Probabilistic Graphical Models

parameter learning in undirected models

Siamak Ravanbakhsh

Winter 2021

Learning objectives

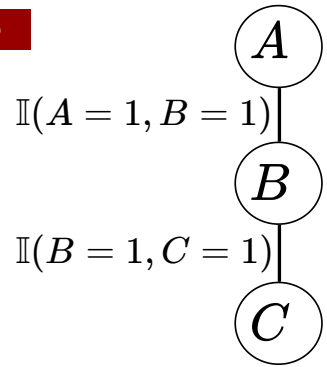
- the form of likelihood for undirected models
  - why is it difficult to optimize?
- conditional likelihood in undirected models
- different approximations for parameter learning
  - MAP inference and regularization
  - pseudo likelihood
  - pseudo moment-matching
  - contrastive learning
- difficulty of structure learning

Likelihood in MRFs

example

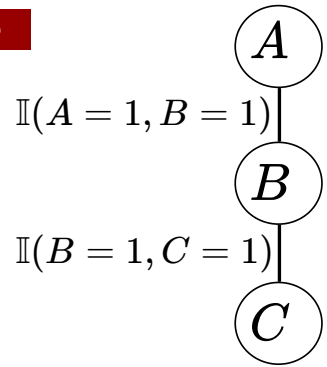
probability dist.

$p(A, B, C; \theta) = \frac{1}{Z} \exp(\theta_1 \mathbb{I}(A = 1, B = 1) + \theta_2 \mathbb{I}(B = 1, C = 1))$



Likelihood in MRFs

example



probability dist.

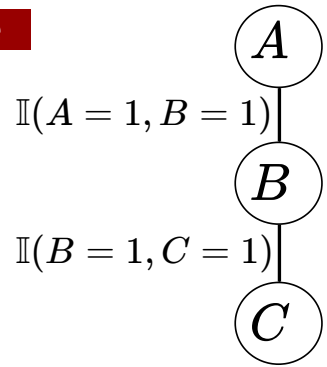
$p(A,B,C;\theta) = \frac{1}{Z}\exp(\theta_1 \mathbb{I}(A=1,B=1) + \theta_2 \mathbb{I}(B=1,C=1))$

observations $|\mathcal{D}| = 100$

- $\mathbb{E}_{\mathcal{D}}[\mathbb{I}(A=1,B=1)] = .4, \mathbb{E}_{\mathcal{D}}[\mathbb{I}(B=1,C=1)] = .4$

Likelihood in MRFs

example



probability dist.

$p(A,B,C;\theta) = \frac{1}{Z}\exp(\theta_1 \mathbb{I}(A=1,B=1) + \theta_2 \mathbb{I}(B=1,C=1))$

observations $|\mathcal{D}| = 100$

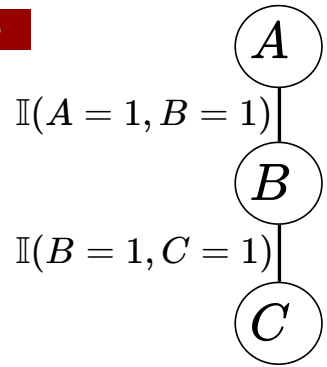
- $\mathbb{E}_{\mathcal{D}}[\mathbb{I}(A=1,B=1)] = .4, \mathbb{E}_{\mathcal{D}}[\mathbb{I}(B=1,C=1)] = .4$

log-likelihood: $\log p(\mathcal{D};\theta) = \sum_{a,b,c\in\mathcal{D}} \theta_1 \mathbb{I}(a=1,b=1) + \theta_2 \mathbb{I}(b=1,c=1) - 100 \log Z(\theta)$

$= 40\theta_1 + 40\theta_2 - 100 \log Z(\theta)$

Likelihood in MRFs

example



probability dist.

$p(A,B,C;\theta) = \frac{1}{Z}\exp(\theta_1 \mathbb{I}(A=1,B=1) + \theta_2 \mathbb{I}(B=1,C=1))$

observations $|\mathcal{D}| = 100$

- $\mathbb{E}_{\mathcal{D}}[\mathbb{I}(A=1,B=1)] = .4, \mathbb{E}_{\mathcal{D}}[\mathbb{I}(B=1,C=1)] = .4$

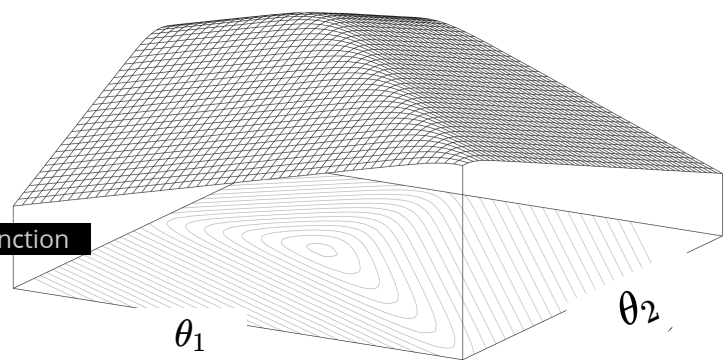
log-likelihood: $\log p(\mathcal{D};\theta) = \sum_{a,b,c\in\mathcal{D}} \theta_1 \mathbb{I}(a=1,b=1) + \theta_2 \mathbb{I}(b=1,c=1) - 100 \log Z(\theta)$

$= 40\theta_1 + 40\theta_2 - 100 \log Z(\theta)$

because of the partition function

the likelihood does not decompose

log-likelihood function



Likelihood in **linear exponential family** (log-linear models)

probability distribution $p(x;\theta) = \frac{1}{Z(\theta)} \exp(\theta^\top \phi(x))$

