}\left[f_i\right]$$

- Use gradient ascent
  - e.g. a quasi-Newton method like L-BFGS (
    - avoids expensive computation of the Hessian
  - e.g. stochastic gradient descent with momentum (see ML-Chapter 9)
- Needed for gradient: expected feature counts
  - in data ✓
  - relative to current model

  $\Longrightarrow$ expensive inference step at each gradient step 😫

Summary

- Partition function couples parameters in likelihood
- No closed form solution but convex optimization
  - solved using gradient ascent
- Gradient computation requires inference at each gradient step to compute expected feature counts
- Features are always within cluster in cluster graph
  - one calibration suffices for all feature expectations

Question 3:

Maximum Likelihood for Log-Linear Models

4 Maximum Likelihood for Conditional Random Fields

Estimation for CRFs

$$\mathscr{D} = \left\{ \boldsymbol{x}[m], \boldsymbol{y}[m] \right\}_{m=1}^{M}$$

$$P_\theta(Y \mid \boldsymbol{x}) = \frac{1}{Z_{\boldsymbol{x}}(\theta)} \tilde{P}_\theta(\boldsymbol{x}, Y) \qquad Z_{\boldsymbol{x}}(\theta) = \sum_{Y} \tilde{P}_\theta(\boldsymbol{x}, Y)$$

$$\ell_{Y|X}(\mathscr{D}; \theta) = \sum_{m=1}^{M} \ln P_\theta(\boldsymbol{y}[m] \mid \boldsymbol{x}[m]; \theta)$$

Considering single data point:

$$\ell_{Y|X}(\theta; \boldsymbol{x}[m], \boldsymbol{y}[m]) = \left(\sum_i \theta_i f_i(\boldsymbol{x}[m], \boldsymbol{y}[m]) \right) - \ln Z_{\boldsymbol{x}[m]}(\theta)$$

$$\frac{\partial}{\partial \theta_i} \frac{1}{M} \ell_{Y|X}(\mathscr{D}; \theta) = \frac{1}{M} \sum_{m=1}^{M} \left(f_i(\boldsymbol{x}[m], \boldsymbol{y}[m]) - \mathbb{E}_\theta\left[f_i \mid \boldsymbol{x}[m] \right] \right)$$

Basically the same formulas as before – with conditions $\boldsymbol{x}[m]$ thrown in

Comparing Computations

MRF

$$\frac{\partial}{\partial \theta_i} \frac{1}{M} \ell(\mathscr{D}; \theta) = \mathbb{E}_{\mathscr{D}}\left[f_i(\mathscr{X})\right] - \mathbb{E}_{\theta}\left[f_i\right]$$

- Requires inference at each gradient step

CRF

$$\frac{\partial}{\partial \theta_i} \frac{1}{M} \ell_{Y|\boldsymbol{X}}(\mathscr{D}; \theta) = \frac{1}{M} \sum_{m=1}^{M} \left(f_i(\boldsymbol{x}[m], \boldsymbol{y}[m]) - \mathbb{E}_{\theta}\left[f_i \mid \boldsymbol{x}[m]\right] \right)$$

- Requires inference for each $\boldsymbol{x}[m]$ at each gradient step

What is better?

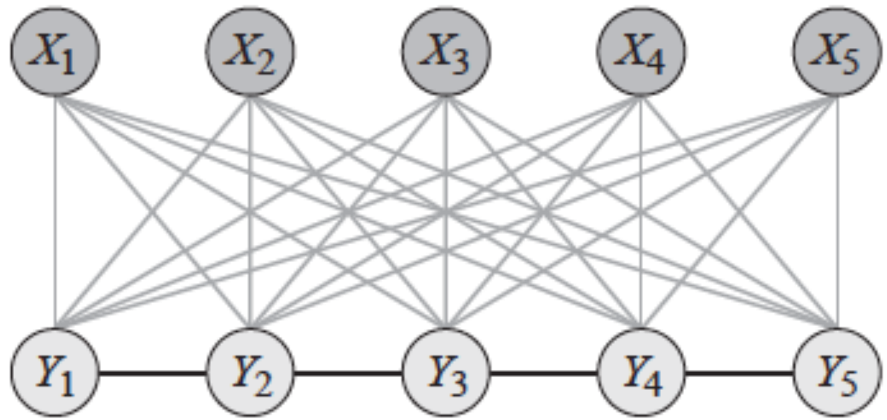


Figure 20.2 A highly connected CRF that allows simple inference when conditioned: The edges that disappear in the reduced Markov network after conditioning on X are marked in gray; the remaining edges form a simple linear chain.

Summary

- CRF learning very similar to MRF learning
  - likelihood function is concave
  - optimized using gradient ascent
- Gradient computation requires inference:
  - one per gradient step, data instance
    - cf: once per gradient step for MRFs
- But conditional model is often much simpler, so inference cost for CRF may even be lower

Question 4: Maximum Likelihood for CRFs

Universität Stuttgart
IPVS

Thank you!

Steffen Staab

E-Mail Steffen.staab@ipvs.uni-stuttgart.de
Telefon +49 (0) 711 685- To be defined
www. ipvs.uni-stuttgart.de/departments/ac/

Universität Stuttgart
Analytic Computing, IPVS
Universitätsstraße 32, 50569 Stuttgart