$\phi(x)$: sufficient statistics

Likelihood in **linear exponential family** (log-linear models)

probability distribution $p(x;\theta) = \frac{1}{Z(\theta)} \exp(\theta^\top \phi(x))$

$\phi(x)$: sufficient statistics

log-likelihood of $\mathcal{D}$ $\ell(\mathcal{D},\theta) = \log p(\mathcal{D};\theta) = \sum_{x\in\mathcal{D}} \theta^\top \phi(x) - |\mathcal{D}| \log Z(\theta)$

Likelihood in **linear exponential family** (log-linear models)

probability distribution $p(x;\theta) = \frac{1}{Z(\theta)} \exp(\theta^\top \phi(x))$

sufficient statistics

log-likelihood of $\mathcal{D}$ $\ell(\mathcal{D},\theta) = \log p(\mathcal{D};\theta) = \sum_{x\in\mathcal{D}} \theta^\top \phi(x) - |\mathcal{D}| \log Z(\theta)$

$\ell(\mathcal{D},\theta) = |\mathcal{D}| \left(\theta^\top \mathbb{E}_{\mathcal{D}}[\phi(x)] - \log Z(\theta)\right)$

expected sufficient statistics $\mu_{\mathcal{D}}$

Likelihood in linear exponential family (log-linear models)

probability distribution $p(x;\theta) = \frac{1}{Z(\theta)} \exp(\theta^\top \phi(x))$

sufficient statistics

log-likelihood of $\mathcal{D}$ $\ell(\mathcal{D},\theta) = \log p(\mathcal{D};\theta) = \sum_{x\in\mathcal{D}} \theta^\top \phi(x) - |\mathcal{D}| \log Z(\theta)$

$\ell(\mathcal{D},\theta) = |\mathcal{D}| \left(\theta^\top \mathbb{E}_{\mathcal{D}}[\phi(x)] - \log Z(\theta)\right)$

expected sufficient statistics $\mu_{\mathcal{D}}$

example

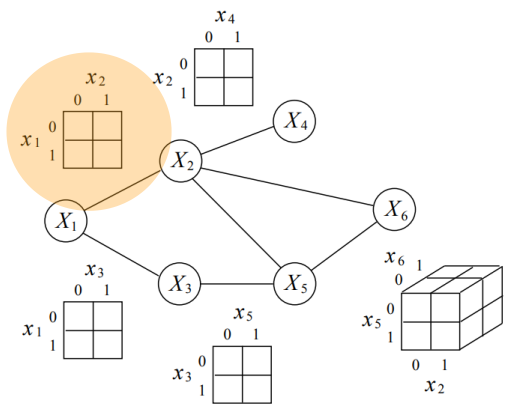


image: Michael Jordan's draft

expected sufficient statistics	params.
$\mathbb{E}_{\mathcal{D}}[\mathbb{I}(X_1=0, X_2=0)] = P(X_1=0, X_2=0)$	$\theta_{1,2,0,0}$
$\mathbb{E}_{\mathcal{D}}[\mathbb{I}(X_1=1, X_2=0)] = P(X_1=1, X_2=0)$	$\theta_{1,2,1,0}$
$\mathbb{E}_{\mathcal{D}}[\mathbb{I}(X_1=0, X_2=1)] = P(X_1=0, X_2=1)$	$\theta_{1,2,0,1}$
$\mathbb{E}_{\mathcal{D}}[\mathbb{I}(X_1=1, X_2=1)] = P(X_1=1, X_2=1)$	$\theta_{1,2,1,1}$

Likelihood in **linear exponential family** (log-linear models)

probability distribution $p(x;\theta) = \frac{1}{Z(\theta)} \exp(\theta^\top \phi(x))$

sufficient statistics

log-likelihood of $\mathcal{D}$ $\ell(\mathcal{D},\theta) = \log p(\mathcal{D};\theta) = \sum_{x\in\mathcal{D}} \theta^\top \phi(x) - |\mathcal{D}| \log Z(\theta)$

$\ell(\mathcal{D},\theta) = |\mathcal{D}| \left(\theta^\top \mathbb{E}_{\mathcal{D}}[\phi(x)] - \log Z(\theta)\right)$

expected sufficient statistics $\mu_{\mathcal{D}}$

$\log Z(\theta)$ has interesting properties

$\frac{\partial}{\partial\theta_i} \log Z(\theta) = \frac{\frac{\partial}{\partial\theta_i} \sum_x \exp(\theta^\top \phi(x))}{Z(\theta)} = \frac{1}{Z(\theta)} \sum_x \phi_i(x) \exp(\theta^\top \phi(x)) = \mathbb{E}_p[\phi_i(x)]$ SO $\nabla_\theta \log Z(\theta) = \mathbb{E}_\theta[\phi(x)]$

Likelihood in linear exponential family (log-linear models)

probability distribution $p(x;\theta) = \frac{1}{Z(\theta)} \exp(\theta^\top \phi(x))$

sufficient statistics

log-likelihood of $\mathcal{D}$ $\ell(\mathcal{D},\theta) = \log p(\mathcal{D};\theta) = \sum_{x\in\mathcal{D}} \theta^\top \phi(x) - |\mathcal{D}| \log Z(\theta)$

$\ell(\mathcal{D},\theta) = |\mathcal{D}| \left(\theta^\top \mathbb{E}_{\mathcal{D}}[\phi(x)] - \log Z(\theta)\right)$

expected sufficient statistics $\mu_{\mathcal{D}}$

$\log Z(\theta)$ has interesting properties

$\frac{\partial}{\partial\theta_i} \log Z(\theta) = \frac{\frac{\partial}{\partial\theta_i} \sum_x \exp(\theta^\top \phi(x))}{Z(\theta)} = \frac{1}{Z(\theta)} \sum_x \phi_i(x) \exp(\theta^\top \phi(x)) = \mathbb{E}_p[\phi_i(x)]$ SO $\nabla_\theta \log Z(\theta) = \mathbb{E}_\theta[\phi(x)]$

$\frac{\partial^2}{\partial\theta_i \partial\theta_j} \log Z(\theta) = \mathbb{E}[\phi_i(x)\phi_j(x)] - \mathbb{E}[\phi_i(x)]\mathbb{E}[\phi_j(x)] = Cov(\phi_i, \phi_j)$

so the Hessian matrix is positive definite → $\log Z(\theta)$ is convex

Likelihood in **linear exponential family** (log-linear models)

probability distribution $p(x;\theta) = \frac{1}{Z(\theta)} \exp(\theta^\top \phi(x))$

log-likelihood of $\mathcal{D}$ $\ell(\mathcal{D},\theta) = |\mathcal{D}| \underbrace{\left(\theta^\top \mathbb{E}_{\mathcal{D}}[\phi(x)]\right)}_{\text{linear in } \theta} \underbrace{- \log Z(\theta)\big)}_{\text{convex}}$

Likelihood in **linear exponential family** (log-linear models)

probability distribution $p(x;\theta) = \frac{1}{Z(\theta)} \exp(\theta^\top \phi(x))$

log-likelihood of $\mathcal{D}$ $\ell(\mathcal{D},\theta) = |\mathcal{D}| \big(\overbrace{\theta^\top \mathbb{E}_{\mathcal{D}}[\phi(x)]}^{\text{linear in } \theta} \overbrace{- \log Z(\theta)}^{\text{convex}}\big)$

$\overbrace{\qquad\qquad\qquad\qquad}^{\text{concave}}$

Likelihood in **linear exponential family** (log-linear models)

probability distribution $p(x;\theta) = \frac{1}{Z(\theta)} \exp(\theta^\top \phi(x))$

log-likelihood of $\mathcal{D}$ $\ell(\mathcal{D},\theta) = |\mathcal{D}| \big(\overbrace{\theta^\top \mathbb{E}_{\mathcal{D}}[\phi(x)]}^{\text{linear in } \theta} - \overbrace{\log Z(\theta)}^{\text{convex}}\big)$

$\overbrace{\qquad\qquad\qquad\qquad}^{\text{concave}}$

should be easy to maximize (?)

Likelihood in **linear exponential family** (log-linear models)

probability distribution $p(x;\theta) = \frac{1}{Z(\theta)} \exp(\theta^\top \phi(x))$

log-likelihood of $\mathcal{D}$ $\ell(\mathcal{D},\theta) = |\mathcal{D}| \big(\underbrace{\theta^\top \mathbb{E}_{\mathcal{D}}[\phi(x)]}_{\text{linear in } \theta} \underbrace{- \log Z(\theta)}_{\text{convex}}\big)$

$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad}_{\text{concave}}$

should be easy to maximize (?) NO!

Likelihood in **linear exponential family** (log-linear models)

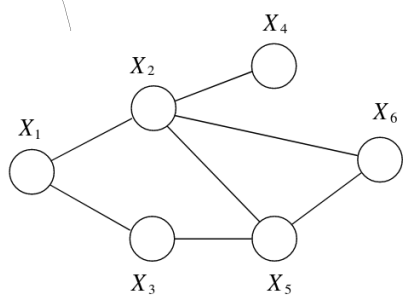
probability distribution $p(x;\theta) = \frac{1}{Z(\theta)} \exp(\theta^\top \phi(x))$

log-likelihood of $\mathcal{D}$ $\ell(\mathcal{D},\theta) = |\mathcal{D}| \big(\underbrace{\theta^\top \mathbb{E}_{\mathcal{D}}[\phi(x)]}_{\text{linear in } \theta} \underbrace{- \log Z(\theta)}_{\text{convex}}\big)$

concave

should be easy to maximize (?) NO!

- estimating $Z(\theta)$ is a difficult inference problem



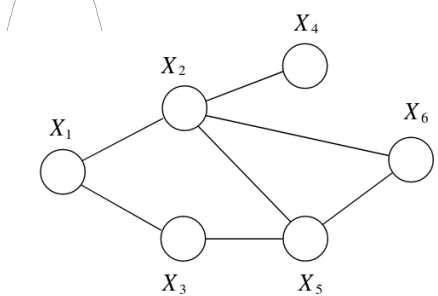
Likelihood in **linear exponential family** (log-linear models)

probability distribution $p(x;\theta) = \frac{1}{Z(\theta)} \exp(\theta^\top \phi(x))$

log-likelihood of $\mathcal{D}$ $\ell(\mathcal{D},\theta) = |\mathcal{D}| \underbrace{\left(\theta^\top \mathbb{E}_{\mathcal{D}}[\phi(x)]\right)}_{\text{linear in } \theta} \underbrace{- \log Z(\theta)}_{\text{convex}}$

$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad}_{\text{concave}}$

should be easy to maximize (?) NO!



- estimating $Z(\theta)$ is a difficult inference problem
- how about just using the gradient info?
  - involves inference as well $\nabla_\theta \log Z(\theta) = \mathbb{E}_\theta[\phi(x)]$
    - any combination of inference-gradient based optimization for learning undirected models

Moment matching for linear exponential family

probability distribution $p(x;\theta) = \frac{1}{Z(\theta)} \exp(\theta^\top \phi(x))$

log-likelihood of $\mathcal{D}$ $\ell(\mathcal{D},\theta) = |\mathcal{D}| \underbrace{\left(\underbrace{\theta^\top \mathbb{E}_{\mathcal{D}}[\phi(x)]}_{\text{linear in } \theta} - \underbrace{\log Z(\theta)}_{\text{convex}}\right)}_{\text{concave}}$

set its derivative to zero $\nabla_\theta \ell(\theta,\mathcal{D}) = |\mathcal{D}|(\mathbb{E}_{\mathcal{D}}[\phi(x)] - \mathbb{E}_{p_\theta}[\phi(x)]) = 0$

$$\Rightarrow \mathbb{E}_{p_\theta}[\phi(x)] = \mathbb{E}_{\mathcal{D}}[\phi(x)]$$

find the parameter θ

that results in the same expected sufficient statistics as the data

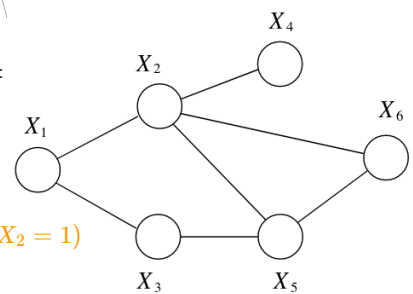
Moment matching for linear exponential family

probability distribution $p(x;\theta) = \frac{1}{Z(\theta)} \exp(\theta^\top \phi(x))$

log-likelihood of $\mathcal{D}$ $\ell(\mathcal{D},\theta) = |\mathcal{D}| \underbrace{\left(\underbrace{\theta^\top \mathbb{E}_{\mathcal{D}}[\phi(x)]}_{\text{linear in } \theta} - \underbrace{\log Z(\theta)}_{\text{convex}}\right)}_{\text{concave}}$

set its derivative to zero $\nabla_\theta \ell(\theta,\mathcal{D}) = |\mathcal{D}|(\mathbb{E}_{\mathcal{D}}[\phi(x)] - \mathbb{E}_{p_\theta}[\phi(x)]) =$

$\Rightarrow \mathbb{E}_{p_\theta}[\phi(x)] = \mathbb{E}_{\mathcal{D}}[\phi(x)]$



$p(X_1 = 0, X_2 = 1; \theta) = p_{\mathcal{D}}(X_1 = 0, X_2 = 1)$

find the parameter θ

that results in the same expected sufficient statistics as the data

Learning needs inference in an inner loop

maximizing the likelihood: $\arg\max_\theta \log p(\mathcal{D}|\theta)$

- gradient $\propto \mathbb{E}_{\mathcal{D}}[\phi(x)] - \mathbb{E}_{p_\theta}[\phi(x)]$
- optimality condition $\mathbb{E}_{\mathcal{D}}[\phi(x)] = \mathbb{E}_{p_\theta}[\phi(x)]$

$\mathbb{E}_{\mathcal{D}}[\phi(x)]$: easy to calculate

$\mathbb{E}_{p_\theta}[\phi(x)]$: inference in the graphical model

Learning needs inference in an inner loop

maximizing the likelihood: $\arg\max_\theta \log p(\mathcal{D}|\theta)$

- gradient $\propto \mathbb{E}_{\mathcal{D}}[\phi(x)] - \mathbb{E}_{p_\theta}[\phi(x)]$
- optimality condition $\mathbb{E}_{\mathcal{D}}[\phi(x)] = \mathbb{E}_{p_\theta}[\phi(x)]$

↓ easy to calculate ↓ inference in the graphical model

example: in discrete pairwise MRF $p_{\mathcal{D}}(x_i, x_j) = p(x_i, x_j; \theta) \quad \forall i, j \in \mathcal{E}$

↓ empirical marginals ↓ marginals in our current model

Learning needs inference in an inner loop

maximizing the likelihood: $\arg\max_\theta \log p(\mathcal{D}|\theta)$

- gradient $\propto \mathbb{E}_{\mathcal{D}}[\phi(x)] - \mathbb{E}_{p_\theta}[\phi(x)]$
- optimality condition $\mathbb{E}_{\mathcal{D}}[\phi(x)] = \mathbb{E}_{p_\theta}[\phi(x)]$

  $\mathbb{E}_{\mathcal{D}}[\phi(x)]$: easy to calculate

  $\mathbb{E}_{p_\theta}[\phi(x)]$: inference in the graphical model

example: in discrete pairwise MRF $p_{\mathcal{D}}(x_i, x_j) = p(x_i, x_j; \theta) \quad \forall i, j \in \mathcal{E}$

$p_{\mathcal{D}}(x_i, x_j)$: empirical marginals

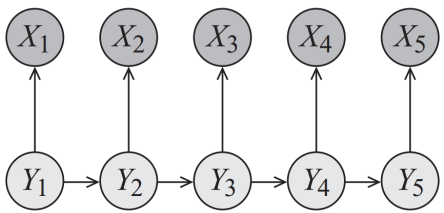
$p(x_i, x_j; \theta)$: marginals in our current model

what if exact inference is infeasible?

- learning with approx. inference often $\equiv$ exact optimization of approx. objective
  - use sampling, variational inference ...

Conditional training

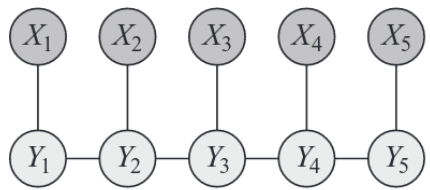
Recall generative vs. discriminative training



Hidden Markov Model (HMM) trained generatively

$$\ell(\mathcal{D}, \theta) = \sum_{(x,y)\in\mathcal{D}} \log p(x, y)$$

- easy to train the Bayes-net (assuming full observation)
- the likelihood decomposes



Conditional random fields (CRF)

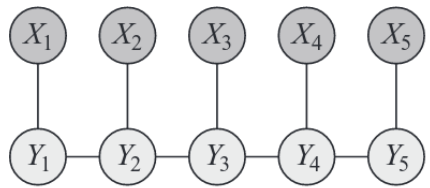
- trained discriminatively
- maximizing conditional log-likelihood

$$\ell_{Y|X}(\mathcal{D}, \theta) = \sum_{(x,y)\in\mathcal{D}} \log p(y|x)$$

- how to maximize this?

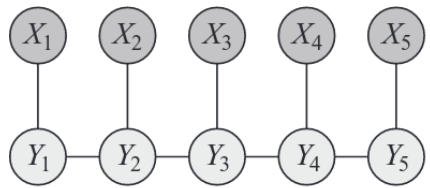
Conditional training

objective: $\arg\max_\theta \ell_{Y|X}(\mathcal{D}, \theta) = \arg\max_\theta \sum_{(x,y)\in\mathcal{D}} \log p(y|x)$



Conditional training

objective: $\arg\max_\theta \ell_{Y|X}(\mathcal{D}, \theta) = \arg\max_\theta \sum_{(x,y)\in\mathcal{D}} \log p(y|x)$



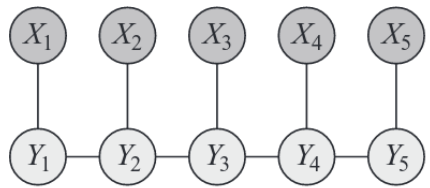
again consider the gradient

$$\nabla_\theta \ell_{Y|X}(\mathcal{D}, \theta) = \sum_{(x',y')\in\mathcal{D}} \phi(x', y') - \mathbb{E}_{p(.|x';\theta)}[\phi(x', y)]$$

- conditional expectation of sufficient statistics
- it is conditioned on the observed x'

Conditional training

objective: $\arg\max_\theta \ell_{Y|X}(\mathcal{D}, \theta) = \arg\max_\theta \sum_{(x,y)\in\mathcal{D}} \log p(y|x)$



again consider the gradient

$$\nabla_\theta \ell_{Y|X}(\mathcal{D}, \theta) = \sum_{(x',y')\in\mathcal{D}} \phi(x', y') - \mathbb{E}_{p(.|x';\theta)}[\phi(x', y)]$$

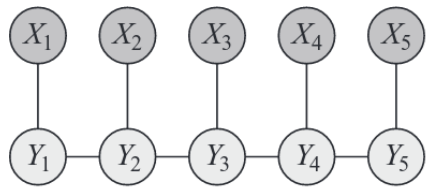
- conditional expectation of sufficient statistics
- it is conditioned on the observed x'

to obtain the gradient:

- for each instance $(x, y) \in \mathcal{D}$
  - run inference conditioned on x

Conditional training

objective: $\arg\max_\theta \ell_{Y|X}(\mathcal{D}, \theta) = \arg\max_\theta \sum_{(x,y)\in\mathcal{D}} \log p(y|x)$



again consider the gradient

$$\nabla_\theta \ell_{Y|X}(\mathcal{D}, \theta) = \sum_{(x',y')\in\mathcal{D}} \phi(x', y') - \mathbb{E}_{p(.|x';\theta)}[\phi(x', y)]$$

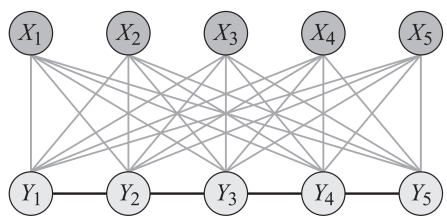
- conditional expectation of sufficient statistics
- it is conditioned on the observed x'

to obtain the gradient:

- for each instance $(x, y) \in \mathcal{D}$
  - run inference conditioned on x

compared to generative training in undirected models

pro: conditioning could simplify inference
con: have to run inference for each datapoint



inference on the reduced MRF
is easy in this case

Pseudo-moment matching

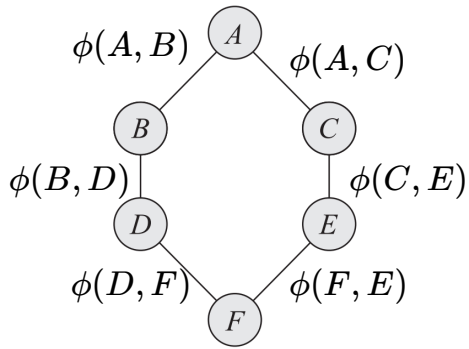
we want to set the parameters θ such that

if/when loopy BP converges:

$$p_{\mathcal{D}}(A, B) = \hat{p}(A, B; \theta), p_{\mathcal{D}}(B, D) = \hat{p}(B, D; \theta) \dots$$

empirical marginals

marginals using BP



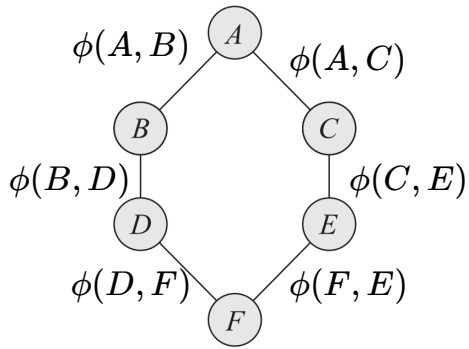
Pseudo-moment matching

we want to set the parameters θ such that

if/when loopy BP converges:

$$p_{\mathcal{D}}(A,B) = \hat{p}(A,B;\theta), p_{\mathcal{D}}(B,D) = \hat{p}(B,D;\theta)\dots$$

empirical marginals — marginals using BP



idea: use the reparametrization in BP

$$p(A,B,C,D,E,F) \propto \frac{\hat{p}(A,B)\dots\hat{p}(C,A)}{\hat{p}(A)\dots\hat{p}(F)}$$

product of clique marginals cancel the double-counts

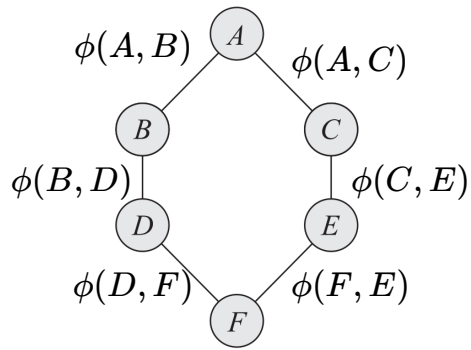
Pseudo-moment matching

we want to set the parameters θ such that

if/when loopy BP converges:

$$p_{\mathcal{D}}(A,B) = \hat{p}(A,B;\theta), p_{\mathcal{D}}(B,D) = \hat{p}(B,D;\theta)\dots$$

empirical marginals — marginals using BP



idea: use the reparametrization in BP

$$p(A,B,C,D,E,F) \propto \frac{\hat{p}(A,B)\dots\hat{p}(C,A)}{\hat{p}(A)\dots\hat{p}(F)}$$

product of clique marginals cancel the double-counts

set the factors using empirical marginals

- e.g., $\phi(A,B) \leftarrow p_{\mathcal{D}}(A,B)/p_{\mathcal{D}}(A)$
- each term in the numerator & denominator of should be used exactly once
- if we run BP on the resulting model we will have $p_{\mathcal{D}}(A,B) = \hat{p}(A,B;\theta), p_{\mathcal{D}}(B,D) = \hat{p}(B,D;\theta)\dots$

Pseudo-likelihood

log-likelihood: $\log p(\mathcal{D};\theta) = \sum_{x\in\mathcal{D}} \sum_i \log p(x_i|x_1,\ldots,x_{i-1};\theta)$ using the chain rule

Pseudo-likelihood

log-likelihood: $\log p(\mathcal{D};\theta) = \sum_{x\in\mathcal{D}} \sum_i \log p(x_i|x_1,\ldots,x_{i-1};\theta)$ using the chain rule

pseudo log-likelihood is an approximation

$$\log p(\mathcal{D};\theta) \approx \sum_{x\in\mathcal{D}} \sum_i \log p(x_i|\underbrace{x_{-i}}_{[x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n]};\theta)$$

Pseudo-likelihood

log-likelihood: $\log p(\mathcal{D};\theta) = \sum_{x\in\mathcal{D}} \sum_i \log p(x_i|x_1,\ldots,x_{i-1};\theta)$ using the chain rule

pseudo log-likelihood is an approximation

$$\log p(\mathcal{D};\theta) \approx \sum_{x\in\mathcal{D}} \sum_i \log p(x_i|\underbrace{x_{-i}}_{[x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n]};\theta)$$

$$\frac{p(x;\theta)}{\sum_{x_i} p(x;\theta)} = \frac{\tilde{p}(x;\theta)}{\sum_{x_i} \tilde{p}(x;\theta)}$$

eliminates the normalization constant

Pseudo-likelihood

log-likelihood: $\log p(\mathcal{D};\theta) = \sum_{x\in\mathcal{D}} \sum_i \log p(x_i|x_1,\ldots,x_{i-1};\theta)$ using the chain rule

pseudo log-likelihood is an approximation

$$\log p(\mathcal{D};\theta) \approx \sum_{x\in\mathcal{D}} \sum_i \log p(x_i|\underbrace{x_{-i}}_{[x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n]};\theta)$$

$$\frac{p(x;\theta)}{\sum_{x_i} p(x;\theta)} = \frac{\tilde{p}(x;\theta)}{\sum_{x_i} \tilde{p}(x;\theta)}$$ eliminates the normalization constant

it simplifies the gradient:

- instead of calculating $\sum_{x\in\mathcal{D}} \phi_k(x) - |\mathcal{D}| \mathbb{E}_{p_\theta}[\phi_k(x)]$ expensive!
- use $\sum_{x\in\mathcal{D}} \phi_k(x) - \sum_i \mathbb{E}_{p(.|x_{-i})}[\phi_k(x'_i, x_{-i})]$ can be further simplified using Markov blanket for each node...
- **upshot:** only conditional expectations are used (tractable!)

Pseudo-likelihood

log-likelihood: $\log p(\mathcal{D};\theta) = \sum_{x\in\mathcal{D}} \sum_i \log p(x_i|x_1,\ldots,x_{i-1};\theta)$ using the chain rule

pseudo log-likelihood is an approximation

$$\log p(\mathcal{D};\theta) \approx \sum_{x\in\mathcal{D}} \sum_i \log p(x_i|\underbrace{x_{-i}}_{[x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n]};\theta)$$

$$\frac{p(x;\theta)}{\sum_{x_i} p(x;\theta)} = \frac{\tilde{p}(x;\theta)}{\sum_{x_i} \tilde{p}(x;\theta)}$$ eliminates the normalization constant

it simplifies the gradient:

- instead of calculating $\sum_{x\in\mathcal{D}} \phi_k(x) - |\mathcal{D}| \mathbb{E}_{p_\theta}[\phi_k(x)]$ expensive!
- use $\sum_{x\in\mathcal{D}} \phi_k(x) - \sum_i \mathbb{E}_{p(.|x_{-i})}[\phi_k(x'_i, x_{-i})]$ can be further simplified using Markov blanket for each node...
- **upshot:** only conditional expectations are used (tractable!)

at the limit of large data (assuming we have the right model), this is exact!

Contrastive methods

log-likelihood: $\log p(\mathcal{D};\theta) = \sum_{x\in\mathcal{D}} \log \tilde{p}(x;\theta) - \log Z(\theta)$

increase the unnormalize prob. of the data

- it's easy to evaluate: e.g, $\theta^\top \phi(x)$

keep the total sum of unnormalized probabilities small $\log \sum_x \tilde{p}(x;\theta)$

- sum over exponentially many terms

Contrastive methods

log-likelihood: $\log p(\mathcal{D};\theta) = \sum_{x\in\mathcal{D}} \log \tilde{p}(x;\theta) - \log Z(\theta)$

increase the unnormalize prob. of the data

- it's easy to evaluate: e.g, $\theta^\top \phi(x)$

keep the total sum of unnormalized probabilities small $\log \sum_x \tilde{p}(x;\theta)$

- sum over exponentially many terms

contrastive methods: replace $\log Z(\theta)$ with a tractable alternative

- **contrastive divergence minimization:** only look at a small "*neighborhood*" of the data
- **margin-based training:** lower the probability of any assignment other than what was observed in the training data $\log \tilde{p}(y|x;\theta) - \max_{y'\neq y} \log \tilde{p}(y'|x;\theta)$
  - only for conditional training, why?

Local priors & regularization

max-likelihood can lead to over-fitting

Bayesian approach:

- in Bayes-nets: decomposed prior $p(\theta)$ → decomposed posterior $p(\theta \mid \mathcal{D})$
- in Markov nets: posterior does not decompose
  - *because of the the likelihood doesn't decomposed due to the partition function.*

Local priors & regularization

max-likelihood can lead to over-fitting

Bayesian approach:

- in Bayes-nets: decomposed prior $p(\theta)$ → decomposed posterior $p(\theta \mid \mathcal{D})$
- in Markov nets: posterior does not decompose
  - *because of the the likelihood doesn't decomposed due to the partition function.*

alternative to a full-Bayesian approach

MAP inference: maximize the log-posterior $\arg\max_\theta \log p(\mathcal{D}|\theta) + \underline{\log p(\theta)}$

- serves as a regularization
- does not have to be conjugate

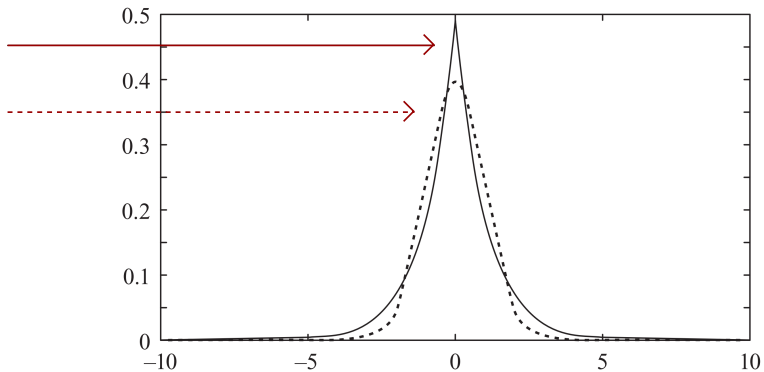
- does not model uncertainty
- sensitive to parametrization

Gaussian & Laplace priors

MAP inference: find the maximum of the posterior $\arg\max_\theta \log p(\mathcal{D}|\theta) + \log p(\theta)$

$p(\theta)$

- the product of univariate Laplace (L1 reg.)
- the product of univariate Gaussian (L2 reg.)



Gaussian & Laplace priors

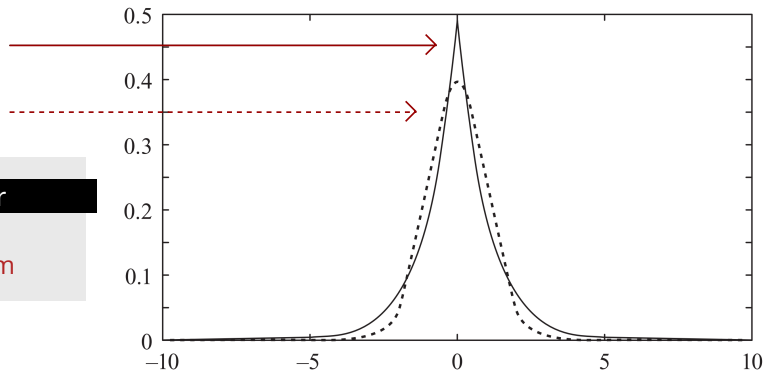
MAP inference: find the maximum of the posterior $\arg\max_\theta \log p(\mathcal{D}|\theta) + \log p(\theta)$

$p(\theta)$

- the product of univariate Laplace (L1 reg.)
- the product of univariate Gaussian (L2 reg.)

$p(\theta;\sigma) \propto \prod_i \exp(-\frac{\theta_i^2}{2\sigma^2}) \Rightarrow$ Gaussian prior

$\log p(\theta;\sigma) = -\frac{1}{2\sigma^2}\sum_i \theta_i^2 + c$ L2 regularization penalty term



Gaussian & Laplace priors

MAP inference: find the maximum of the posterior $\arg\max_\theta \log p(\mathcal{D}|\theta) + \log p(\theta)$

$p(\theta)$

- the product of univariate Laplace (L1 reg.)
- the product of univariate Gaussian (L2 reg.)

$p(\theta;\sigma) \propto \prod_i \exp(-\frac{\theta_i^2}{2\sigma^2}) \Rightarrow$

Gaussian prior

$\log p(\theta;\sigma) = -\frac{1}{2\sigma^2}\sum_i \theta_i^2 + c$

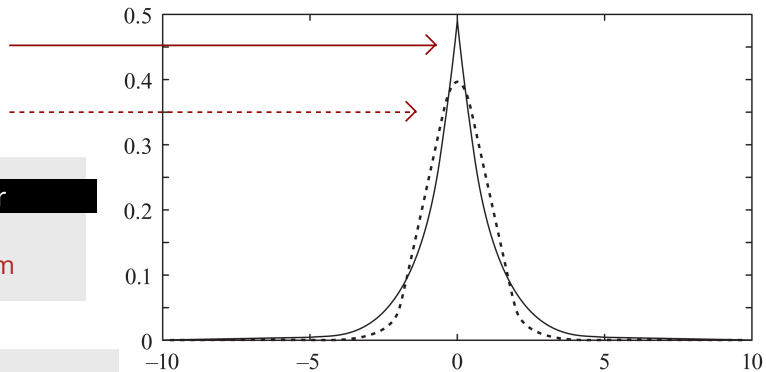
L2 regularization penalty term

$p(\theta;\beta) = \prod_i \frac{1}{2\beta}\exp(-\frac{|\theta_i|}{\beta}) \Rightarrow$

Laplace prior

$\log p(\theta;\beta) = -\frac{1}{\beta}\sum_i |\theta_i|$

L1 regularization penalty term



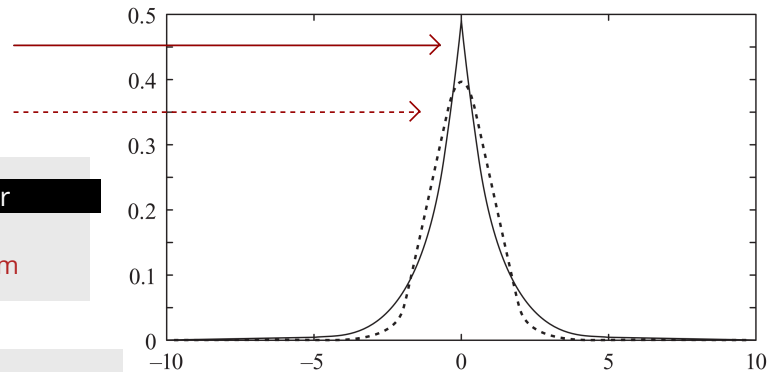
sparsity-inducing

Gaussian & Laplace priors

MAP inference: find the maximum of the posterior $\arg\max_\theta \log p(\mathcal{D}|\theta) + \log p(\theta)$

$p(\theta)$

- the product of univariate Laplace (L1 reg.)
- the product of univariate Gaussian (L2 reg.)



sparsity-inducing

$p(\theta;\sigma) \propto \prod_i \exp(-\frac{\theta_i^2}{2\sigma^2}) \Rightarrow$

Gaussian prior

$\log p(\theta;\sigma) = -\frac{1}{2\sigma^2}\sum_i \theta_i^2 + c$

L2 regularization penalty term

$p(\theta;\beta) = \prod_i \frac{1}{2\beta}\exp(-\frac{|\theta_i|}{\beta}) \Rightarrow$

Laplace prior

$\log p(\theta;\beta) = -\frac{1}{\beta}\sum_i |\theta_i|$

L1 regularization penalty term

- both of these penalize large parameter values
  - both reduce fluctuations in the density

$\log \frac{p(x;\theta)}{p(x',\theta)} = \theta^T(\phi(x) - \phi(x'))$

Structure learning

similarities to structure learning for Bayesian networks

Conditional independence test $X - Y \Rightarrow X \perp Y \mid MB(Y) \vee X \perp Y \mid MB(X)$

- similar to finding the *undirected skeleton* of a Bayes Net
- bound on the *size of Markov Blanket* (versus #parents in the BN)

Structure learning

similarities to structure learning for Bayesian networks

Conditional independence test $X - Y \Rightarrow X \perp Y \mid MB(Y) \vee X \perp Y \mid MB(X)$

- similar to finding the *undirected skeleton* of a Bayes Net
- bound on the *size of Markov Blanket* (versus #parents in the BN)

Maximizing a score:

- **likelihood score** or **Bayesian score (approx. BIC)**
- these scores do not decompose, need inference to estimate (expensive) ☹
  - learn models with low-tree width (efficient inference for estimating the likelihood) ☺
- Use greedy search
- **MAP score** (L1 regularized log-likelihood)
  - convex problem
  - introduce features 1-by-1 until convergence ☺

Summary

- parameter learning in MRFs is difficult
  - normalization constant ties the parameters together
    - likelihood does not decompose
    - Bayesian inference is also difficult

Summary

- parameter learning in MRFs is difficult
  - normalization constant ties the parameters together
    - likelihood does not decompose
    - Bayesian inference is also difficult
- (conditional) log-likelihood is convex
  - gradient steps: need inference on the current model
  - global optima satisfies moment-matching condition
  - combine inference methods + gradient descent for learning

Summary

- parameter learning in MRFs is difficult
  - normalization constant ties the parameters together
    - likelihood does not decompose
    - Bayesian inference is also difficult
- (conditional) log-likelihood is convex
  - gradient steps: need inference on the current model
  - global optima satisfies moment-matching condition
  - combine inference methods + gradient descent for learning
- alternative approaches:
  - *pseudo moment matching, pseudo likelihood, contrastive divergence, margin-based training*